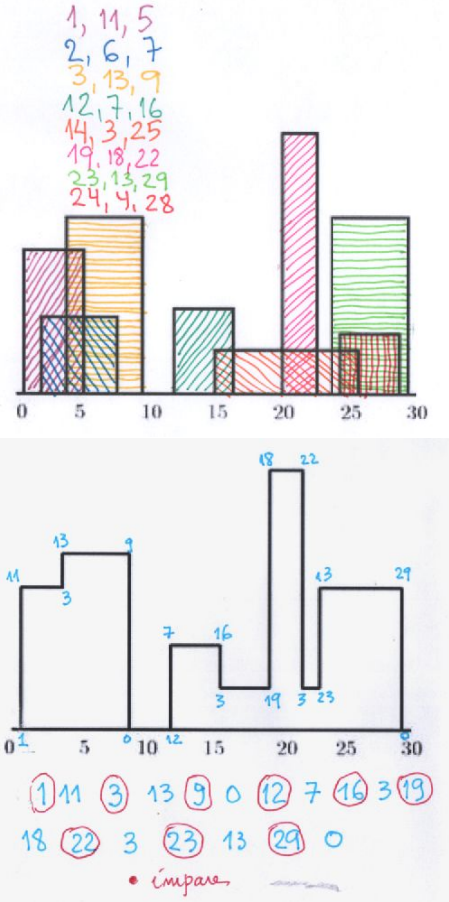


UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas ($1, 11, 5$), ($2, 6, 7$), ($3, 13, 9$), ($12, 7, 16$), ($14, 3, 25$), ($19, 18, 22$), ($23, 13, 29$), ($24, 4, 28$) o horizonte, mostrado à direita, é representado pela sequência: ($1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0$) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
        variando i de pcom até pfin
            altura[i] = max(altura[i],altu)
        fim{variando}
    fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 1 6
16 19 25
19 1 22
19 4 26
32 17 34
35 28 42
35 29 39
42 5 44
42 25 43
46 27 51

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],[23,28,26],[29,19,30],[37,16,42],[40,22,47]]

deve dar:

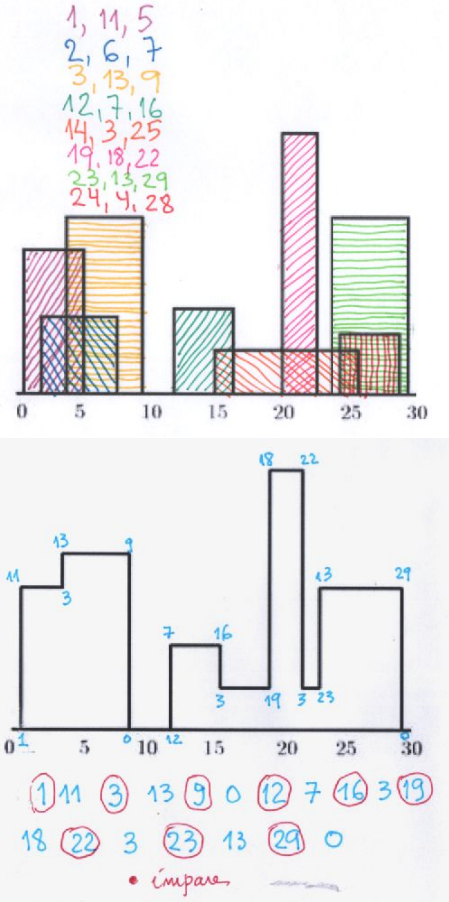
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76001 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuradas por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharrem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 9 7
15 11 19
23 17 28
27 7 35
28 25 33
33 4 39
35 1 43
38 6 44
46 21 50
47 30 51

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

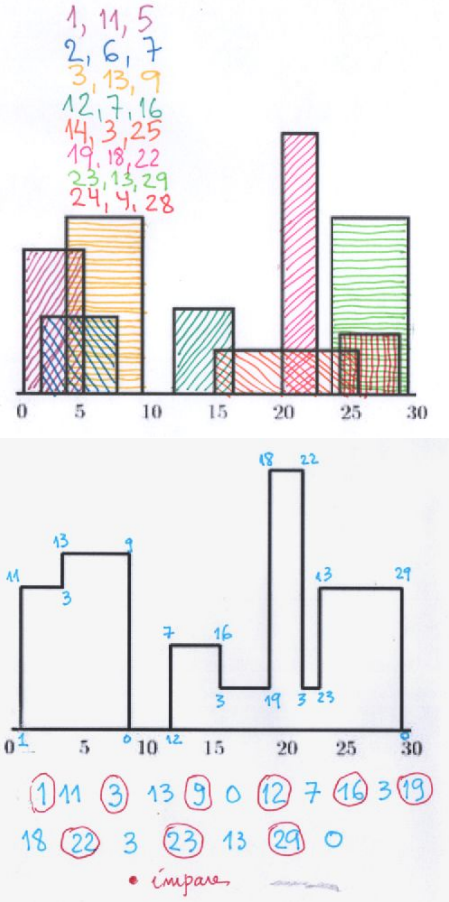
a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76199 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 28 2
16 28 19
18 1 28
25 18 34
28 1 37
29 5 31
31 6 40
37 5 44
39 6 46
48 11 58
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios			
prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

```
a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
```

deve dar:

