Programação Script

Introdução, Hello World e Variáveis

Aula 01

Prof. Felipe A. Louza

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

Introdução

O que significa "programar" um computador?

- Fornecer um conjunto de instruções que indica o que deve ser feito[1]
- Essa sequência de passos bem definida é um **algoritmo**
- Um algoritmo/programa pode ser bem simples (ex. calculadora) ou altamente complexo (ex. controle de vôos)

[1] Como em um GPS, só que ao contrário.

Algoritmo

Exemplo:

- Determinar se um aluno foi aprovado ou reprovado ($NF \geq 60$)
  1. Obtenha a média final (NF)
  2. Se $NF \geq 60$ então:
     1. Informe "Aprovado!!"
  3. Senão:
     1. Informe "Reprovado!!"

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

O que é um computador?

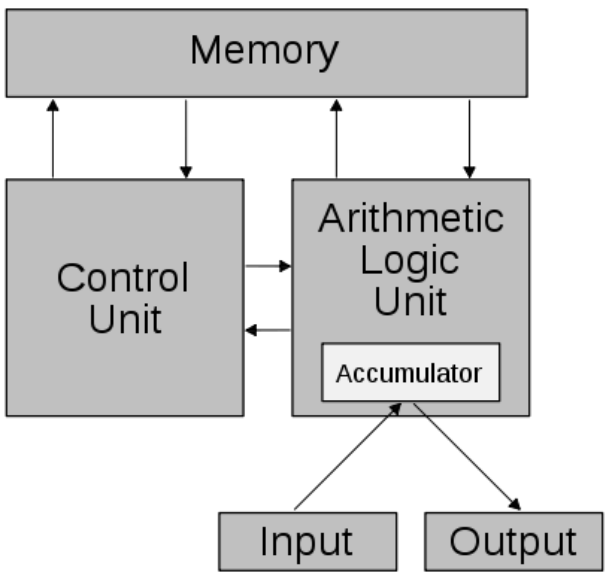
Um computador é uma máquina que, a partir de uma entrada, realiza um número muito grande de cálculos matemáticos e lógicos, gerando uma saída.

- Computadores fazem isto muito rápido. Enquanto leio esta frase um computador típico executou mais de 1 Bilhão de instruções.

Hardware e dispositivos

Usualmente chamamos de **Hardware** todos os dispositivos físicos que compõem um computador, como CPU, Disco Rígido, Memória, etc.

- Estes dispositivos seguem uma organização básica como na figura (*Arq. de Von Neumann*).



Hardware e dispositivos

Todo o hardware opera com sinais digitais: sem energia e com energia. Usamos valores **0** e **1** para representar isto.

- Chamamos estes sinais de **Bit** $\rightarrow$ Valores **0** ou **1**.
- Chamamos de **Byte** $\rightarrow$ um agrupamento de 8 bits.
- Todas as informações armazenadas no computador são representadas por números **0**s e **1**s.
  - Informações como letras, símbolos, imagens, programas são todas vários 0s e 1s.

0	0	0	0	0	0	1	0

Figura: 1 byte

Software

Softwares são os programas que executam tarefas utilizando o hardware de um computador

- Os softwares são compostos por um conjunto de instruções[2] que operam o hardware.
- Temos abaixo, por exemplo, três instruções para um computador:

```
01000010 00110101 01010100 00110110
01001110 11001100 10010110 01101000
00000101 11111110 11010011 00001100
```

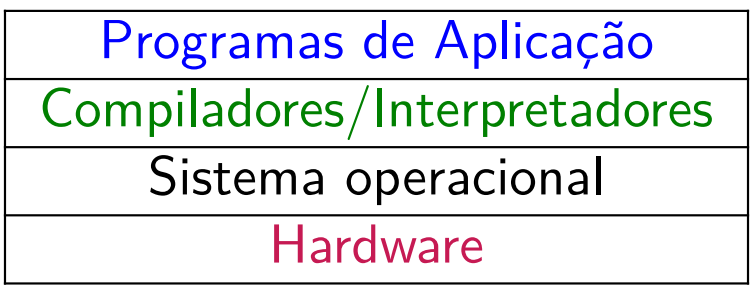
- Um software é composto por milhares de instruções deste tipo.

[2] Ou seja, algoritmos.

Linguagens de Programação

Neste curso iremos construir novos programas.

