Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

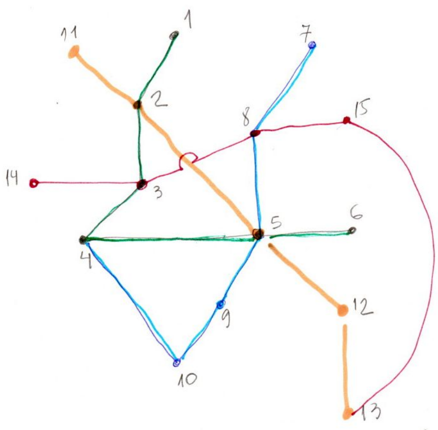
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

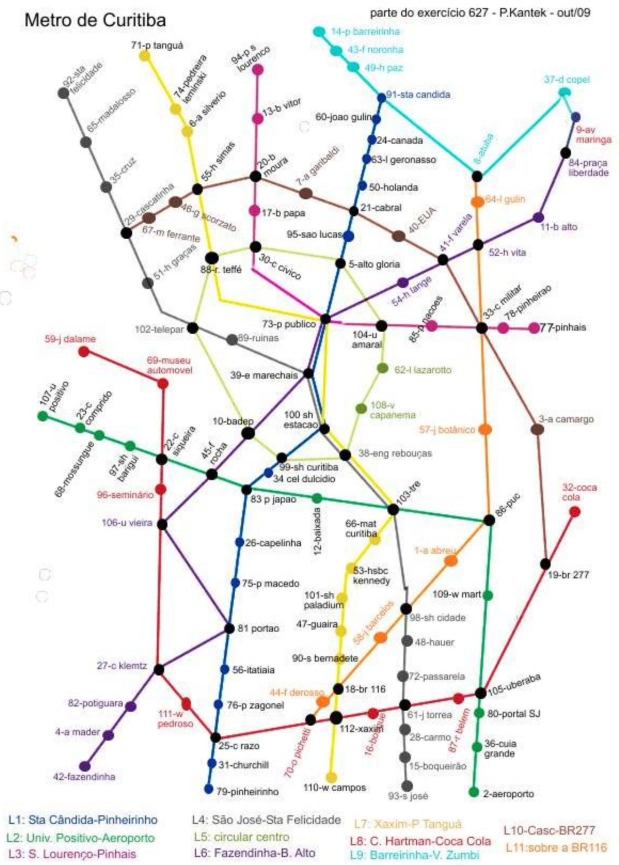
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] ← 1 e também MADJ[27;83] ← 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 23-campo comprido e 91-santa cândida ?
2. Qual o número da quinta estação na rota mínima entre a estação 58-joão barcelos e a estação 94-são lourenço ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75565 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

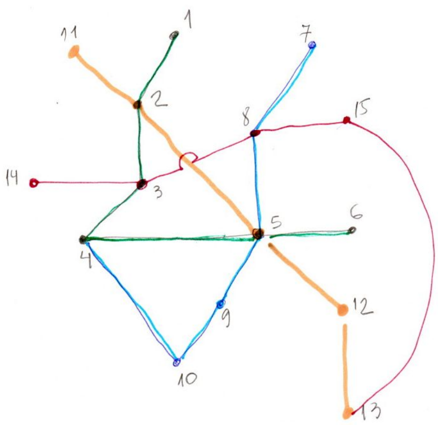
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

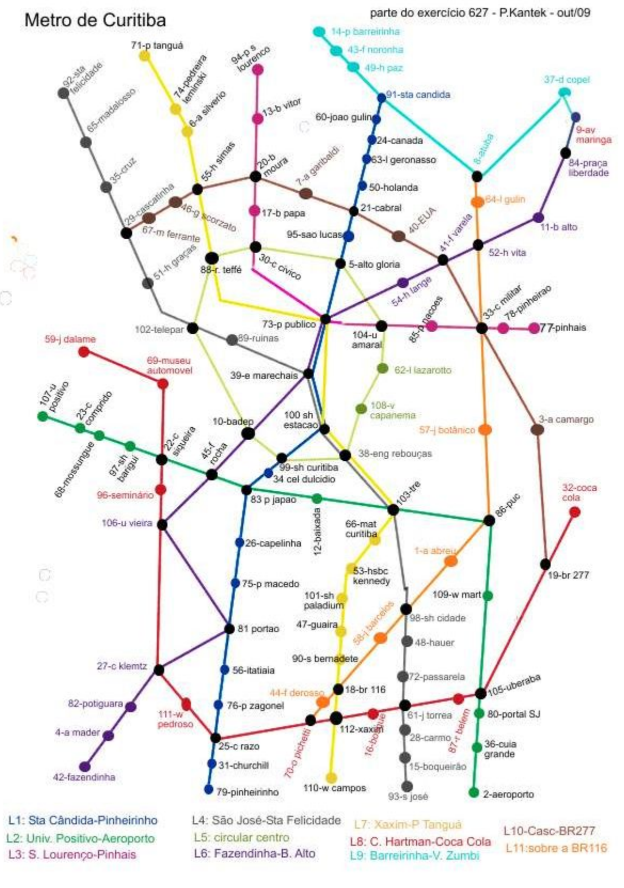
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 61-joaquim torrea e 107-universidade positivo ?
2. Qual o número da quinta estação na rota mínima entre a estação 43-fernando noronha e a estação 86-puc ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75572 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

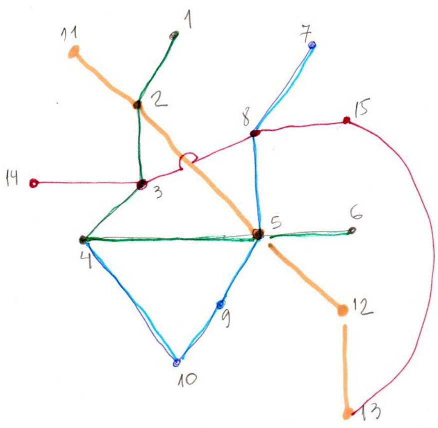
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

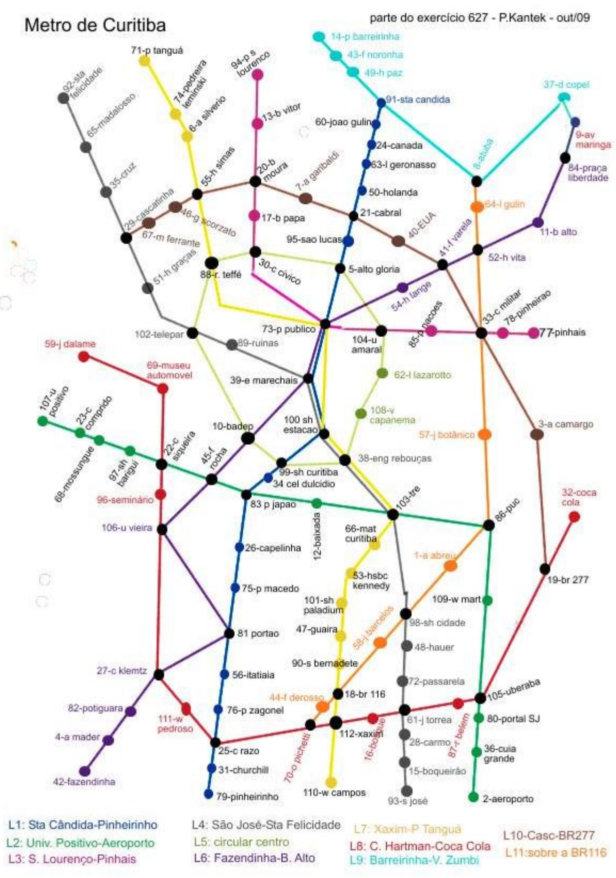
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 97-shopping barigui e 25-capão raso ?
2. Qual o número da sexta estação na rota mínima entre a estação 112-xaxim e a estação 50-holanda ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75589 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

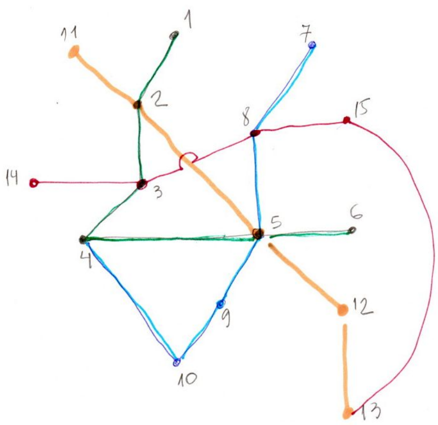
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

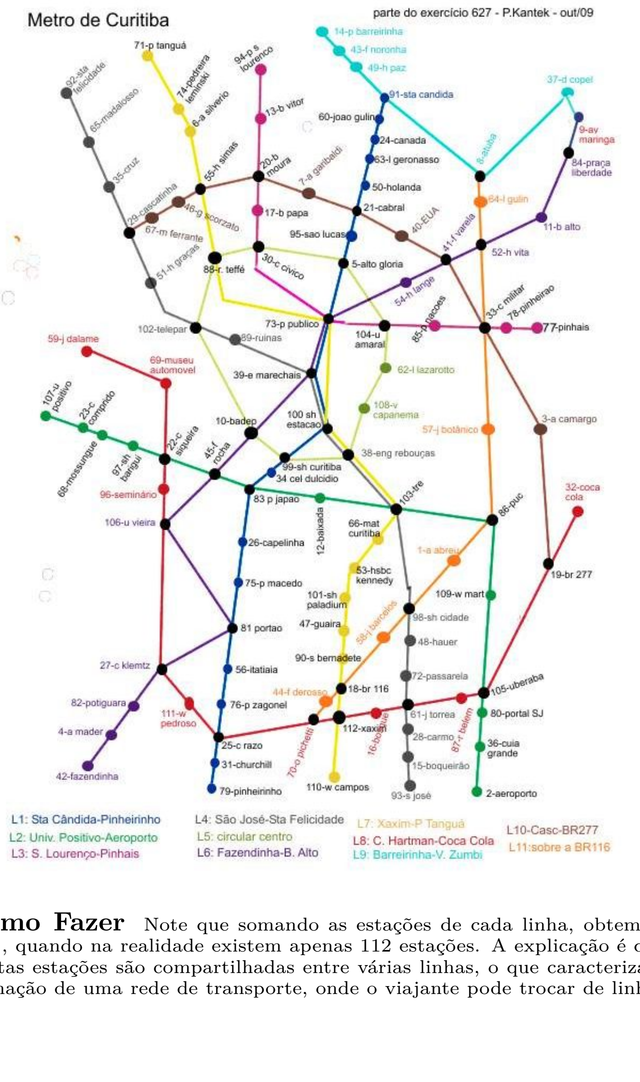
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 31-churchill e 76-pedro zagonel ?