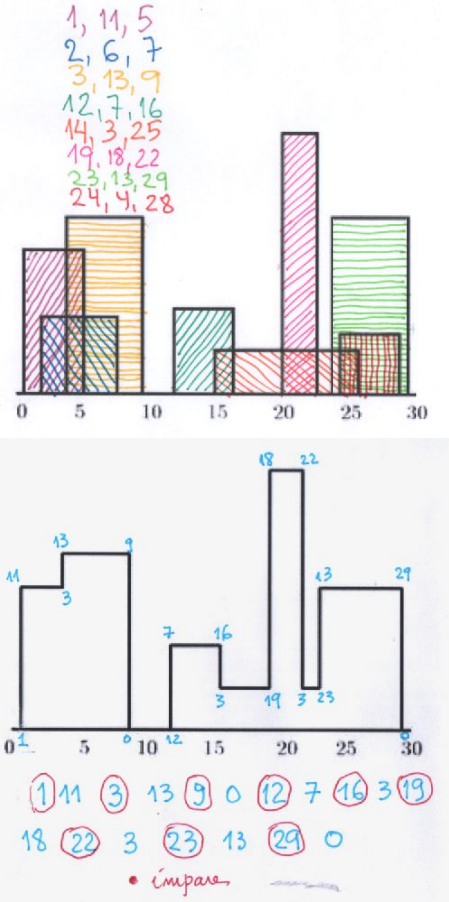
```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76018 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuradas por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
        variando i de pcom até pfin
            altura[i] = max(altura[i],altu)
        fim{variando}
    fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharrem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 22 11
1 18 5
4 15 13
11 26 15
16 4 25
18 30 28
28 17 36
37 17 45
44 15 47
46 18 47
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

```
a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
```

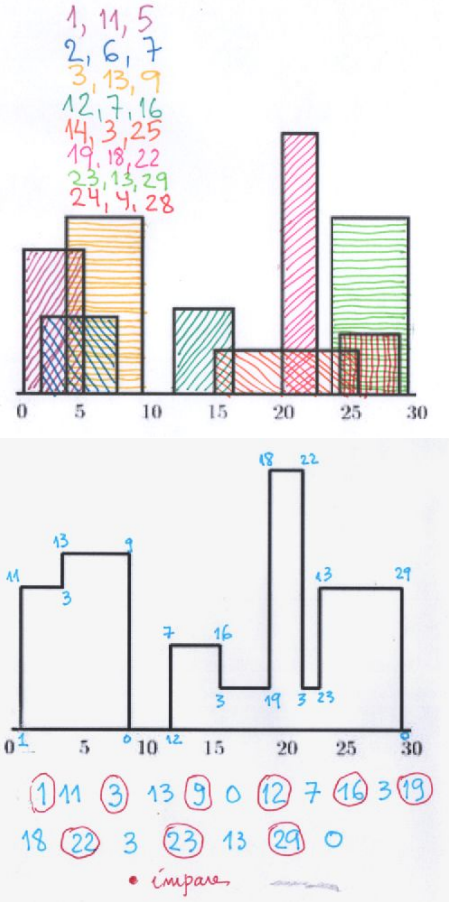
```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76025 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 29 11
7 26 16
7 22 9
27 4 33
29 7 33
30 18 37
31 10 40
33 17 37
39 6 46
44 13 51
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios			
prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

