

Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que – cada uma – e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

1. sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
2. sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
3. em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `windowspython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
4. Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente JULIA e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a in-

dentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a “entrada de dados” meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assumira esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

<code>int(a)</code>	converte o string a em inteiro (se possível)
<code>float(a)</code>	converte o string a em flutuante (idem)
<code>str(b)</code>	converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

🔧 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

1. Os 3 primeiros dígitos de `fat(39)`
2. As raízes de $20x^2 + 13x + 7$
3. O volume da esfera de raio 2.68000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que – cada uma – e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

1. sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
2. sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
3. em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `windowspython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
4. Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente JULIA e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a in-

dentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = sen(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a “entrada de dados” meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assumira esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

- `int(a)` converte o string a em inteiro (se possível)
- `float(a)` converte o string a em flutuante (idem)
- `str(b)` converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

🔧 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

1. Os 3 primeiros dígitos de `fat(39)`
2. As raízes de $20x^2 + 29x + 24$
3. O volume da esfera de raio 1.43000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(31)`
- As raízes de $23x^2 + 26x + 16$
- O volume da esfera de raio 1.25000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	a++;	a=a+1
$a = a^2$	pow(a,2);	a=a**2
$a = a - 3$	a=a-3;	a=a-3
$a = a * x$	a=a*x;	a=a*x
$a = a/x$	a=a/x;	a=a/x (int) a=a/x (float)
\sqrt{x}	sqrt(x);	a**0.5
$a + bi$?	a+bj
$a * (b + c)$	a*(b+c);	a*(b+c)
$b = \text{sen}(a)$	b=sin(a);	import numpy as np b=np.sin(a)

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	int a; cin>>a;	a=int(input('msg'))

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	cout<< a;	print(a)
		print('msg')
		print('m',a)

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(32)`
- As raízes de $21x^2 + 3x + 14$
- O volume da esfera de raio 1.89000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY	e frente
MATPLO-TLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSOR FLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(39)`
- As raízes de $18x^2 + 21x + 17$
- O volume da esfera de raio 1.62000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que – cada uma – e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

1. sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
2. sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
3. em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `windowspython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
4. Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente JULIA e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a in-

dentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a “entrada de dados” meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

<code>int(a)</code>	converte o string a em inteiro (se possível)
<code>float(a)</code>	converte o string a em flutuante (idem)
<code>str(b)</code>	converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

🔧 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

1. Os 3 primeiros dígitos de `fat(32)`
2. As raízes de $24x^2 + 18x + 31$
3. O volume da esfera de raio 1.12000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	a++;	a=a+1
$a = a^2$	pow(a,2);	a=a**2
$a = a - 3$	a=a-3;	a=a-3
$a = a * x$	a=a*x;	a=a*x
$a = a/x$	a=a/x;	a=a/x (int) a=a/x (float)
\sqrt{x}	sqrt(x);	a**0.5
$a + bi$?	a+bj
$a * (b + c)$	a*(b+c);	a*(b+c)
$b = \text{sen}(a)$	b=sin(a);	import numpy as np b=np.sin(a)

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	int a; cin>>a;	a=int(input('msg'))

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	cout<< a;	print(a)
		print('msg')
		print('m',a)

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

📖 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(37)`
- As raízes de $18x^2 + 7x + 11$
- O volume da esfera de raio 3.34000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assumira esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)     converte o string a em inteiro (se possível)
float(a)   converte o string a em flutuante (idem)
str(b)     converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(37)`
- As raízes de $6x^2 + 23x + 27$
- O volume da esfera de raio 2.45000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(31)`
- As raízes de $24x^2 + 5x + 23$
- O volume da esfera de raio 2.53000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assumira esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(31)`
- As raízes de $5x^2 + 13x + 17$
- O volume da esfera de raio 3.08000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que – cada uma – e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

1. sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
2. sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
3. em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `windowspython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
4. Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente JULIA e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a in-

dentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a “entrada de dados” meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

<code>int(a)</code>	converte o string a em inteiro (se possível)
<code>float(a)</code>	converte o string a em flutuante (idem)
<code>str(b)</code>	converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

🔧 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

1. Os 3 primeiros dígitos de `fat(36)`
2. As raízes de $18x^2 + 17x + 22$
3. O volume da esfera de raio 1.09000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que – cada uma – e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

1. sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
2. sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
3. em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `windowspython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
4. Jupyter: um ambiente `sui-generis`, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente JULIA e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e	
MATPLO-TLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a in-

dentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
var = var2, como por exemplo: `a=b`
var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a “entrada de dados” meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

- `int(a)` converte o string a em inteiro (se possível)
- `float(a)` converte o string a em flutuante (idem)
- `str(b)` converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

🔧 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

1. Os 3 primeiros dígitos de `fat(36)`
2. As raízes de $8x^2 + 6x + 9$
3. O volume da esfera de raio 2.25000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assumira esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(37)`
- As raízes de $18x^2 + 3x + 15$
- O volume da esfera de raio 2.96000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à
SYMPY e	frente
MATPLO-	
TLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSOR FLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(36)`
- As raízes de $19x^2 + 9x + 8$
- O volume da esfera de raio 2.37000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que – cada uma – e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

1. sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
2. sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
3. em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `windowspython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
4. Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente JULIA e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a in-

dentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a “entrada de dados” meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assumira esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

<code>int(a)</code>	converte o string a em inteiro (se possível)
<code>float(a)</code>	converte o string a em flutuante (idem)
<code>str(b)</code>	converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

🔧 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

1. Os 3 primeiros dígitos de `fat(39)`
2. As raízes de $27x^2 + 26x + 23$
3. O volume da esfera de raio 3.67000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(36)`
- As raízes de $19x^2 + 10x + 14$
- O volume da esfera de raio 3.61000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY	e frente
MATPLO-TLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSOR FLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
var = var2, como por exemplo: `a=b`
var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

- `int(a)` converte o string a em inteiro (se possível)
- `float(a)` converte o string a em flutuante (idem)
- `str(b)` converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(34)`
- As raízes de $24x^2 + 19x + 7$
- O volume da esfera de raio 3.90000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que – cada uma – e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

1. sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
2. sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
3. em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `windowspython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
4. Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente JULIA e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e	
MATPLO-TLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a in-

dentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
var = var2, como por exemplo: `a=b`
var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a “entrada de dados” meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

- `int(a)` converte o string a em inteiro (se possível)
- `float(a)` converte o string a em flutuante (idem)
- `str(b)` converte o numérico b em string

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`. A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

🔧 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

1. Os 3 primeiros dígitos de `fat(31)`
2. As raízes de $10x^2 + 17x + 28$
3. O volume da esfera de raio 2.56000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(36)`
- As raízes de $12x^2 + 19x + 10$
- O volume da esfera de raio 2.59000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(37)`
- As raízes de $25x^2 + 26x + 11$
- O volume da esfera de raio 1.76000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY	e frente
MATPLO-TLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSOR FLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)     converte o string a em inteiro (se possível)
float(a)   converte o string a em flutuante (idem)
str(b)     converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(33)`
- As raízes de $5x^2 + 15x + 25$
- O volume da esfera de raio 3.95000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

📖 Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(32)`
- As raízes de $5x^2 + 8x + 22$
- O volume da esfera de raio 2.35000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e MATPLOTLIB	
ASTROPY	Astronomia, mas com profundo impacto no tratamento de imagens
TENSORFLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)   converte o string a em inteiro (se possível)
float(a) converte o string a em flutuante (idem)
str(b)   converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code> <code>print('msg')</code> <code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(33)`
- As raízes de $22x^2 + 2x + 20$
- O volume da esfera de raio 2.56000

Responda neste espaço

1.	2.	3.
----	----	----



Introdução a Python

Nascido no finzinho do século XX, e portanto herdeiro de:

- centenas de boas linguagens de programação que vieram antes e que - cada uma - e todas tinham coisas boas ou ótimas que deviam ser aproveitadas (C, smalltalk, APL, Pascal, C++, Java,...). Aproveitadas é modo educado de falar: as coisas deviam ser copiadas *as is*.
- máquina sobrando: até aqui, faziam-se das tripas coração para economizar uns poucos bytes ou alguns ciclos de CPU. Isto não era mais necessário: havia e há máquina sobrando.