2. Qual o número da quarta estação na rota mínima entre a estação 101-shopping paladium e a estação 24-canadá ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75596 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

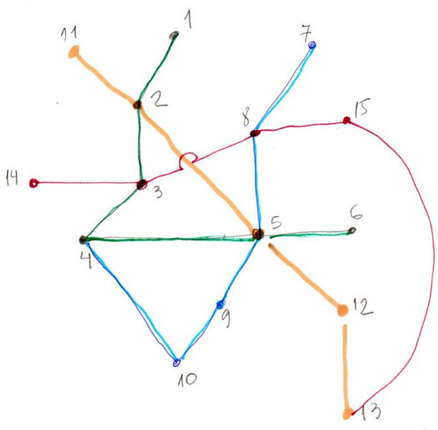
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô
Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

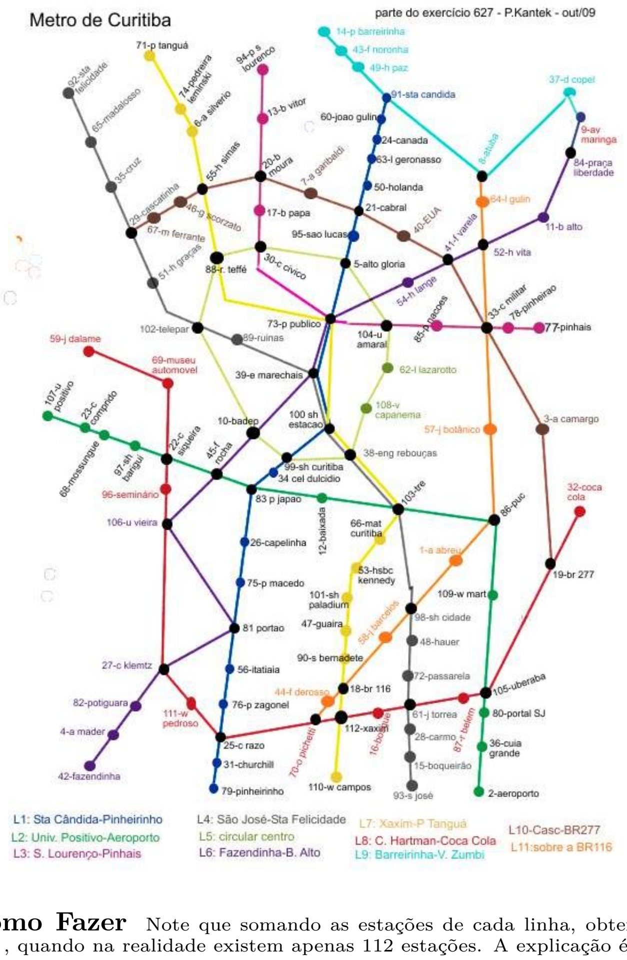
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 43-fernando noronha e 31-churchill ?
2. Qual o número da sexta estação na rota mínima entre a estação 37-depósito copel e a estação 108-vila capanema ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75608 - /

UFPR-UP-UTFPR-PUC/Pr Sistemas de Informação 16/09/2024 - 09:47:59.3
Algoritmos Prof Dr P Kantek (pkantek@gmail.com)
Achando o menor caminho no metrô de Curitiba VIVXq86a V: 1.02
75615 IAN DOLABELLA OHATA
24JOG409 - 6 ______ / ______ / ______

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

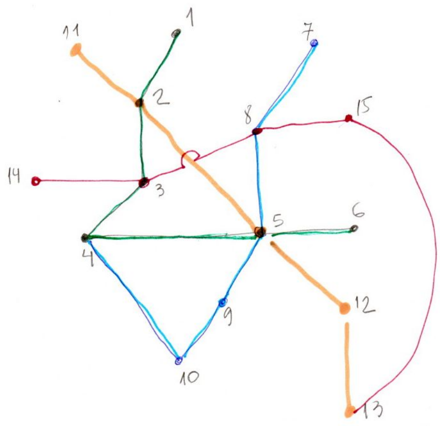
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \to 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \to 2 \to 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \to 2 \to 3 \to 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \to 7 = 11 \to 2 \to 3 \to 8 \to 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

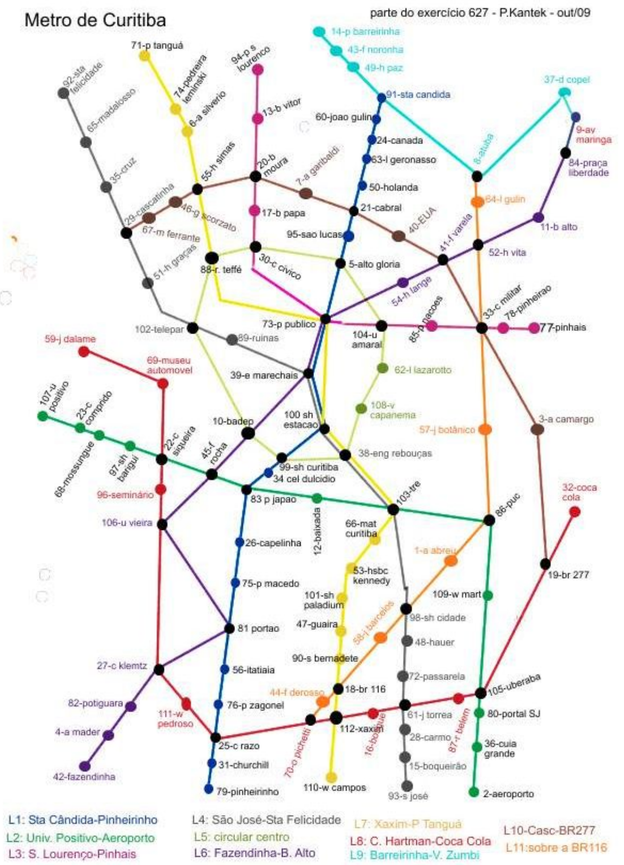
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 29-cascatinha e 26-capelinha ?
2. Qual o número da quinta estação na rota mínima entre a estação 54-hugo lange e a estação 93-são josé ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75615 - /

UFPR-UP-UTFPR-PUC/Pr Sistemas de Informação 16/09/2024 - 09:47:59.3
Algoritmos Prof Dr P Kantek (pkantek@gmail.com)
Achando o menor caminho no metrô de Curitiba VIVXq86a V: 1.02
75622 JOAO BONFIN LINO
24JOG409 - 7 ______ / ______ / ______

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

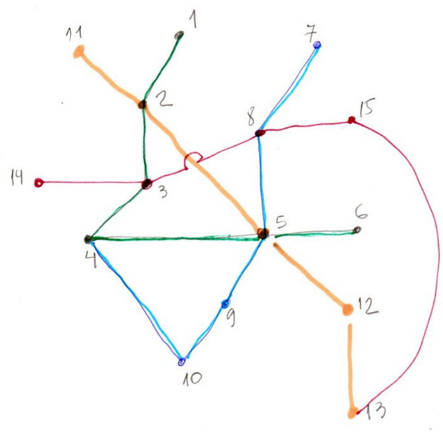
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

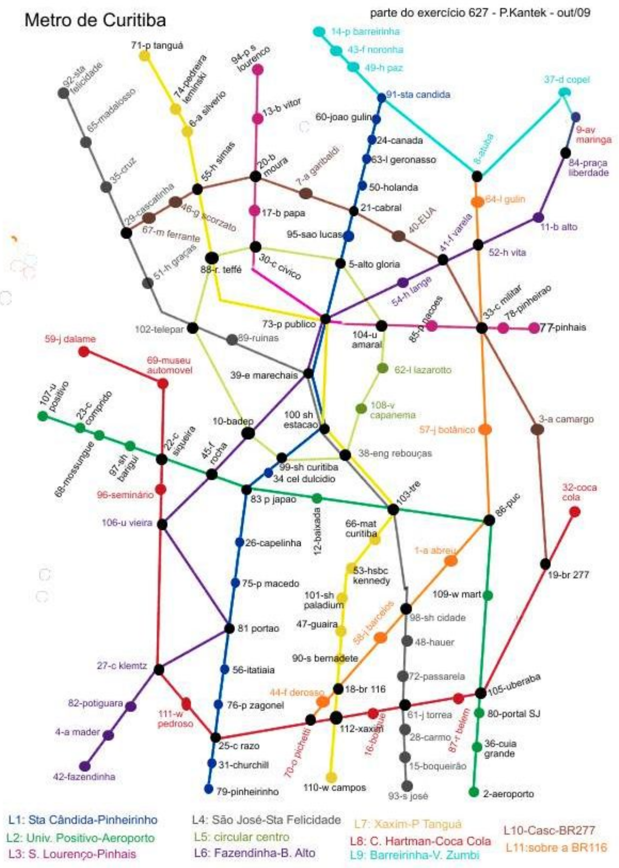
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 68-mossunguê e 49-hildo paz ?