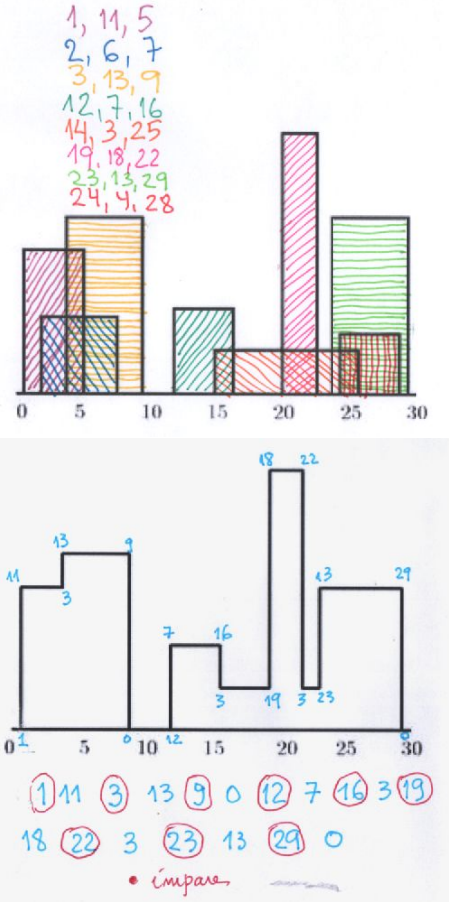
```
a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76032 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 29 3
6 9 11
11 9 19
13 15 14
19 21 29
33 19 40
34 10 42
35 25 40
37 5 40
42 28 46

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

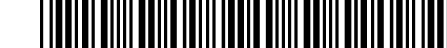
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],[23,28,26],[29,19,30],[37,16,42],[40,22,47]]

deve dar:

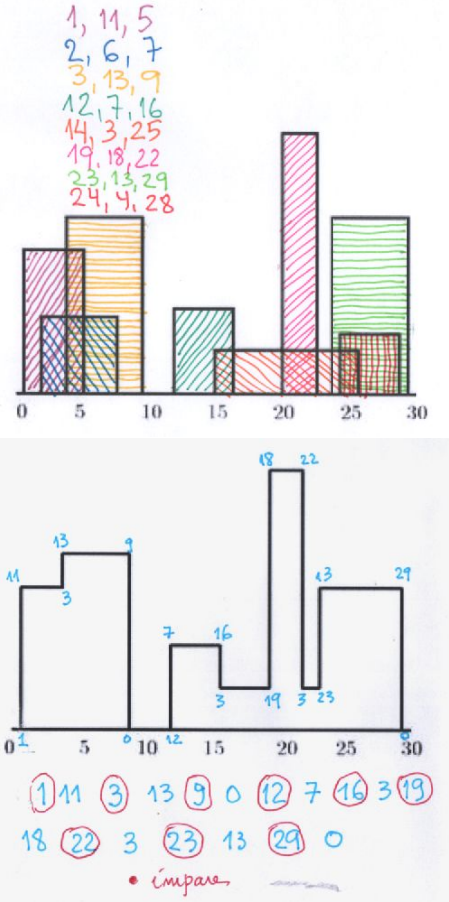
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76049 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

se infim==0 (início)
 insere a altura na lista segmentos
 ordena segmentos em ordem decrescente
senão
 remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
 (ou 0 se vazia)
se nova-altura <> corrente
 insere altura na vista
 corrente = nova-altura
retornar vista

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
    variando i de pcom até pfin
        altura[i] = max(altura[i],altu)
    fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharrem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 23 7
13 26 15
15 24 21
16 13 23
19 13 23
22 26 27
28 28 38
29 1 34
33 28 36
45 3 48

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

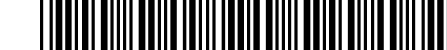
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]

deve dar:

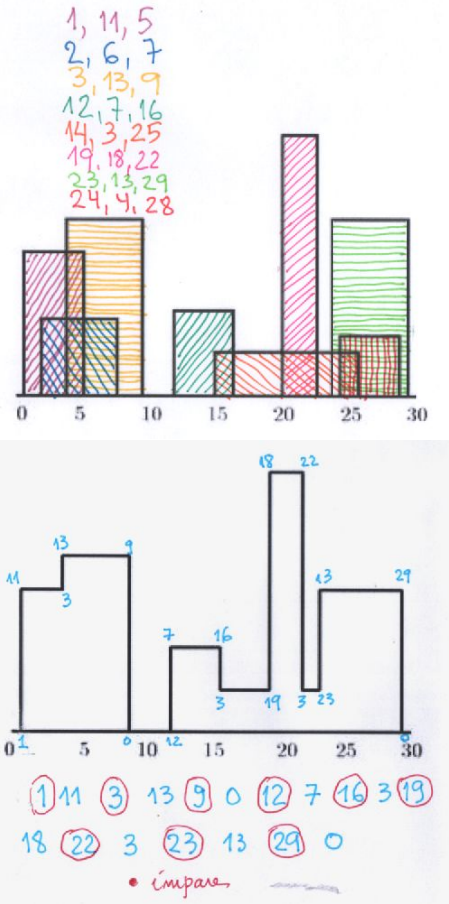
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76056 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

se infim==0 (início)
 insere a altura na lista segmentos
 ordena segmentos em ordem decrescente
senão
 remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
 (ou 0 se vazia)
se nova-altura <> corrente
 insere altura na vista
 corrente = nova-altura
retornar vista

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
 se pfin>final
 final=pfin (localizando o final do desenho)
 fim{se}
 variando i de pcom até pfin
 altura[i] = max(altura[i],altu)
 fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
 se cara <> altura[i]
 imprima (i, altura[i])
 cara = altura[i]
 fim{se}
fim{variando}
imprima (final, 0)

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:
1 12 7
4 17 10
19 1 26
21 24 31
26 20 28
31 17 41
39 1 41
39 18 46
41 11 46
47 3 48

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite
calcular

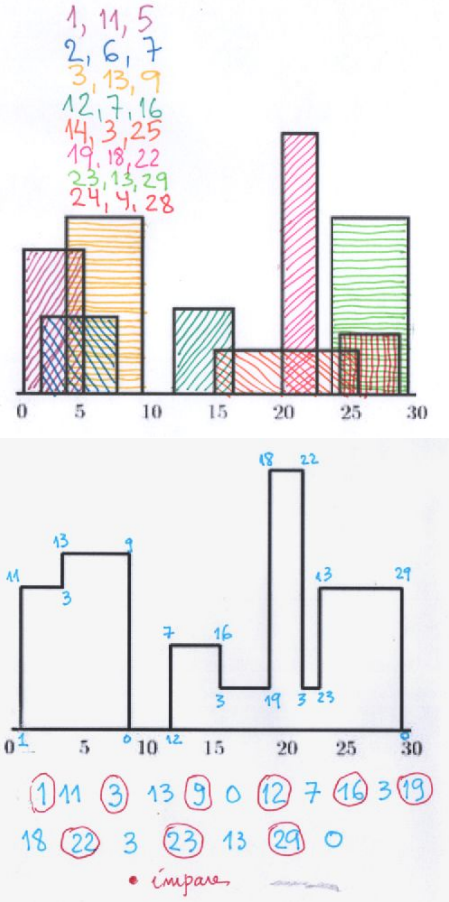
A skyline é
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
        variando i de pcom até pfin
            altura[i] = max(altura[i],altu)
        fim{variando}
    fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharrem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 4 5
11 30 19
13 25 17
13 18 19
14 3 18
18 27 21
21 27 29
29 18 34
29 23 33
42 26 44
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