Tenha essas duas coisas em mente ao procurar entender porque o Python é a linguagem mais usada no mundo em 2022 e mais além. Se tiver alguma dúvida consulte <https://www.tiobe.com/tiobe-index/>. Aqui vale a pena lembrar aquela velha propaganda: *Tostines vende mais porque é fresquinho ou é fesquinho porque vende mais ?*

Instalação

Na continuação, você precisa instalar e usar algum Python. As alternativas

- sob windows: vá em python.org.br e siga as instruções para ter o ambiente no seu computador (você precisa ser o administrador).
- sob android: vá na loja do android e busque `python`. Escolha uma opção (pode ser o QPython 3) e mande ver. Mas, tenha em mente que a ergonomia aqui não é o ponto forte (basicamente pela dificuldade de usar o teclado).
- em um pendrive: Se você tiver que usar computadores públicos e/ou emprestados, uma alternativa é ir a algum repositório público (por exemplo sourceforge.net) e lá baixar o `winpython` possivelmente em um pendrive: você terá um ambiente Python completo podendo ser usado em qualquer computador. Só precisa ter alguma paciência que o dito cujo é grande.
- Jupyter: um ambiente sui-generis, mistura de compilador com editor de textos (este usando o padrão Markdown). Funciona em uma interface HTML (usada na web) e permite misturar textos e códigos. Aceita igualmente Julia e há outros ambientes na fila (C++?).

Freeware - Não é desimportante que o Python entregue tudo o que ele entrega a custo zero. Seus concorrentes em boa medida cobram licenças de centenas ou milhares de dólares. Sem contar que a comunidade Python é a maior do mundo.

Pacotes - milhares deles. Eis alguns que podem interessar:

NUMPY,	serão objeto de aulas mais à frente
SYMPY e	MATPLO-
TLLIB	ASTROPY
Astronomia, mas com profundo impacto no tratamento de imagens	
TENSOR FLOW	A rede neural do Google
PYGAME	A base para gerar jogos e animações
PANDAS	ferramentas para Data analysis
OPENCV	Bibliotecas para computer vision
TURTLE	A célebre Tartaruga de Seymour Papert no ambiente Logo
NLTK	Natural Language Toolkit
PILLOW	Ferramentas para tratar imagens digitais

Algumas pérolas

Digite o programa abaixo. Ele calcula e retorna o fatorial de um número inteiro e positivo (a indentação precisa ser religiosamente observada)

```
def fat(x):
    if x>1:
        return x*fat(x-1)
    else:
        return 1
```

Tendo digitado e salvo (escolha um nome qualquer) o programa acima, é hora de executá-lo: Chame `fat(5)` e a resposta deverá ser, como sabemos, 120. Agora chame `fat(100)` e você deverá ter um número enorme. Veja que nenhuma modificação foi necessária no código.

Baskara: A seguir, digite o programa abaixo, que como se pode ver, acha as 2 raízes de uma equação de segundo grau: $ax^2 + bx + c = 0$.

```
def baskara(a,b,c):
    return [(-b+(b**2-4*a*c)**0.5)/2*a, \
            (-b-(b**2-4*a*c)**0.5)/2*a]
```

A seguir, de volta ao prompt do Python digite `baskara(1,4,-10)` e veja o resultado. Aparecem x_1 e x_2 , nenhuma novidade. Mas, agora execute `baskara(1,4,10)`. Perceba que agora as raízes são complexas, mas, de novo, nenhuma novidade, exceto que talvez a unidade imaginária que geralmente é i agora virou j pelo risco de confusão com alguma coisa.

Volume de uma esfera: A questão agora é calcular o volume de uma esfera de raio dado. Como se sabe da geometria $V = 4/3\pi \times r^3$. Então, o programa Python que calcula este volume é

```
def volesf(r):
    pi=3.14159265
    return pi*r**3
```

Agora chamando `volesf(1)` teremos a resposta que é igual a 3.14159265

Uma linguagem fácil de aprender

A seguir, alguns conceitos introdutórios que nos ajudarão a começar a navegação no oceano chamado Python. Lembrando que estamos falando de máquinas de Von Neumann e seus 5 comandos: (E/S, movimentação, aritmética e lógica, condicional e desvio).

atribuição a criação de uma variável, ou a alteração de seu valor

Pseudo-cód	C++	Python
A ← valor	int a=valor;	a=valor

O comando de atribuição pode ser dividido em 3 possibilidades:

var = constante, como por exemplo: `a=0`
 var = var2, como por exemplo: `a=b`
 var = expressão, como por exemplo: `a=a+1`

Aritmética Tudo o que sabemos da matemática vale aqui, sem exceção.