2. Qual o número da quinta estação na rota mínima entre a estação 52-hospital vita e a estação 27-carlos klemtz ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75622 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

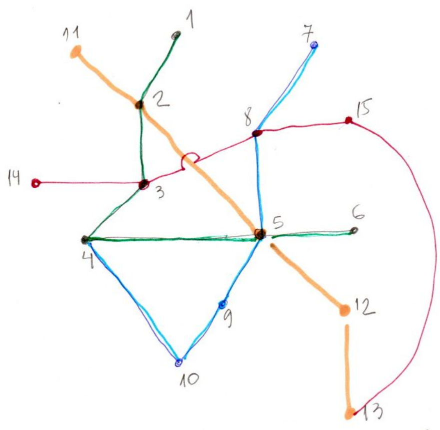
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

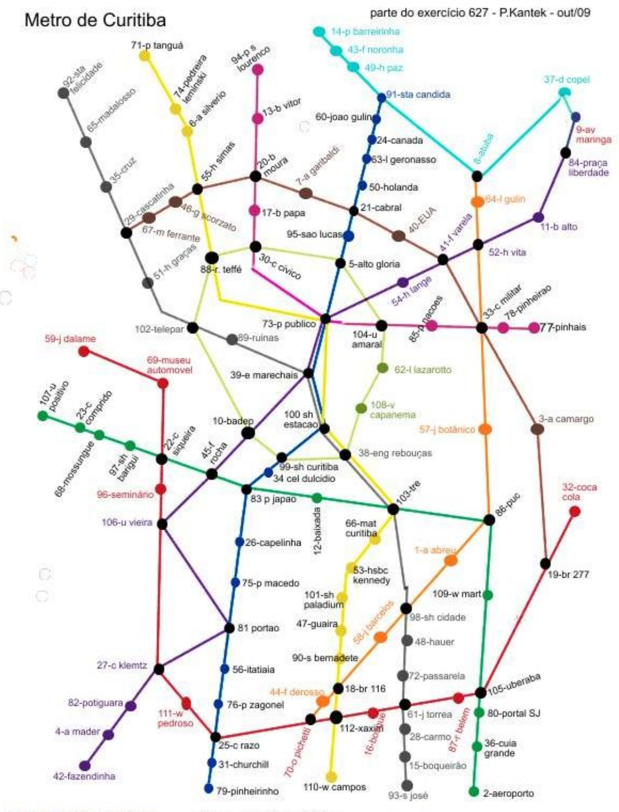
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 66-maternidade curitiba e 82-potiguara ?
2. Qual o número da quarta estação na rota mínima entre a estação 4-algacyr mader e a estação 105-uberaba ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75639 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

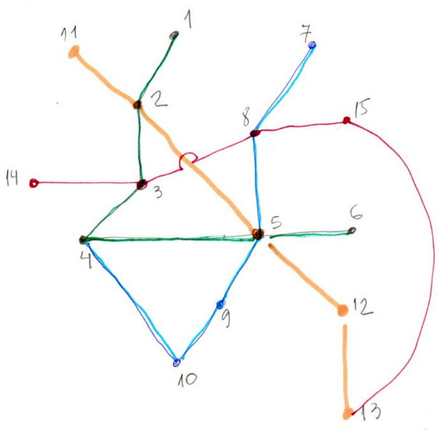
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô
Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

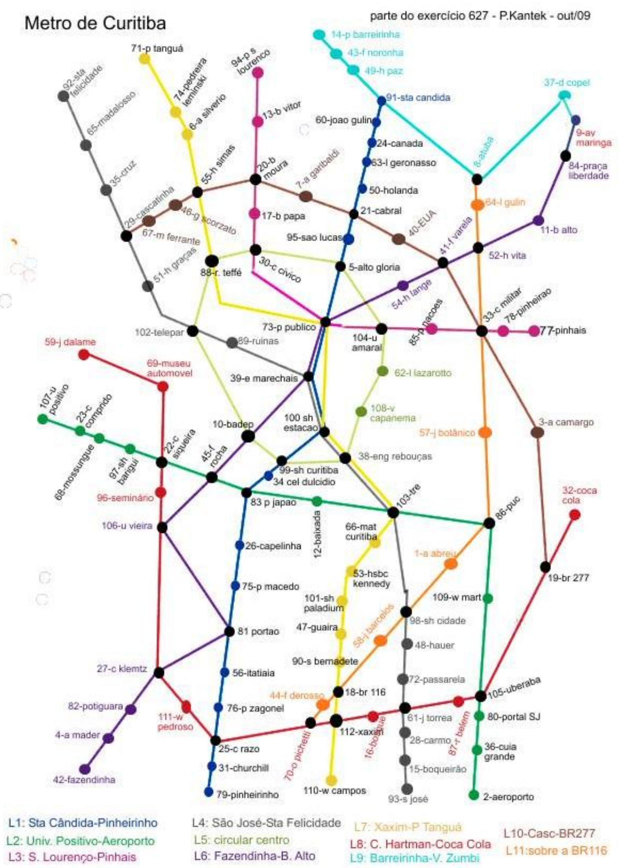
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] ← 1 e também MADJ[27;83] ← 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 81-portão e 33-colégio militar ?
2. Qual o número da quarta estação na rota mínima entre a estação 43-fernando noronha e a estação 36-cuia grande ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75646 - /

UFPR-UP-UTFPR-PUC/Pr Sistemas de Informação 16/09/2024 - 09:47:59.3
Algoritmos Prof Dr P Kantek (pkantek@gmail.com)
Achando o menor caminho no metrô de Curitiba VIVXq86a V: 1.02
75653 JOAO PEDRO MARIANO SUEKE
24JOG409 - 10 ______ / ______ / ______

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

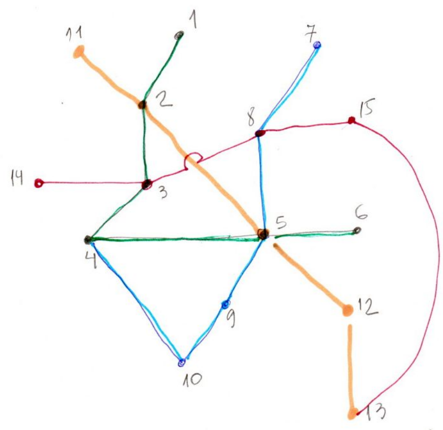
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

```
                        1 1 1 1 1 1
     1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
     - - - - - - - - - - - - - - -
 1-  0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
 2-  1 0 1 0 1 0 0 0 0 0 1 0 0 0 0
 3-  0 1 0 1 0 0 0 1 0 0 0 0 0 1 0
 4-  0 0 1 0 1 0 0 0 0 1 0 0 0 0 0
 5-  0 1 0 1 0 1 0 1 1 0 0 1 0 0 0
 6-  0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
 7-  0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
 8-  0 0 1 0 1 0 1 0 0 0 0 0 0 0 1
 9-  0 0 0 0 1 0 0 0 0 1 0 0 0 0 0
10-  0 0 0 1 0 0 0 0 1 0 0 0 0 0 0
11-  0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
12-  0 0 0 0 1 0 0 0 0 0 0 0 1 0 0
13-  0 0 0 0 0 0 0 0 0 0 0 1 0 0 1
14-  0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
15-  0 0 0 0 0 0 0 1 0 0 0 0 1 0 0
```

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

```
                        1 1 1 1 1 1
     1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
     - - - - - - - - - - - - - - -
 1-  0 1 2 3 2 3 4 3 3 4 2 3 4 3 4
 2-  1 0 1 2 1 2 3 2 2 3 1 2 3 2 3
 3-  2 1 0 1 2 3 2 1 3 2 2 3 3 1 2
 4-  3 2 1 0 1 2 3 2 2 1 3 2 3 2 3
 5-  2 1 2 1 0 1 2 1 1 2 2 1 2 3 2
 6-  3 2 3 2 1 0 3 2 2 3 3 2 3 4 3
 7-  4 3 2 3 2 3 0 1 3 4 4 3 3 3 2
 8-  3 2 1 2 1 2 1 0 2 3 3 2 2 2 1
 9-  3 2 3 2 1 2 3 2 0 1 3 2 3 4 3
10-  4 3 2 1 2 3 4 3 1 0 4 3 4 3 4
11-  2 1 2 3 2 3 4 3 3 4 0 3 4 3 4
12-  3 2 3 2 1 2 3 2 2 3 3 0 1 4 2
13-  4 3 3 3 2 3 3 2 3 4 4 1 0 4 1
14-  3 2 1 2 3 4 3 2 4 3 3 4 4 0 3
15-  4 3 2 3 2 3 2 1 3 4 4 2 1 3 0
```