```
a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
```

deve dar:

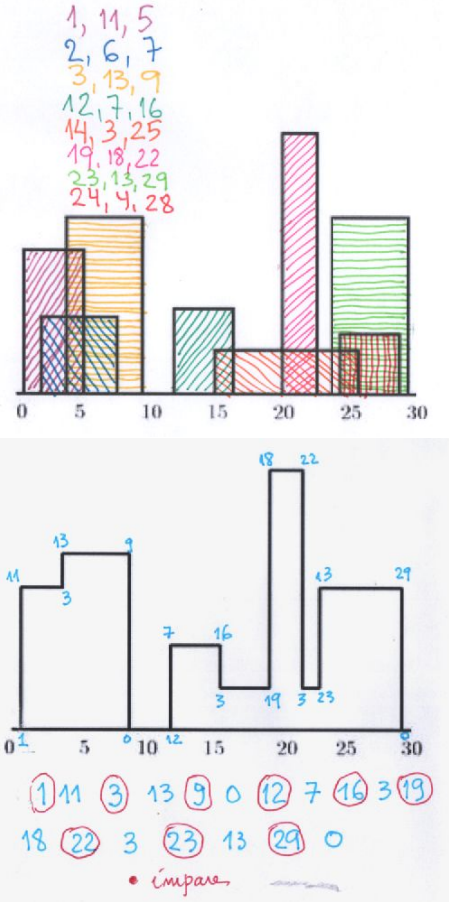
```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76063 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
    variando i de pcom até pfin
        altura[i] = max(altura[i],altu)
    fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 1 11
8 25 11
8 12 12
19 12 21
21 1 26
27 12 33
34 29 42
38 28 42
40 19 44
49 10 51
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

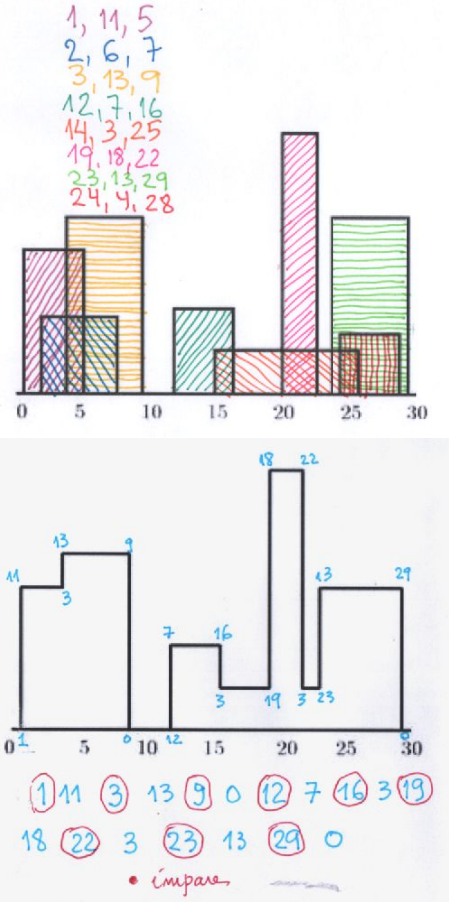
```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

```
a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
    variando i de pcom até pfin
        altura[i] = max(altura[i],altu)
    fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 6 4
9 5 13
14 17 21
20 3 21
21 3 22
22 6 24
29 9 32
30 4 35
32 22 41
45 14 52
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

```
a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
```