- Para isso, podemos escrever diretamente códigos binários que serão executados por um computador.
- Mas usaremos uma linguagem de programação (de alto nível) específica para gerar o nosso programa.
  - Programas escritos em linguagens de alto nível precisam ser "traduzidos" antes que possam rodar.



Compiladores/Interpretadores

Existem duas formas de traduzir um programa escrito em uma linguagem de programação para ser executado pelo computador.

1. Um **compilador** é um programa que "traduzirá" a totalidade dos comandos em um programa.
   - C, C++, FORTRAN, ...

```
for(i=0; i<10; i++)        loop: add c, a, b        01000010 00110101 01010100 00110110
    c = a+b;                     add i, i, 1        01100110 01110101 01010100 00110110
                                 bnq i, 10, loop    11110000 01110101 01010100 00110110
```

Compiladores/Interpretadores

Outra forma:

2. Um **interpretador** toma um comando de cada vez e "traduz" o seu significado, executando-o em uma máquina virtual (MV), que simula um computador.
   - Lisp, Perl, Ruby, ...

   - Python é "interpretada/compilada"[3]

[3] O programa é traduzido para uma linguagem intermediária (bytecode) que é interpretado pela MV do Python.

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

A linguagem Python

Python (versão 3):

- Criada por Guido Van Rossum
- Primeira versão em 1991
- Comunidade dinâmica
- Muitas bibliotecas e recursos disponíveis

Primeiro programa em Python

Um programa em Python é um arquivo texto, contendo declarações e operações da linguagem.

- Isto é chamado de *código fonte*.

```
1 print("Ola turma de PS!!")
```

Você pode salvar este arquivo como hello.py

Como executar este programa

Para executar este programa basta abrir um `terminal`[4] e escrever:

```
1 $ python3 hello.py
2 Ola turma!!
```

[4] No Windows, você pode usar: `idle3`, `pycharm`, ...

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

Shell Interativa

Abra um terminal de comando e execute `python3`.

- Se **Python** estiver instalado em seu computador será inicializado a shell de Python.

```
$ python3
Python 3.8.6 (default, Sep 25 2020, 00:00:00)
[GCC 10.2.1 20200723 (Red Hat 10.2.1-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Shell Interativa

Você pode executar comandos diretamente na shell.

```
$ python3
Python 3.8.6 (default, Sep 25 2020, 00:00:00)
[GCC 10.2.1 20200723 (Red Hat 10.2.1-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Ola turma")
Ola turma
>>> 5+5
10
>>>
```

Shell Interativa

A `shell iterativa` é muito útil durante a criação de um programa para testar partes do seu código:

- Mas na maioria das vezes criaremos um código completo que deve ser salvo em um arquivo com a extensão `.py`.
- Depois este código poderá ser executado em um terminal da seguinte forma

```
$python nomeArquivo.py
```

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

Estrutura Básica de um Programa em Python

Um programa em **Python** é uma sequencia de definições e comandos que serão executados pelo interpretador.

```
1 Comando1
2 .
3 .
4 .
5 ComandoN
```

- O programa deve ter um comando por linha.
- Os comandos serão executados nesta ordem, de cima para baixo, um por vez.

Estrutura Básica de um Programa em Python

Exemplos:

1. ✓

```
1 print("Ola turma!")
2 print("Vamos programar em Python")
```

2. ✗

```
1 print("Ola turma!")  print("Vamos programar em Python")
```

   – Este programa gera um erro pois temos dois comandos em uma mesma linha.

3. ✓

```
1 print("Ola turma!"); print("Vamos programar em Python")
```

Vamos usar o padrão de um comando por linha.

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

Objetos

Um programa executa comandos para manipular informações/dados.

- Qualquer dado em **Python** é um objeto, que é de um certo **tipo** específico.
  - O **tipo** de um objeto especifica quais operações podem ser realizadas sobre o objeto.
  - Por exemplo, o **número 5** é representado com um objeto do tipo `int` em **Python**.

```
1 >>> type(5)
2 <class 'int'>
```

  - No **Python** temos três tipos básicos:
    1. `int`;
    2. `float`; e
    3. `string`

O Comando `type` informa o tipo de um objeto associado a um objeto.

Tipo `int`

Números inteiros: tipo `int`.

1. Positivos e negativos:

```
>>> type(3)
<class 'int'>
>>> type(1024)
<class 'int'>
>>> type(-7)
<class 'int'>
```

Tipo `float`

Números racionais: tipo `float`.

1. Decimais exatos:

```
>>> type(2.4142)
<class 'float'>
```

2. Dizimas periódicas:

```
>>> 10/3
3.3333333333333335
>>> type(10/3)
<class 'float'>
```

   - O computador tem uma limite de precisão pois há uma quantidade limitada de memória. Nesses casos o número é *"arredondado"*.