Com a seguinte matriz de roteamento

```
      1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
      -  -  -  -  -  -  -  -  - -- -- -- -- -- --
 1-   1  2  2  2  2  2  2  2  2  2  2  2  2  2  2
 2-   1  2  3  3  5  5  3  3  5  3 11  5  5  3  3
 3-   2  2  3  4  2  2  8  8  2  4  2  2  8 14  8
 4-   3  3  3  4  5  5  3  3  5 10  3  5  5  3  3
 5-   2  2  2  4  5  6  8  8  9  4  2 12 12  2  8
 6-   5  5  5  5  5  6  5  5  5  5  5  5  5  5  5
 7-   8  8  8  8  8  8  7  8  8  8  8  8  8  8  8
 8-   3  3  3  3  5  5  7  8  5  3  3  5 15  3 15
 9-   5  5  5  5  5  5  5  5  9 10  5  5  5  5  5
10-   4  4  4  4  4  4  4  4  9 10  4  4  4  4  4
11-   2  2  2  2  2  2  2  2  2  2 11  2  2  2  2
12-   5  5  5  5  5  5  5  5  5  5  5 12 13  5 13
13-  12 12 15 12 12 12 15 15 12 12 12 12 13 15 15
14-   3  3  3  3  3  3  3  3  3  3  3  3  3 14  3
15-   8  8  8  8  8  8  8  8  8  8  8 13 13  8 15
```

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \to 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \to 2 \to 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \to 2 \to 3 \to 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \to 7 = 11 \to 2 \to 3 \to 8 \to 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

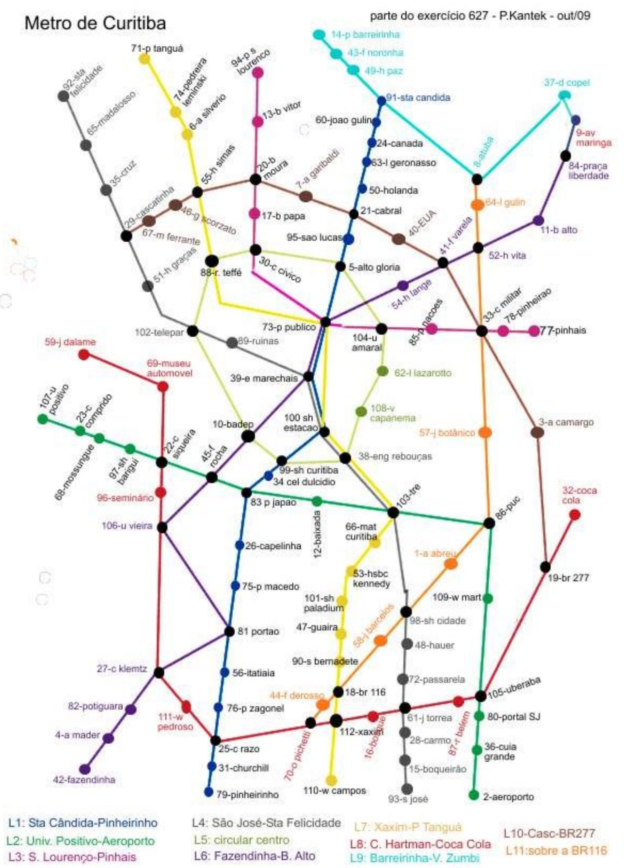
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 66-maternidade curitiba e 23-campo comprido ?
2. Qual o número da quinta estação na rota mínima entre a estação 24-canadá e a estação 76-pedro zagonel ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75653 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

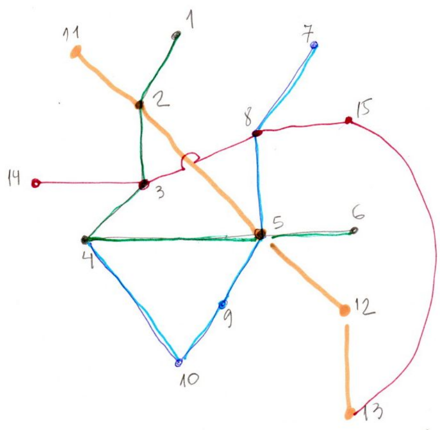
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

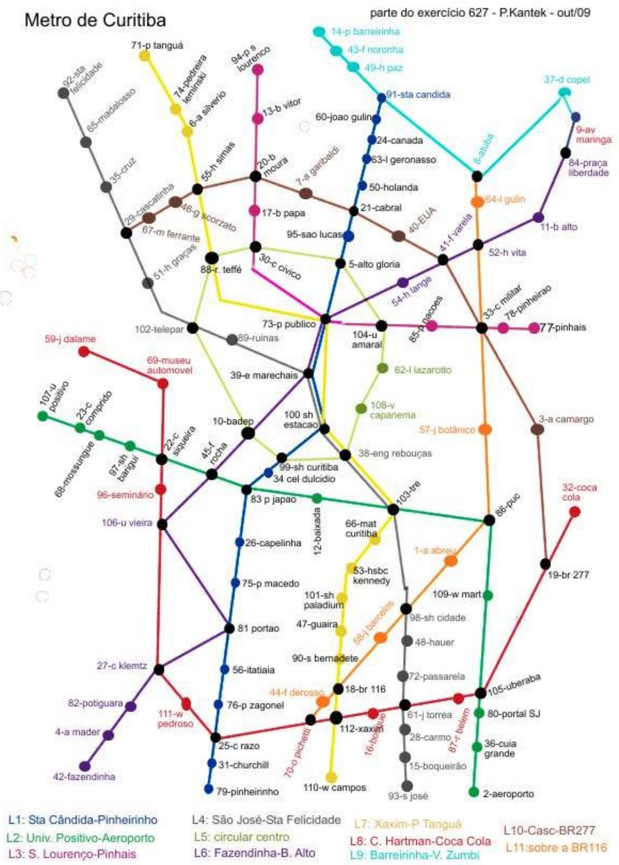
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 106-ulisses vieira e 29-cascatinha ?
2. Qual o número da sexta estação na rota mínima entre a estação 52-hospital vita e a estação 44-francisco derosso ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75660 - /

UFPR-UP-UTFPR-PUC/Pr Sistemas de Informação 16/09/2024 - 09:47:59.3
Algoritmos Prof Dr P Kantek (pkantek@gmail.com)
Achando o menor caminho no metrô de Curitiba VIVXq86a V: 1.02
75765 LUCAS ANTOSZCZYSZEN
24JOG409 - 12 ______ / ______ / ______

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

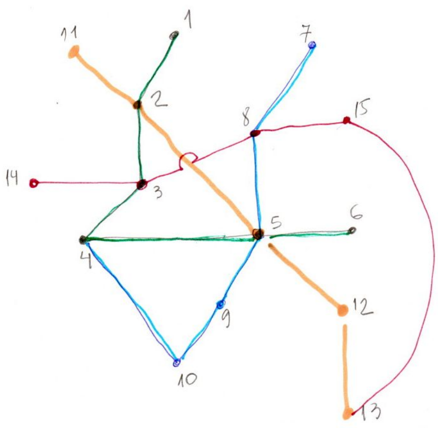
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

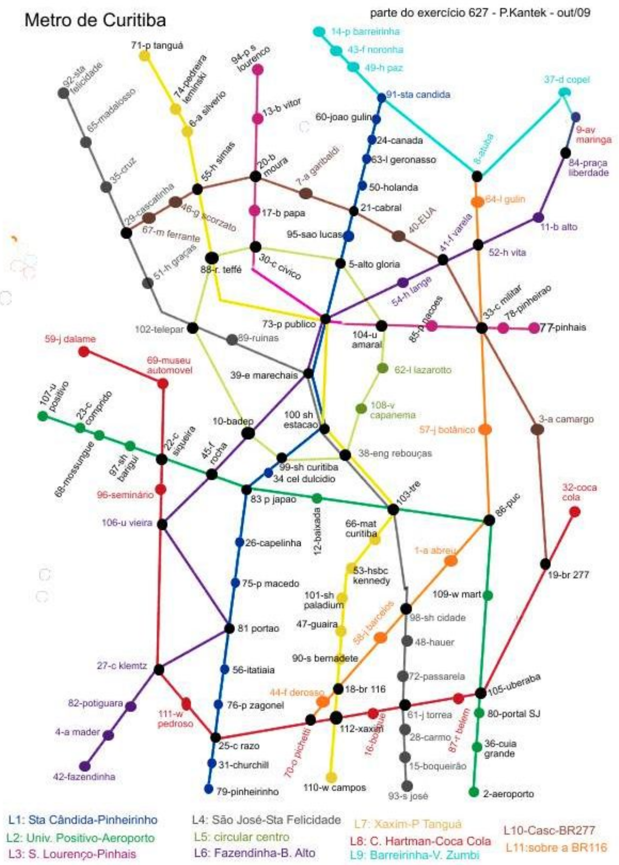
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 110-waldemar campos e 28-carmo ?