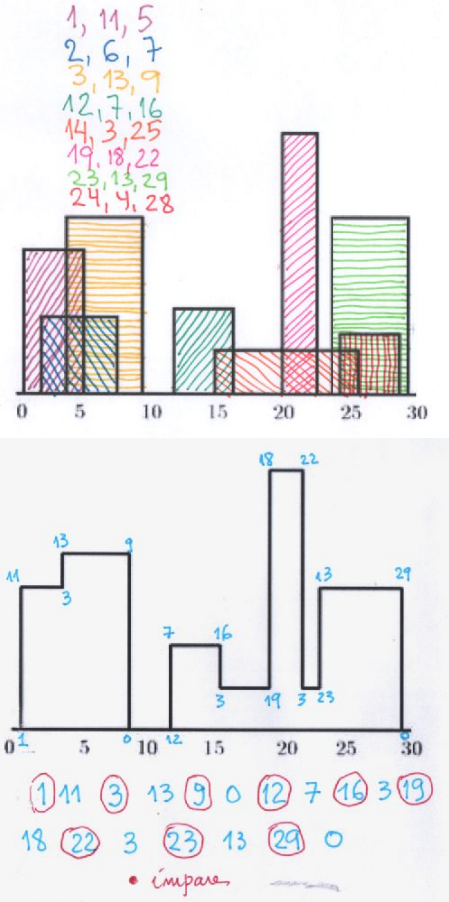
```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76087 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 25 6
3 3 9
5 17 7
9 29 16
14 5 19
37 20 41
41 11 47
44 22 54
46 23 54
48 24 53

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios			
prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

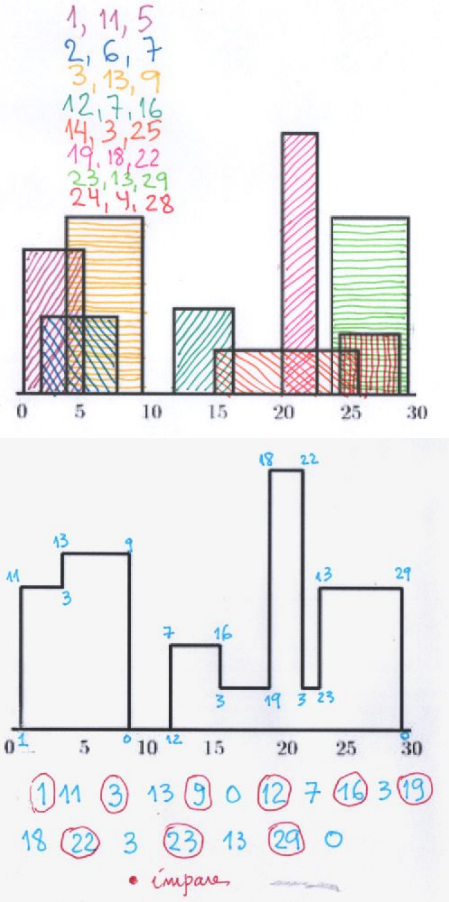
a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76687 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
    variando i de pcom até pfin
        altura[i] = max(altura[i],altu)
    fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharrem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 28 7
4 5 13
8 2 15
9 18 19
18 18 21
26 5 33
30 7 35
37 26 42
37 3 39
44 22 54
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

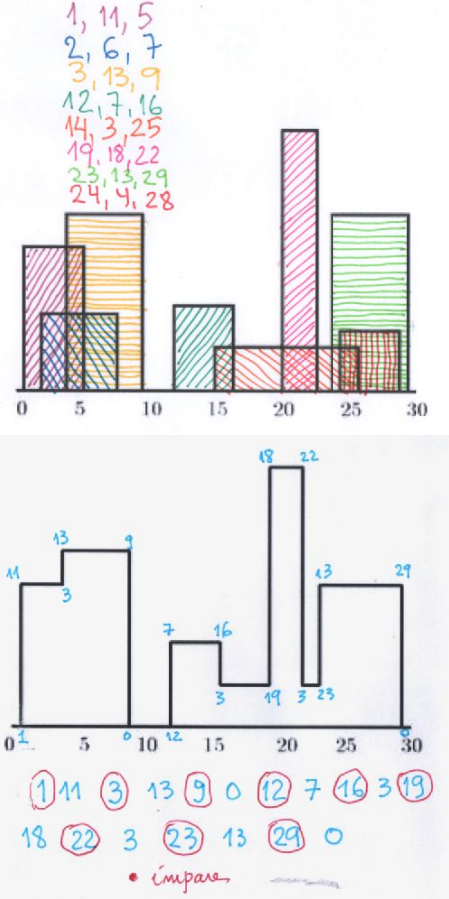
```
a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76106 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda das triplas ($1, 11, 5$), ($2, 6, 7$), ($3, 13, 9$), ($12, 7, 16$), ($14, 3, 25$), ($19, 18, 22$), ($23, 13, 29$), ($24, 4, 28$) o horizonte, mostrado à direita, é representado pela sequência: ($1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0$) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
        variando i de pcom até pfin
            altura[i] = max(altura[i],altu)
        fim{variando}
    fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao [onlinejudge](http://onlinejudge.org) da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 15 4
3 15 4
26 4 29
26 28 34
26 14 36
34 27 38
41 13 50
43 12 44
44 28 48
44 1 45
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

```
a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
```

deve dar:

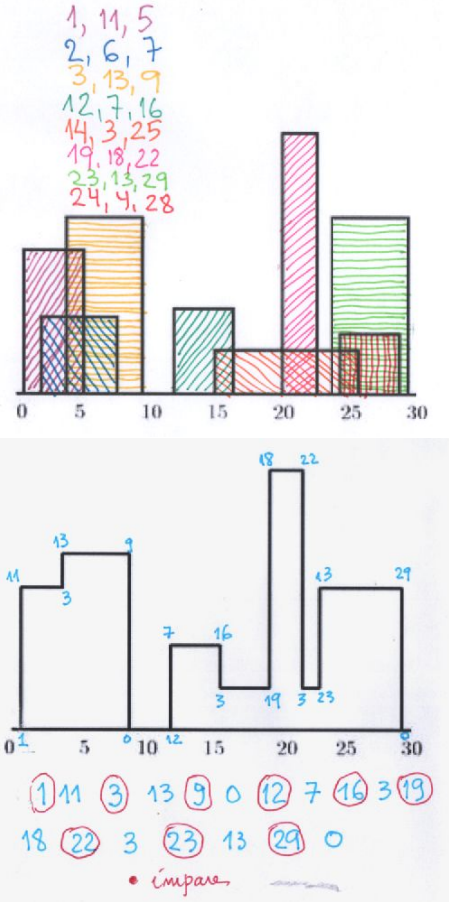
```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76113 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda das triplas ($1, 11, 5$), ($2, 6, 7$), ($3, 13, 9$), ($12, 7, 16$), ($14, 3, 25$), ($19, 18, 22$), ($23, 13, 29$), ($24, 4, 28$) o horizonte, mostrado à direita, é representado pela sequência: ($1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0$) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

```
Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0
```

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
    insere a altura na lista segmentos
    ordena segmentos em ordem decrescente
senão
    remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
    insere altura na vista
    corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
    se pfin>final
        final=pfin (localizando o final do desenho)
    fim{se}
        variando i de pcom até pfin
            altura[i] = max(altura[i],altu)
        fim{variando}
    fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
    se cara <> altura[i]
        imprima (i, altura[i])
        cara = altura[i]
    fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

```
1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47
```

Deu como resultado

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0
```

Caso 2

```
1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55
```

Deu como resultado

```
1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0
```

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

```
1 8 8
3 30 13
12 29 21
17 9 18
20 22 29
26 9 28
39 17 45
45 21 49
45 10 51
48 18 52
```

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios			
prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```

Para testar

```
a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
```

deve dar:

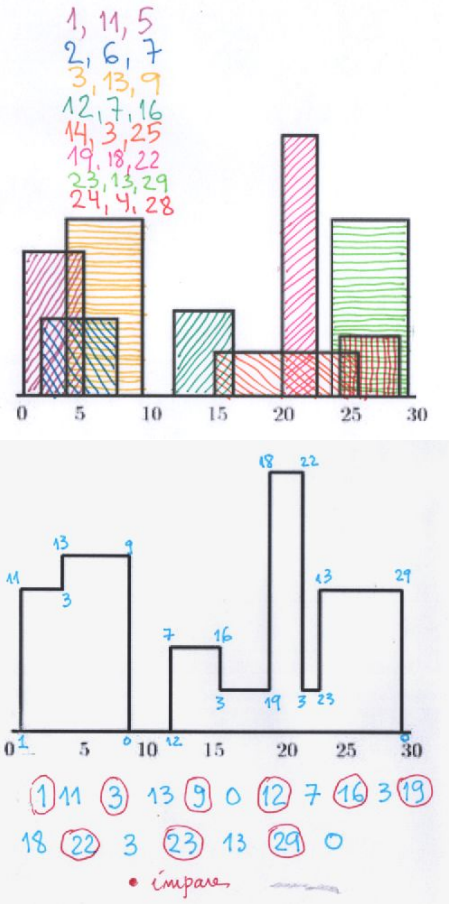
```
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0
```



509-76120 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas ($1, 11, 5$), ($2, 6, 7$), ($3, 13, 9$), ($12, 7, 16$), ($14, 3, 25$), ($19, 18, 22$), ($23, 13, 29$), ($24, 4, 28$) o horizonte, mostrado à direita, é representado pela sequência: ($1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0$) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao [onlinejudge](http://onlinejudge.org) da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 27 6
8 15 15
9 1 10
17 12 26
22 30 25
33 6 42
36 3 38
40 17 50
47 11 53
47 29 48

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]

deve dar:

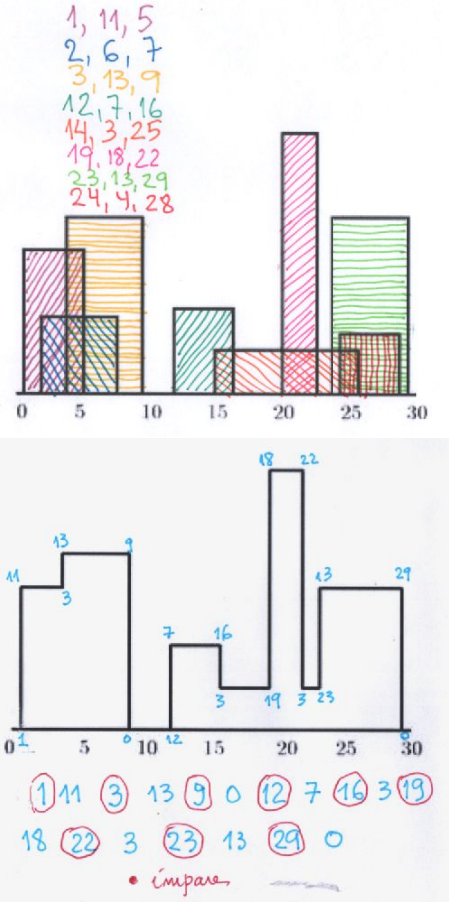
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76137 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construções de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

se infim==0 (início)
 insere a altura na lista segmentos
 ordena segmentos em ordem decrescente
senão
 remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
 (ou 0 se vazia)
se nova-altura <> corrente
 insere altura na vista
 corrente = nova-altura
retornar vista

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
 se pfin>final
 final=pfin (localizando o final do desenho)
 fim{se}
 variando i de pcom até pfin
 altura[i] = max(altura[i],altu)
 fim{variando}
 fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
 se cara <> altura[i]
 imprima (i, altura[i])
 cara = altura[i]
 fim{se}
fim{variando}
imprima (final, 0)

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharrem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 15 8
13 8 14
18 9 22
23 13 24
26 14 34
32 11 35
38 28 39
41 11 45
49 11 57
50 11 59

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],[23,28,26],[29,19,30],[37,16,42],[40,22,47]]

deve dar:

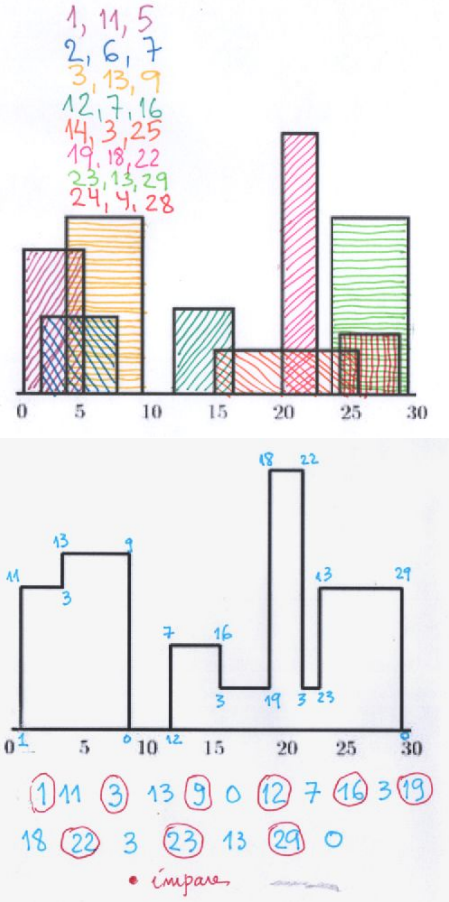
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76144 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuradas por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas (1, 11, 5), (2, 6, 7), (3, 13, 9), (12, 7, 16), (14, 3, 25), (19, 18, 22), (23, 13, 29), (24, 4, 28) o horizonte, mostrado à direita, é representado pela sequência: (1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 3 3
13 4 14
15 14 21
17 14 26
18 25 19
31 8 35
33 11 34
44 26 48
46 2 50
50 15 60