```
>>> 0.1+0.2
0.30000000000000004
```

Tipo `str`

Strings: tipo `str`.

1. Um texto (cadeia de caracteres):

```
>>> type("Ola turma!")
<class 'str'>
```

2. Toda string é representada entre aspas simples ou duplas:

```
>>> type("5")
<class 'str'>
```

   - 5 é um número inteiro, mas como está entre aspas é uma string.

Outros tipos

Outros tipos básicos: `booleanos`, `bytes`, `listas`, `tuplas`, ... serão vistos ao longo do curso.

Variáveis

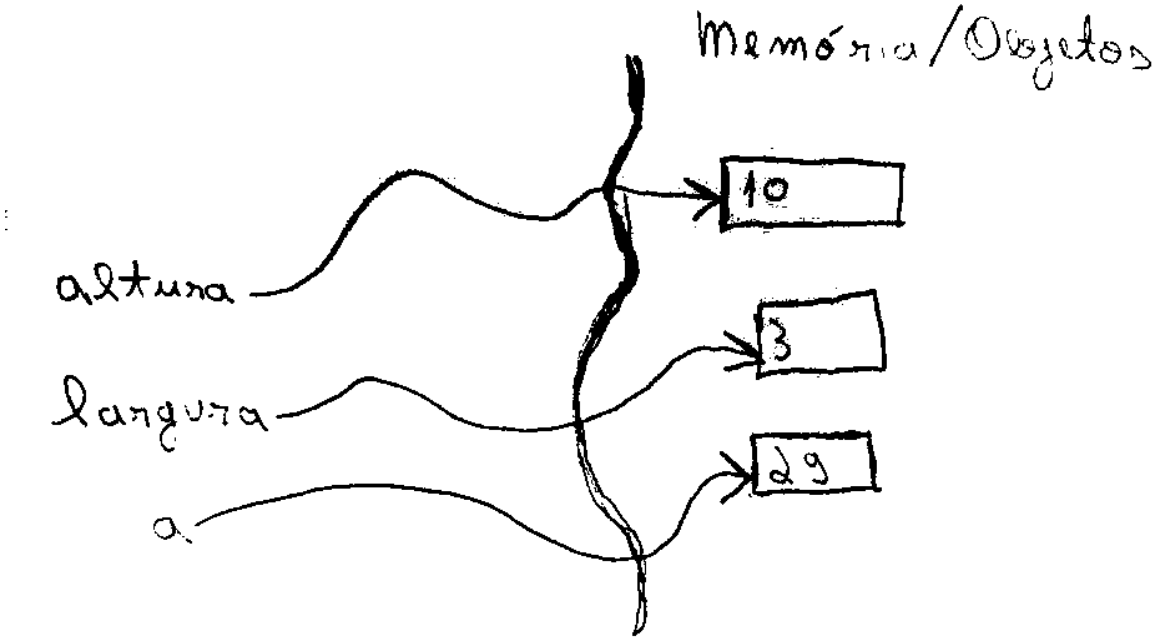
Definição

Uma variável é um espaço na memória (com um nome associado) utilizado para guardar valores de um objeto

- No exemplo abaixo associamos os nomes `altura`, `largura` e `a` com os valores 10, 3, e 29.

```
1 altura = 10
2 largura = 3
3 a = 29
```

Variáveis



Variáveis

Regras para nomes de variáveis:

- **Deve** começar com uma letra (maíuscula ou minúscula) ou subscrito(_). **Nunca** pode começar com um número.
- Pode conter letras maiúsculas, minúsculas, números e subscrito.
- Não pode-se utilizar como parte do nome de uma variável:

```
{ ( + - * / \ ; . , ?
```

- Letras maiúsculas e minúsculas são diferentes:

```
c = 4
C = 3
```

Variáveis

Regras para nomes de variáveis:

- Além disso existem palavras reservadas da linguagem:

```
and       as      assert   break     class    continue
def       del     elif     else      except   exec
finally   for     from     global    if       import
in        is      lambda   nonlocal  not      or
ass       raise   return   try       while    with
yield     True    False    None
```

Atribuição

O comando = do **Python** é o comando de atribuição:

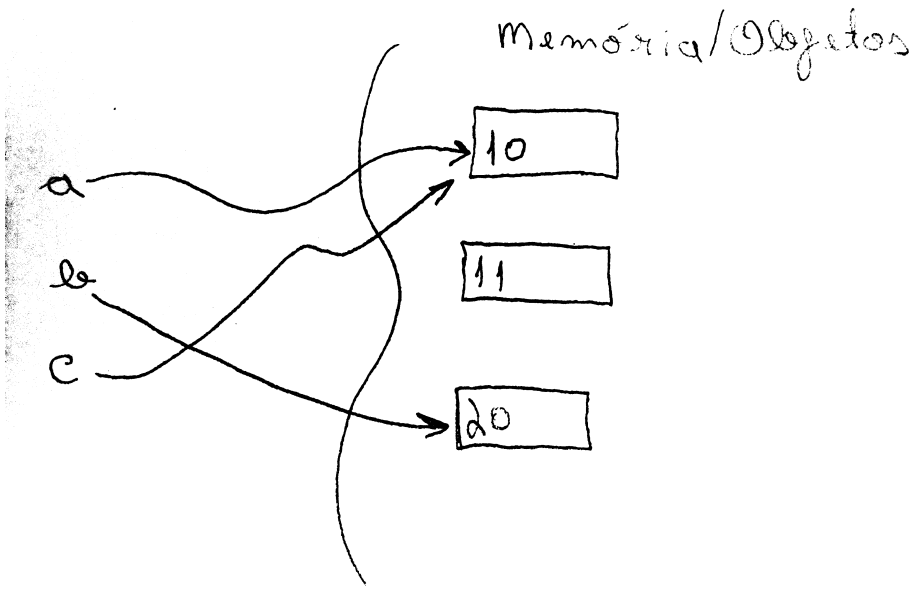
- Ele associa a variável do lado esquerdo do comando com o objeto do lado direito.

```
1 >>> a = 10
2 >>> b = 11
```

Atribuição

No exemplo abaixo, o objeto 10 terá duas variáveis associadas com ele, o objeto 20 uma, e 11 nenhuma.

```
1 >>> a = 10
2 >>> b = 11
3 >>> c = 10
4 >>> b = 20
```



Atribuição

O comando `id()` retorna o identificador de um objeto na memória.

```
1 >>> a = 10
2 >>> b = 11
3 >>> id(a)
4 140181897291968
5 >>> id(b)
6 140181897292000
7 >>> b = 10
8 >>> id(b)
9 140181897291968
```

Comando de Atribuição

O comando de atribuição pode conter expressões do lado direito:

```
variável = expressão
```

- Atribuir um valor de uma expressão para uma variável significa calcular o valor daquela expressão e somente depois associar o valor calculado com a variável.

```
1 >>> a = 3 + 10
2 >>> b = (6.57 * 90) + 40
3 >>> a
4 13
5 >>> b
6 622.3
```

Atribuição

Se uma variável for usada sem estar associada com nenhum objeto, um erro ocorre.

```
>>> a = 10
>>> b = 10
>>> a = a+b
>>> a
20
>>> a = a + c
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
NameError: name 'c' is not defined
```

- Nesse exemplo não podemos usar a variável c, pois esta não foi definida (associada com algum objeto).

Tipagem em Python

Uma variável em **Python** possui o tipo correspondente ao objeto que ela está associada naquele instante.

- **Python** não possui tipagem estática como outras linguagens.
  - Isto significa que você pode atribuir objetos de diferentes tipos para uma mesma variável.
  - Como uma variável não possui tipo pré-definido, dizemos que **Python** tem **tipagem dinâmica**.
- O programa abaixo é perfeitamente legal em **Python**:

```
1 >>> a = 3
2 3
3 >>> a = 90.45
4 90.45
5 >>> a = "Ola voces!"
6 "Ola voces!"
```

Em outras linguagens cria-se variáveis de tipos específicos e elas só podem armazenar valores daquele tipo (**tipagem forte**).

Fim

Dúvidas?

Roteiro

1. Introdução
2. Hardware e Software
3. A linguagem Python
4. Shell Interativa
5. Estrutura de um Programa em Python
6. Objetos, Variáveis e Atribuição
7. Referências

Referências

1. Materiais adaptados dos slides do Prof. Eduardo C. Xavier, da Universidade Estadual de Campinas.