2. Qual o número da quinta estação na rota mínima entre a estação 112-xaxim e a estação 77-pinhais ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75765 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

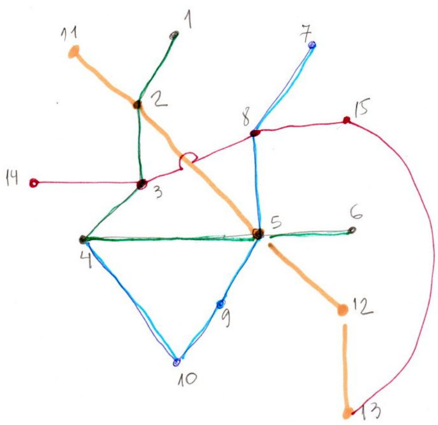
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \to 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \to 2 \to 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \to 2 \to 3 \to 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \to 7 = 11 \to 2 \to 3 \to 8 \to 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

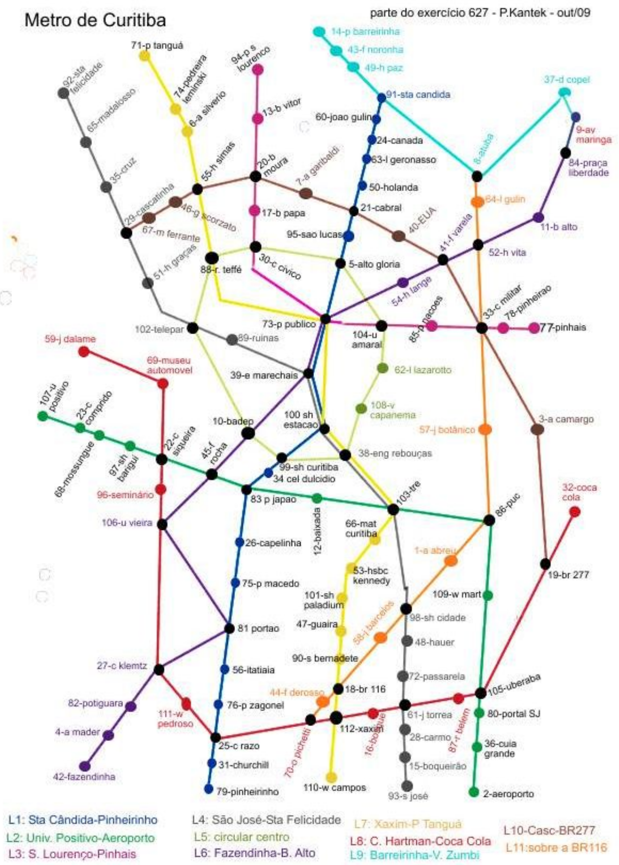
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 85-praça nações e 101-shopping paladium ?
2. Qual o número da sexta estação na rota mínima entre a estação 50-holanda e a estação 23-campo comprido ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75677 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

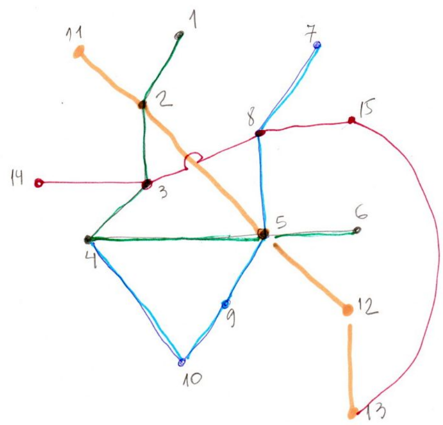
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

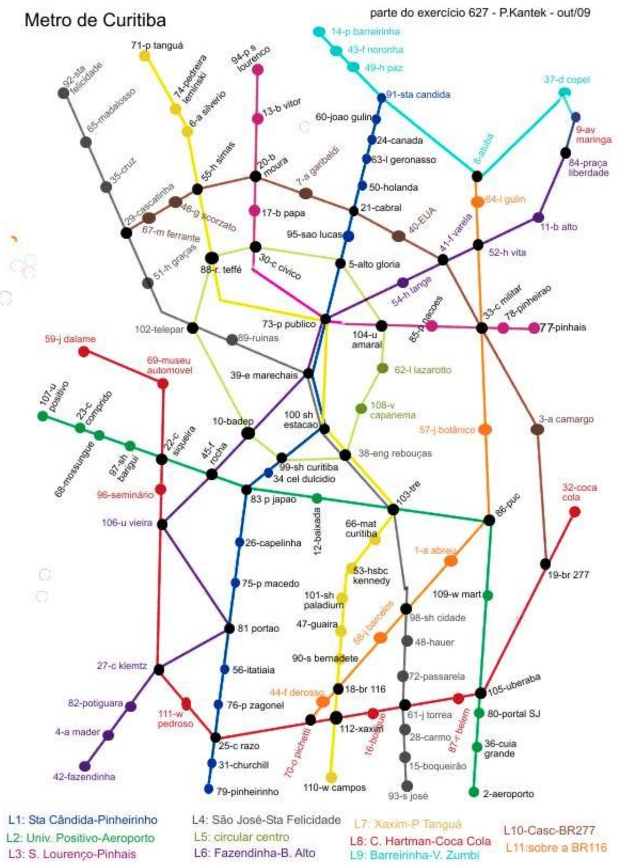
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 27-carlos klemtz e 68-mossunguê ?
2. Qual o número da sexta estação na rota mínima entre a estação 24-canadá e a estação 82-potiguara ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75684 - /

UFPR-UP-UTFPR-PUC/Pr Sistemas de Informação 16/09/2024 - 09:47:59.3
Algoritmos Prof Dr P Kantek (pkantek@gmail.com)
Achando o menor caminho no metrô de Curitiba VIVXq86a V: 1.02
75691 MARIA EDUARDA DA SILVA
24JOG409 - 15 ______ / ______ / ______

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

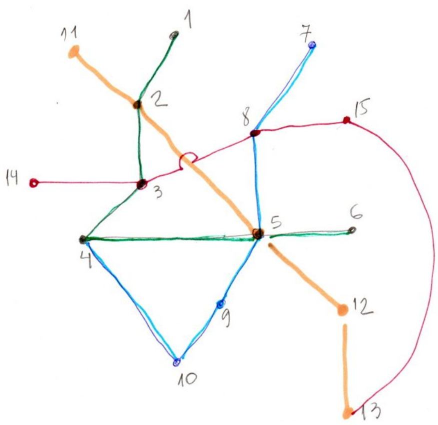
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô
Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

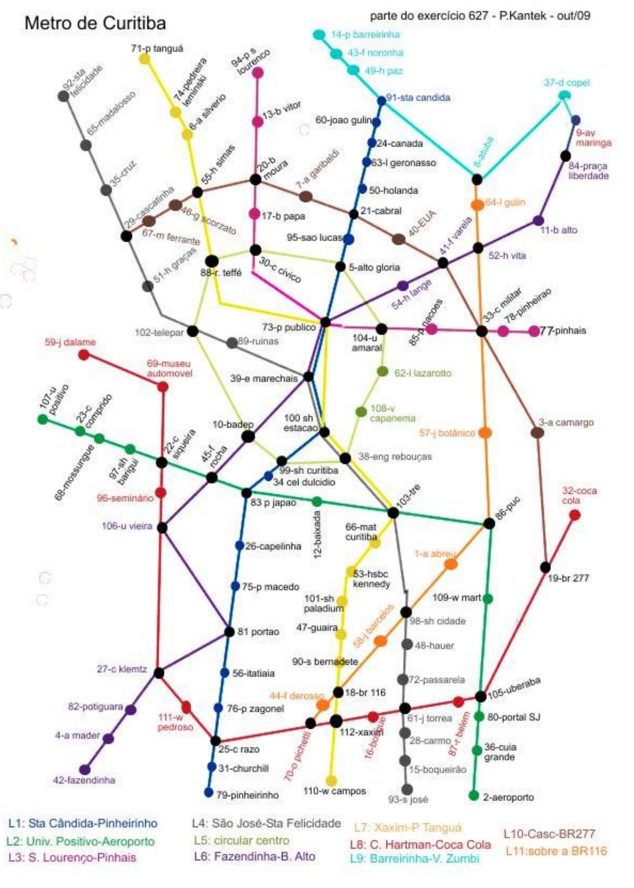
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 97-shopping barigui e 45-francisco rocha ?
2. Qual o número da sexta estação na rota mínima entre a estação 68-mossunguê e a estação 53-hsbc kennedy ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75691 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

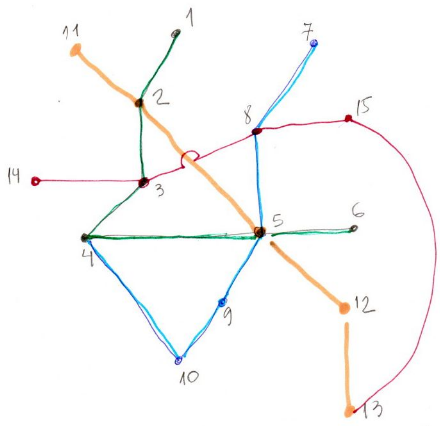
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