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios			
prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite
calcular

A skyline é
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

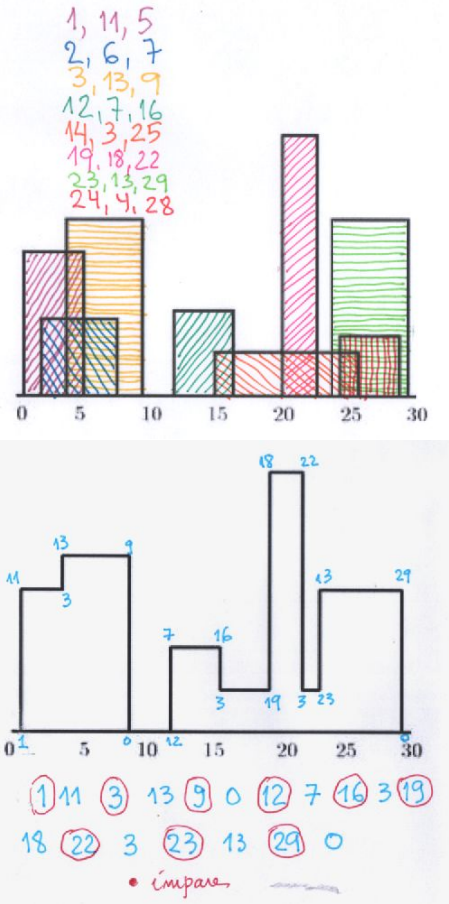
Para testar

a=[[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],[23,28,26],[29,19,30],[37,16,42],[40,22,47]]
deve dar:
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas ($1, 11, 5$), ($2, 6, 7$), ($3, 13, 9$), ($12, 7, 16$), ($14, 3, 25$), ($19, 18, 22$), ($23, 13, 29$), ($24, 4, 28$) o horizonte, mostrado à direita, é representado pela sequência: ($1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0$) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao [onlinejudge](http://onlinejudge.org) da UVA. Se você obtiver o ACCEPTED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharrem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 9 4
14 25 15
19 12 25
22 10 24
35 30 43
37 11 40
46 24 53
47 21 52
48 2 50
50 11 59

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]

deve dar:

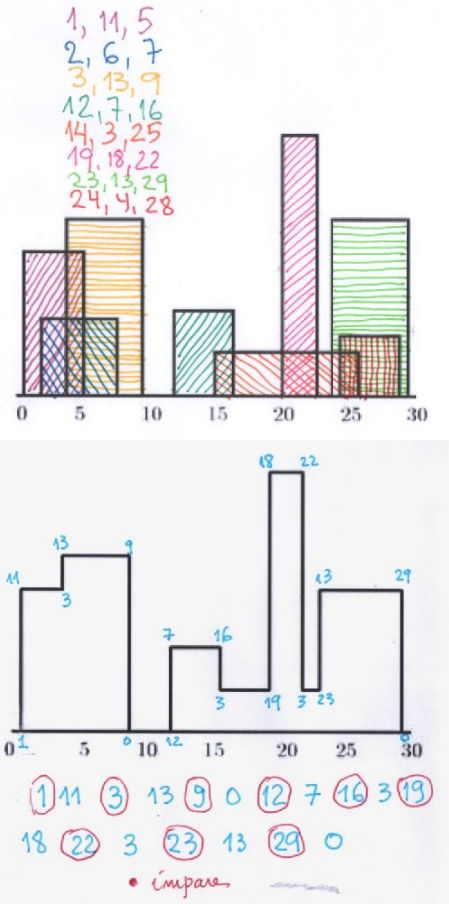
1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76175 - gar a

UVA105-Perfil da skyline

Tal como esta lá no site (<http://onlinejudge.org>): Com o advento das estações de trabalho gráficas de alta velocidade, CAD (design auxiliado por computador) e outras áreas (design CAM, VLSI) têm feito uso cada vez mais eficaz dos computadores. Um dos problemas com desenhar imagens é a eliminação de linhas ocultas (linhas obscuras por outras partes de um desenho). Você deverá projetar um programa para ajudar um arquiteto a desenhar o horizonte de uma cidade, considerando os locais dos edifícios da cidade. Para tornar o problema tratável, todos os edifícios são de forma retangular e eles compartilham um fundo comum (a cidade onde estão construídos é muito plana). A cidade também é vista como bidimensional. Um edifício é especificado por uma tripla ordenada (L_i, H_i, R_i) onde L_i e R_i são as coordenadas à esquerda e à direita, respectivamente, do edifício i , ($0 < L_i < R_i$) e H_i é a altura do edifício. No diagrama abaixo, os edifícios são mostrados à esquerda com triplas ($1, 11, 5$), ($2, 6, 7$), ($3, 13, 9$), ($12, 7, 16$), ($14, 3, 25$), ($19, 18, 22$), ($23, 13, 29$), ($24, 4, 28$) o horizonte, mostrado à direita, é representado pela sequência: ($1, 11, 3, 13, 9, 0, 12, 7, 16, 3, 19, 18, 22, 3, 23, 13, 29, 0$) Estes dados correspondem à seguinte imagem:



Entrada A entrada é uma sequência de construção de triplas. Todas as coordenadas dos edifícios são números inteiros menores que 10.000 e haverá pelo menos um e no máximo 5.000 edifícios no arquivo de entrada. Cada tripla (edifício) é uma linha sozinha no arquivo de entrada. Todos os inteiros em uma tripla são separados por um ou mais espaços. As triplas serão classificadas por L_i , a coordenada x esquerda do edifício, então o edifício com o menor valor à esquerda (a coordenada x) é o primeiro no arquivo de entrada.