Pseudo-cód	C++	Python
$a = a + 1$	<code>a++;</code>	<code>a=a+1</code>
$a = a^2$	<code>pow(a,2);</code>	<code>a=a**2</code>
$a = a - 3$	<code>a=a-3;</code>	<code>a=a-3</code>
$a = a * x$	<code>a=a*x;</code>	<code>a=a*x</code>
$a = a/x$	<code>a=a/x;</code>	<code>a=a/x (int)</code> <code>a=a/x (float)</code>
\sqrt{x}	<code>sqrt(x);</code>	<code>a**0.5</code>
$a + bi$?	<code>a+bj</code>
$a * (b + c)$	<code>a*(b+c);</code>	<code>a*(b+c)</code>
$b = \text{sen}(a)$	<code>b=sin(a);</code>	<code>import numpy as np</code> <code>b=np.sin(a)</code>

entrada de dados - Como os dados são obtidos

Pseudo-cód	C++	Python
leia a	<code>int a;</code> <code>cin>>a;</code>	<code>a=int(input('msg'))</code>

Uma maneira alternativa de fazer a "entrada de dados" meio fake é definir uma função recebendo parâmetros e imediatamente a seguir chamar essa mesma função com valores já digitados no mesmo script. Por exemplo

```
def dobro(a):
    return a*2
dobro(5)
```

Ajuda durante os testes não ter que redigitar as vezes grandes volumes de dados a cada teste. Quando tudo funcionar adequadamente, a função estará apta a funcionar com os dados da hora.

Tipos de dados - Embora o Python assuma esta responsabilidade (ao contrário de C++, por exemplo), ainda assim às vezes será necessário especificar algum tipo particular. Lembrando que os tipos básicos são `int`, `float`, `str` e `logic (True e False)`. Existem funções Python que fazem algumas conversões:

```
int(a)     converte o string a em inteiro (se possível)
float(a)   converte o string a em flutuante (idem)
str(b)     converte o numérico b em string
```

Saída de dados - Como os resultados são comunicados

Pseudo-cód	C++	Python
escreva a	<code>cout<< a;</code>	<code>print(a)</code>
		<code>print('msg')</code>
		<code>print('m',a)</code>

Uma grande vantagem do `print()` é que não é necessário compatibilizar os formatos numérico e caracter (como por exemplo em `print('valor = ',val)`). A função `print()` se encarrega disso.

Python é interpretado

Se isso acarreta um desempenho menor (e quem se importa ?) em compensação ele têm diversas vantagens

- uso como calculadora
- aceita scripts e não apenas programas
- permite execução passo a passo, interativa, inclusive mudando o valor de variáveis

Conceito de bloco

Para atender à programação estruturada, qualquer linguagem de programação precisa implementar o conceito de bloco: um conjunto que qualquer quantidade de comandos e/ou blocos que são tratados como se fossem um único comando.

Em C++, tal conceito é implementado pelo uso de chaves. Tudo o que estiver entre `{ e }` é tratado como um bloco. Em Python, e com o objetivo de não poluir o código, usa-se a indentação para estabelecer os blocos. Portanto, o que lá no C++ era apenas uma recomendação para ajudar o leitor (humano) a entender o código, em Python é mandatório: Um código bagunçado no Python não levará a lugar algum, certamente dando erros sintáticos ou semânticos.

Ainda falta muita coisa

Não é muita, mas é bem importante. Ainda esta-se limitado aqui no Python por não dispor do comando condicional (`if condição:`) nem dos comandos de repetição (`while condição:` ou `for...`), mas eles virão nas próximas aulas.

Para você fazer

Você deve arrumar um acesso a algum Python e digitar os 3 scripts acima e descobrir:

- Os 3 primeiros dígitos de `fat(37)`
- As raízes de $6x^2 + 17x + 29$
- O volume da esfera de raio 3.16000

Responda neste espaço

1.	2.	3.
----	----	----