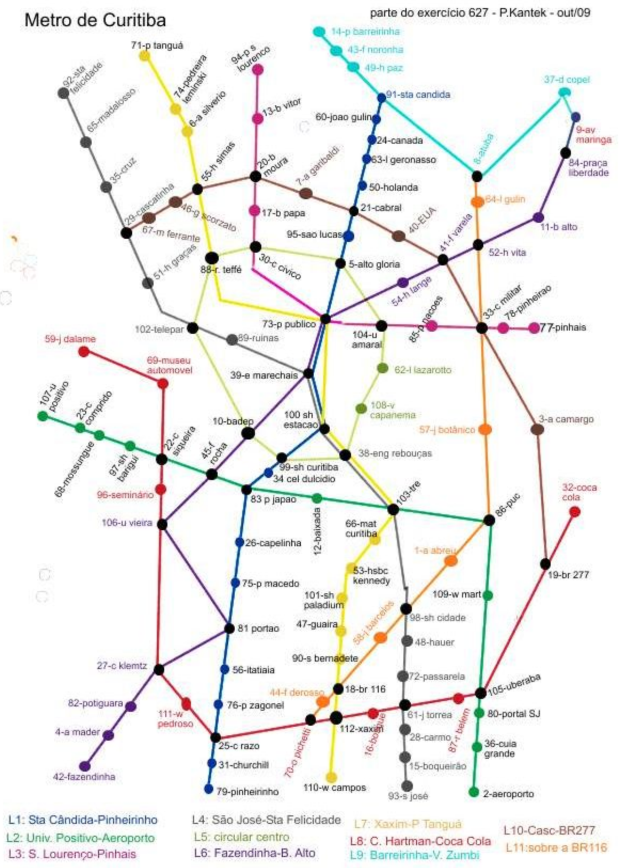
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 73-passeio público e 76-pedro zagonel ?
2. Qual o número da sexta estação na rota mínima entre a estação 60-joão gulin e a estação 98-shopping cidade ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75703 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

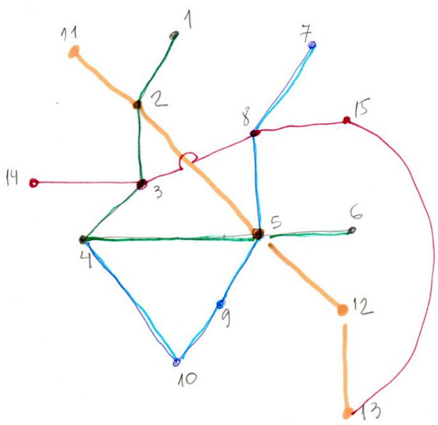
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

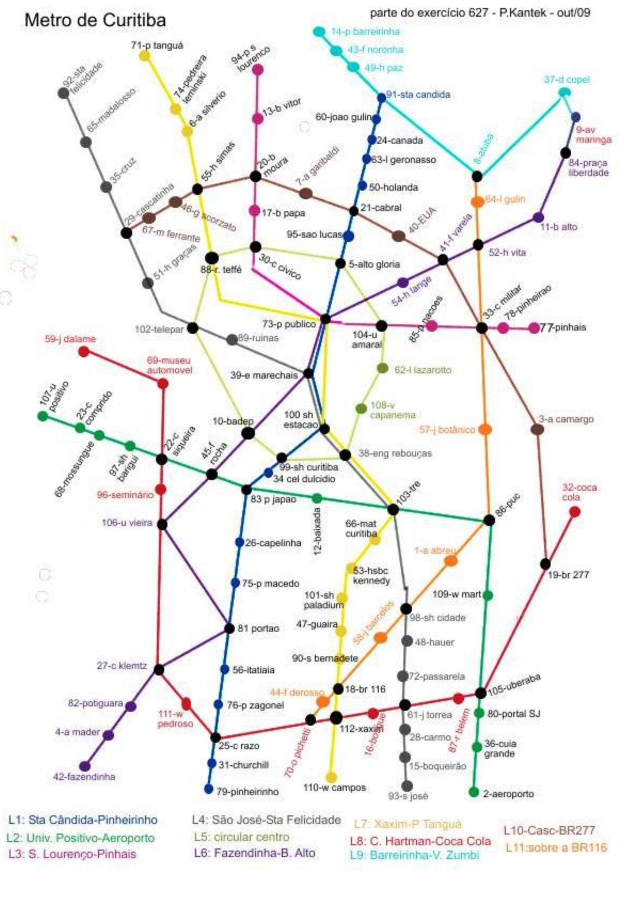
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 21-cabral e 10-badep ?
2. Qual o número da quinta estação na rota mínima entre a estação 57-jardim botânico e a estação 59-joão dallami ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75710 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

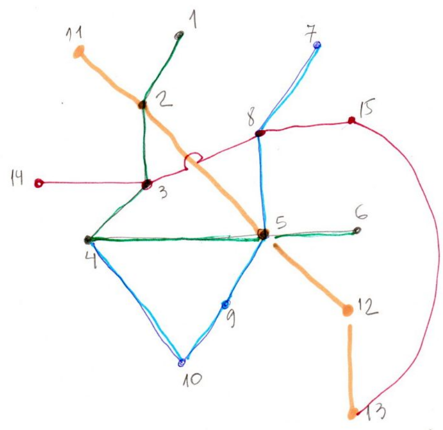
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

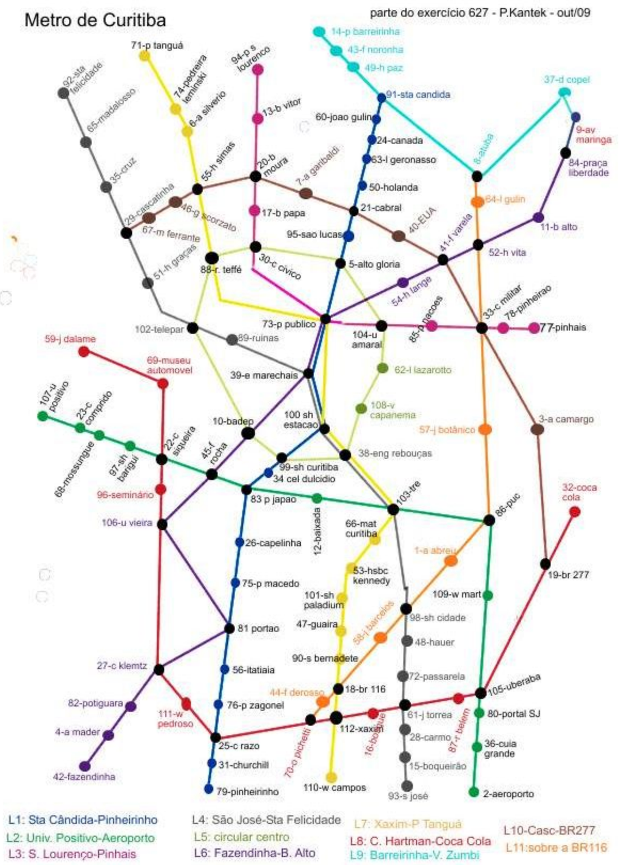
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 33-colégio militar e 27-carlos klemtz ?
2. Qual o número da quinta estação na rota mínima entre a estação 53-hsbc kennedy e a estação 37-depósito copel ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75727 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

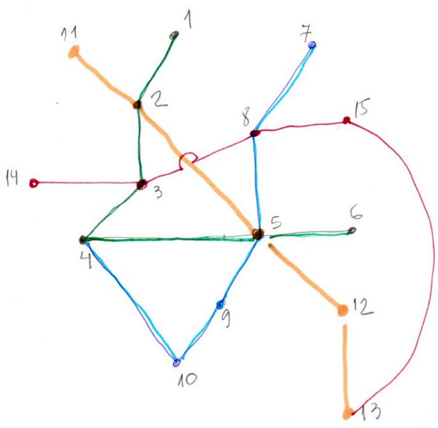
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

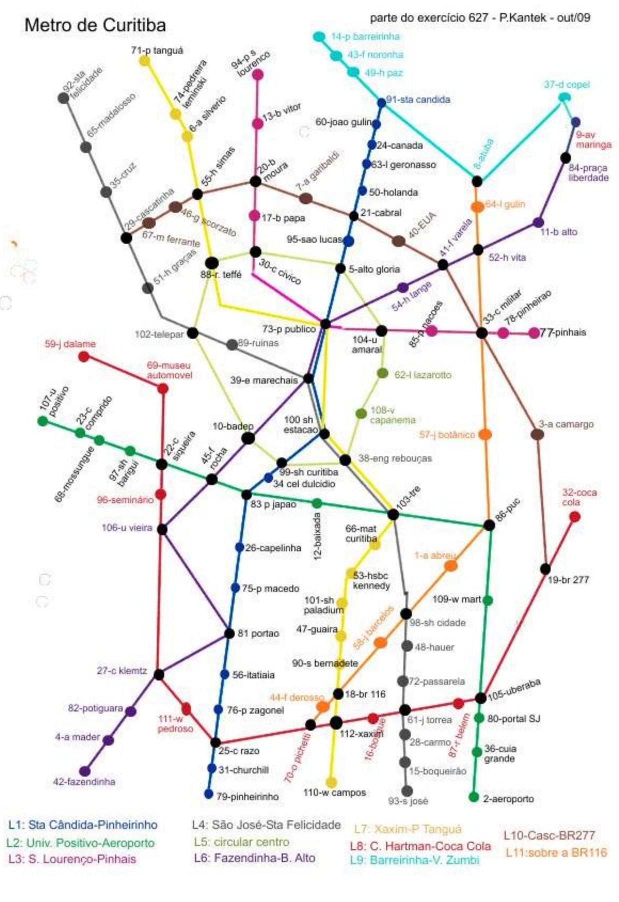
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 90-santa bernadete e 15-boqueirão ?