Saída A saída deve consistir no vetor que descreve o horizonte conforme mos-

trado no exemplo acima. No vetor skyline ($v_1, v_2, v_3, \dots, v_{n-2}, v_{n-1}, v_n$), o v_i é tal que se i é um número par representa um linha horizontal (altura). Se i é um número ímpar representa uma linha vertical (coordenada x). O vetor skyline deve representar o 'caminho' percorrido, por exemplo, por uma formiga começando na menor coordenada x e viajando horizontal e verticalmente sobre todas as linhas que definem o horizonte. Por isso a última entrada em todos os vetores do horizonte será '0'.

Exemplo O desenho acima está assim representado:

Entrada
1 11 5
2 6 7
3 13 9
12 7 16
14 3 25
19 18 22
23 13 29
24 4 28
Saída
1 11 3 13 9 0 12 7 16 3 19 18 22 3 23 13 29 0

Algoritmo 1 Como quase todo problema não trivial, este admite diversas estratégias de solução. Uma possível é desdobrar a cada elemento da lista de entrada em dois eventos: O primeiro determina o início de um prédio e o segundo o seu final. Os eventos terão a estrutura: (esquerda, altura, 0=início) e (direita, altura, 1=final). Para o conjunto de dados do exemplo, ficará: [(1, 11, 1), (5, 11, 0), (2, 6, 1), (7, 6, 0), (3, 13, 1), (9, 13, 0), (12, 7, 1), (16, 7, 0), (14, 3, 1), (25, 3, 0), (19, 18, 1), (22, 18, 0), (23, 13, 1), (29, 13, 0), (24, 4, 1), (28, 4, 0)] A seguir ordena-se esta lista pela primeira posição (o x) cria-se uma lista chamada vista vazia ([]) e outra chamada segmentos vazia ([]) e corrente valendo 0 e depois processa-se a lista:

```
se infim==0 (início)
  insere a altura na lista segmentos
  ordena segmentos em ordem decrescente
senão
  remove esta altura de segmentos
nova-altura = primeiro elemento de segmentos
                (ou 0 se vazia)
se nova-altura <> corrente
  insere altura na vista
  corrente = nova-altura
retornar vista
```

Este algoritmo implementado reproduziu os dados do exemplo corretamente, mas foi recusado pelo online judge. Certamente algum conjunto de dados (oculto) ressaltou algum bug ou imprecisão.

Algoritmo 2 Uma saída foi buscar outro algoritmo mais simples (embora menos eficiente) e baseado numa informação perdida do enunciado (os valores são menores do que 10000). Agora define-se um vetor de 10001 elementos que representa as ordenadas x de todos os envolvidos no problema. Agora a estratégia ficou mais fácil: basta examinar o que acontece em cada ponto do vetor (já que todas as medidas são inteiras).

```
final=0
altura = [0] * 10001
processando (pcom, altu, pfin)
  se pfin>final
    final=pfin (localizando o final do desenho)
  fim{se}
  variando i de pcom até pfin
    altura[i] = max(altura[i],altu)
  fim{variando}
fim{processando}
cara = altura[1]
imprima (1, cara)
variando i de 2 até final
  se cara <> altura[i]
    imprima (i, altura[i])
    cara = altura[i]
  fim{se}
fim{variando}
imprima (final, 0)
```

Exemplos resolvidos Caso 1

1 18 11
4 25 7
7 6 15
17 26 25
19 25 25
21 12 26
23 28 26
29 19 30
37 16 42
40 22 47

Deu como resultado

1 18 4 25 7 18 11 6 15 0 17 26 23 28
26 0 29 19 30 0 37 16 40 22 47 0

Caso 2

1 3 9
12 17 21
24 4 28
24 1 29
31 9 38
39 10 45
41 7 45
42 3 46
44 23 53
45 18 51
45 7 52
47 18 55

Deu como resultado

1 3 9 0 12 17 21 0 24 4 28 1 29 0 31 9
38 0 39 10 44 23 53 18 55 0

Para você fazer

Obviamente, antes de executar este caso, você está convidado (intimado !) a submeter sua solução ao online.judge da UVA. Se você obtiver o ACCEP-TED no site, pode exigir um 10 nesta folha independente do que o professor (e seu corretor) acharem. Combinado ?

Para deixar as coisas mais ágeis, considere que todos os números deste caso são menores do que 100. Considere também que serão 10 prédios.

Eis os seus dados aqui:

1 8 9
4 10 13
8 15 15
15 4 18
16 9 23
40 9 41
41 2 46
47 1 51
47 14 48
49 12 57

Responda aqui:

Apoio à folha hb1

Linha skyline

Considerando 10 prédios

prédio	início	altura	fim
1	1	18	11
2	4	25	7
3	7	6	15
4	17	26	25
5	19	25	25
6	21	12	26
7	23	28	26
8	29	19	30
9	37	16	42
10	40	22	47

• Digite

calcular

A skyline é

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0

Para testar

a=[1,18,11],[4,25,7],[7,6,15],[17,26,25],[19,25,25],[21,12,26],
[23,28,26],[29,19,30],[37,16,42],[40,22,47]]

deve dar:

1 18 4 25 7 18 11 6 15 0 17 26 23 28 26 0 29 19 30 0 37 16 40 22 47 0



509-76182 - gar a