2. Qual o número da sexta estação na rota mínima entre a estação 101-shopping paladium e a estação 29-cascatinha ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75734 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

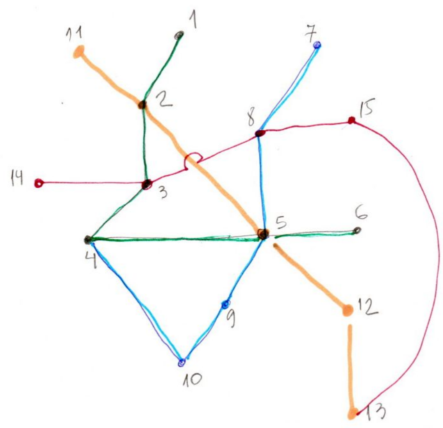
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

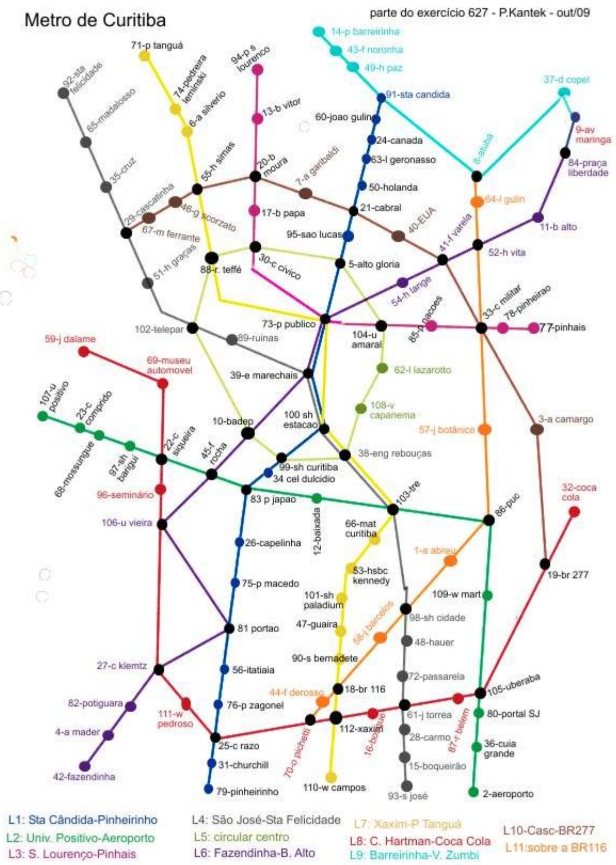
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer

Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 89-ruinas sãofrancisco e 61-joaquim torrea ?
2. Qual o número da quinta estação na rota mínima entre a estação 21-cabral e a estação 18-br116 ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75741 - /

Caminho mínimo

No exercício q81 já se estudou o algoritmo de Dijkstra (lê-se como Dikster) que permite calcular uma origem para um destino, é muito eficiente, e é a base dos inúmeros *wazes* e *garmin* que existem por aí.

Nesta folha o problema é muito parecido (achar o menor caminho), mas com algumas diferenças:

- Vai-se calcular TODAS as origens para TODOS os destinos.
- Por conta disso é um algoritmo altamente consumidor de recursos ($O(n^3)$).
- Portanto, só se justifica se os pontos forem fixos ao longo do tempo. O que não acontece no waze.
- O nome do algoritmo é Floyd-Warshall.

O algoritmo de Floyd-Warshall é um clássico da ciência da computação, utilizado para encontrar os caminhos mínimos entre todos os pares de vértices em um grafo ponderado. Sua origem é um pouco complexa, pois, como muitas descobertas científicas, envolve contribuições de diversos pesquisadores.

Múltiplas descobertas independentes:

- Bernard Roy (1959): Um dos primeiros a descrever o algoritmo, embora sua publicação tenha passado relativamente despercebida na época.
- Stephen Warshall (1962): Desenvolveu um algoritmo similar para determinar o fechamento transitivo de um grafo, que é essencialmente o mesmo problema abordado pelo algoritmo de Floyd-Warshall.
- Robert Floyd (1962): Publicou uma versão mais otimizada e conhecida do algoritmo, que acabou levando seu nome.

Apesar das contribuições anteriores, o nome "Floyd-Warshall" se popularizou devido à publicação de Robert Floyd, que apresentou uma versão mais eficiente e clara do algoritmo. Além disso, a publicação de Floyd ocorreu em uma revista de grande impacto, o que contribuiu para a disseminação do algoritmo na comunidade científica.

O que torna o algoritmo de Floyd-Warshall tão importante?

Versatilidade: Pode ser aplicado a uma ampla variedade de problemas, desde rotas em mapas até análise de redes sociais.

Eficiência: Possui uma complexidade de tempo de $O(n^3)$, onde n é o número de vértices do grafo, tornando-o eficiente para grafos de tamanho moderado.

Simplicidade: A ideia por trás do algoritmo é relativamente fácil de entender, o que facilita sua implementação e análise.

Roteamento: um subproduto muito bom do algoritmo é uma matriz de roteamento. Assim, sabe-se quanto vale o menor caminho e também por onde ele passa.

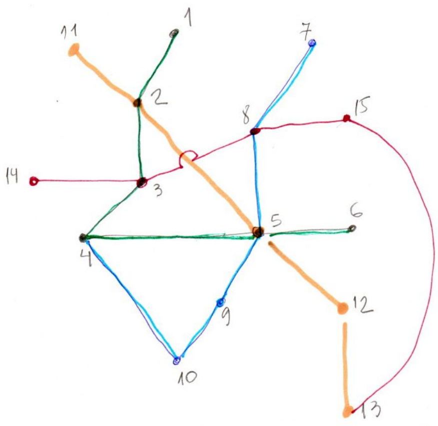
O algoritmo de Floyd-Warshall é um exemplo de como a ciência progride através de colaboração e descobertas independentes. Embora tenha múltiplos "pais", o nome de Robert Floyd ficou associado ao algoritmo devido à sua publicação e à sua contribuição para a otimização e popularização da técnica. (O texto acima teve a colaboração do Gemini).

Eis o algoritmo:

```
função floyd c
rota=mesma ordem de c, contendo zeros
cus=0 999 edit c  // troque 0 por 999 (infinito)
For i In range(len(c[0]))
  cus[i;i]=0  // zera a diagonal principal
End
For q In range(len(c[0]))
  For s In range(len(c[0]))
    For t In range(len(c[0]))
      If (cus[s;q]+cus[q;t])<cus[s;t]
        cus[s;t]=cus[s;q]+cus[q;t]
        If rota[s;q]==0
          rota[s;t]=q
        Else
          rota[s;t]=rota[s;q]
        EndIf
      EndIf
    EndFor
  EndFor
EndFor
for i in range(len(c[0]))
  for q in range(len(c[0]))
    if rota[i;q]==0
      rota[i;q]=q
    endif
  endfor
endfor
return cus,rota
```

Um exemplo

Suponha o seguinte grafo



que deu origem à seguinte matriz de custos

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2-	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0
3-	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
4-	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0
5-	0	1	0	1	0	1	0	1	1	0	0	1	0	0	0
6-	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7-	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8-	0	0	1	0	1	0	1	0	0	0	0	0	0	0	1
9-	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
10-	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
11-	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
12-	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
13-	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
14-	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15-	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

A execução do algoritmo de Floyd-Warshall sobre esta matriz deu origem ao seguinte resultado

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	0	1	2	3	2	3	4	3	3	4	2	3	4	3	4
2-	1	0	1	2	1	2	3	2	2	3	1	2	3	2	3
3-	2	1	0	1	2	3	2	1	3	2	2	3	3	1	2
4-	3	2	1	0	1	2	3	2	2	1	3	2	3	2	3
5-	2	1	2	1	0	1	2	1	1	2	2	1	2	3	2
6-	3	2	3	2	1	0	3	2	2	3	3	2	3	4	3
7-	4	3	2	3	2	3	0	1	3	4	4	3	3	3	2
8-	3	2	1	2	1	2	1	0	2	3	3	2	2	2	1
9-	3	2	3	2	1	2	3	2	0	1	3	2	3	4	3
10-	4	3	2	1	2	3	4	3	1	0	4	3	4	3	4
11-	2	1	2	3	2	3	4	3	3	4	0	3	4	3	4
12-	3	2	3	2	1	2	3	2	2	3	3	0	1	4	2
13-	4	3	3	3	2	3	3	2	3	4	4	1	0	4	1
14-	3	2	1	2	3	4	3	2	4	3	3	4	4	0	3
15-	4	3	2	3	2	3	2	1	3	4	4	2	1	3	0

Com a seguinte matriz de roteamento

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1-	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2
2-	1	2	3	3	5	5	3	3	5	3	11	5	5	3	3
3-	2	2	3	4	2	2	8	8	2	4	2	2	8	14	8
4-	3	3	3	4	5	5	3	3	5	10	3	5	5	3	3
5-	2	2	2	4	5	6	8	8	9	4	2	12	12	2	8
6-	5	5	5	5	5	6	5	5	5	5	5	5	5	5	5
7-	8	8	8	8	8	8	7	8	8	8	8	8	8	8	8
8-	3	3	3	3	5	5	7	8	5	3	3	5	15	3	15
9-	5	5	5	5	5	5	5	5	9	10	5	5	5	5	5
10-	4	4	4	4	4	4	4	4	9	10	4	4	4	4	4
11-	2	2	2	2	2	2	2	2	2	2	11	2	2	2	2
12-	5	5	5	5	5	5	5	5	5	5	5	12	13	5	13
13-	12	12	15	12	12	12	15	15	12	12	12	12	13	15	15
14-	3	3	3	3	3	3	3	3	3	3	3	3	3	14	3
15-	8	8	8	8	8	8	8	8	8	8	8	13	13	8	15

Interpretação

Suponha que você quer ir do nodo 11 para o nodo 7. Olhando a matriz de custos (linha 11, coluna 7) descobre-se que este caminho envolve 4 arestas ($cus[11][7] = 4$). Para descobrir quais são elas, começa-se na mesma posição (11,7) da matriz de rotas. O valor lá é ($rota[11][7] = 2$), o que significa que a rota é: $11 \rightarrow 2$. Daí, na mesma coluna, busca-se o conteúdo da linha 2, que é $rota[7][7] = 3$. Agora a rota é $11 \rightarrow 2 \rightarrow 3$. Agora ve-se $rota[3][7] = 8$. A nova rota: $11 \rightarrow 2 \rightarrow 3 \rightarrow 8$ Finalmente, olha-se em $rota[8][7] = 7$. Este resultado caracteriza o final da rota, veja-se $11 \rightarrow 7 = 11 \rightarrow 2 \rightarrow 3 \rightarrow 8 \rightarrow 7$, que tem 4 arestas, como já se sabia.

O metrô de Curitiba

Considere as seguintes linhas de metrô

Linha 1: Santa Cândida-Pinheirinho (azulão) santa cândida; joão gulin; canadá; ludovico geronasso; holanda; cabral; são lucas; alto glória; passeio público; esquina marechais; shopping estação; shopping curitiba; coronel dulcídio; praça japão; capelinha; pedro macedo; portão; itatiaia; pedro zagonel; capão raso; churchill; pinheirinho. Total de estações: 22

Linha 2: UFPR-UP-UTFPR-PUC/Pr -Aeroporto (verdão) universidade positivo; campo comprido; mossunguê; shopping barigui; campina siqueira; francisco rocha; praça japão; baixada; tre; puc; wal mart; uberaba; portal sãojosé; cuia grande; aeroporto. Total de estações: 15

Linha 3: São Lourenço-Pinhais (rosa) são lourenço; bar vitor; brasilino moura; bosque papa; centro cívico; passeio público; ubaldino amaral; praça nações; colégio militar; pinheirão; pinhais. Total de estações: 11

Linha 4: São José-Santa Felicidade (cinza) são josé; boqueirão; carmo; joaquim torrea; passarela; hauer; shopping cidade; tre; engenheiros rebouças; shopping estação; esquina marechais; ruinas sãofrancisco; telepar; hospital graças; cascatinha; cruz; madalosso; santa felicidade. Total de estações: 18

Linha 5: circular centro (verde claro) rua teffe; telepar; badep; shopping curitiba; engenheiros reboucas; vila capanema; largo lazzarotto; ubaldino amaral; alto gloria; centro civico e retorna novamente à estação da rua teffé. Há aqui 10 estações.

Linha 6: Fazendinha-Bairro Alto (roxo) fazendinha; algacyr mader; potiguara; carlos klemtz; portão; ulisses vieira; francisco rocha; badep; esquina marechais; passeio público; hugo lange; fagundes varela; hospital vita; bairro alto; praça liberdade; avenida maringá. Total de estações: 16

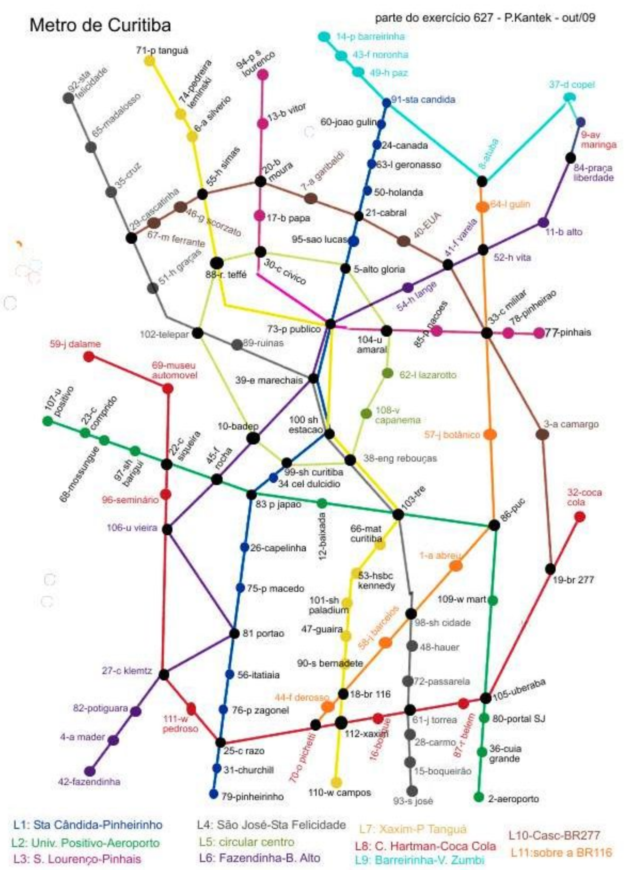
Linha 7: Xaxim-Parque Tanguá (amarelo) waldemar campos; xaxim; br116; santa bernadete; guaira; shopping paladium; hsbc kennedy; maternidade curitiba; tre; engenheiros rebouças; shopping estação; passeio público; rua teffé; hugo simas; amaury silvério; pedreira leminski; parque tanguá. Total de estações: 17

Linha 8: Candido Hartmann-Coca Cola (vermelho) joão dallami; museu automóvel; campina siqueira; seminário; ulisses vieira; carlos klemtz; waldemiro pedroso; capão raso; omar pichetti; xaxim; bosque maack; joaquim torrea; rio belém; uberaba; br277; coca cola. Total de estações: 16

Linha 9: Barreirinha-Vila Zumbi (azul claro) barreirinha; fernando noronha; hildo paz; santa cândida; atuba; depósito copel; avenida maringá. Total de estações: 7

Linha 10: Cascatinha-BR 277 (marrom) cascatinha; mbá ferrante; gardênio scorzato; hugo simas; brasilino moura; anita garibaldi; cabral; estados unidos; fagundes varela; colégio militar; afonso camargo; br277. Total de estações: 12

Linha 11: Sobre a BR116 (laranja) omar pichetti; francisco derosso; br116; joão barcelos; shopping cidade; abílio abreu; puc; jardim botânico; colégio militar; hospital vita; luiza gulin; atuba. Total de estações: 12



Como Fazer Note que somando as estações de cada linha, obtem-se 156 , quando na realidade existem apenas 112 estações. A explicação é que muitas estações são compartilhadas entre várias linhas, o que caracteriza a formação de uma rede de transporte, onde o viajante pode trocar de linhas dentro da estação sem ter que pagar nova passagem. A estação que mais tem linhas é a do passeio público, onde passam as linhas 1, 3 6 e 7.

1. Inicialmente, tente obter ou fazer um desenho da distribuição das linhas no espaço urbano. Este desenho serve de aquecimento e um bom lugar "por onde começar".
2. Há que se ter uma lista de estações (sem duplicidades). Esta lista pode ser obtida das paradas de cada linha. Ela precisa ter alguma ordenação única, sugere-se a alfabética a fim de coincidir com as estações pedidas nesta folha. Este número de estações tem que coincidir com aquele dado no problema.
3. Agora a parte mais difícil: Construir a matriz de adjacência do problema (quadrada de ordem igual ao número de estações). Colocar **1** nas ligações entre estações. (Por exemplo, se a linha 1 conecta a praça do japão com a capelinha, e se a praça do japão é a estação 83 enquanto a capelinha é a estação 27, e se a matriz de adjacência por hipótese é MADJ, deve-se fazer MADJ[83;27] $\leftarrow$ 1 e também MADJ[27;83] $\leftarrow$ 1).
4. Tendo construído (e conferido) esta matriz é hora de entregá-la ao algoritmo de Floyd Warshall, lembrando de construir também a matriz de roteamentos.
5. Com estas duas matrizes é possível responder e acertar as 3 perguntas aqui colocadas.

Para você usar

No local de sempre, há diversos arquivos que você deverá obter para poder resolver o exercício:

linhas Um arquivo de texto contendo as 11 linhas e os nomes das estações que compõe a linha

estacoes Um arquivo com todas as estações (em ordem alfabética) indicando qual(is) a(s) linha(s) que passa(m) nesta estação, e depois os números das estações que são vizinhas a esta estação.

matlig Matriz de 112 linhas por 112 colunas (tamanho=3) contendo 1 ou 0 quando há ou não há ligação entre as estações.

matcus Matriz de 112 linhas por 112 colunas (tamanho=3) contendo a comprimento do caminho mínimo entre as duas estações (em arestas)

matrot Matriz de 112 linhas por 112 colunas (tamanho=4) contendo a rota mínima entre as estações, conforme descrito no exemplo acima.

☞ Para você fazer

1. Quantos trechos de metrô compõe o menor caminho entre as estações 48-hauer e 74-pedreira leminski ?

2. Qual o número da quinta estação na rota mínima entre a estação 68-mossunguê e a estação 112-xaxim ? (Obs: a. A origem é a estação número 1; b. Havendo mais de um caminho mínimo, você deve simular o algoritmo e responder o que ele responderá.)

1	2

409-75758 - /