Algoritmos e Programação de Computadores

Variáveis, Objetos e Atribuição

Raquel Cabral

Dr. Eng. Eletrica, UFMG

mjara.perez@ic.unicamp.br

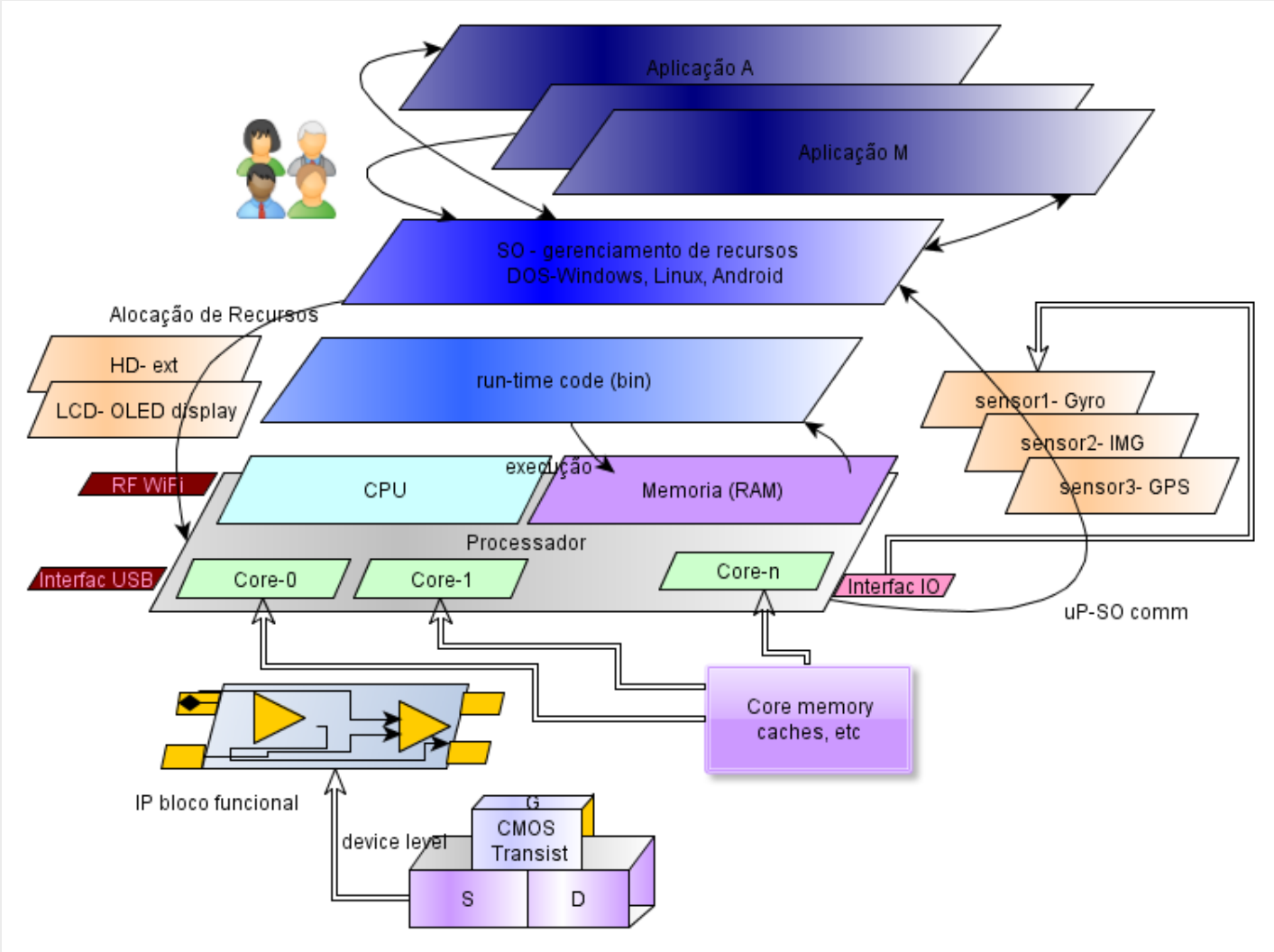
Ref.: material original (1o S., T. KLMN). por **Profa. Sandra Avila**, Instituto de Computação (IC/ Unicamp)

MC102 Turma Z, 07 Agosto, 2018

Agenda

- Estrutura básica de um sistema de computação
- A linguagem de programação Python
- Estrutura básica de um programa em Python
- Objetos, Variáveis e Atribuição
- Tipos de Objetos: int, float, string

Arquitetura Sistema de Computação

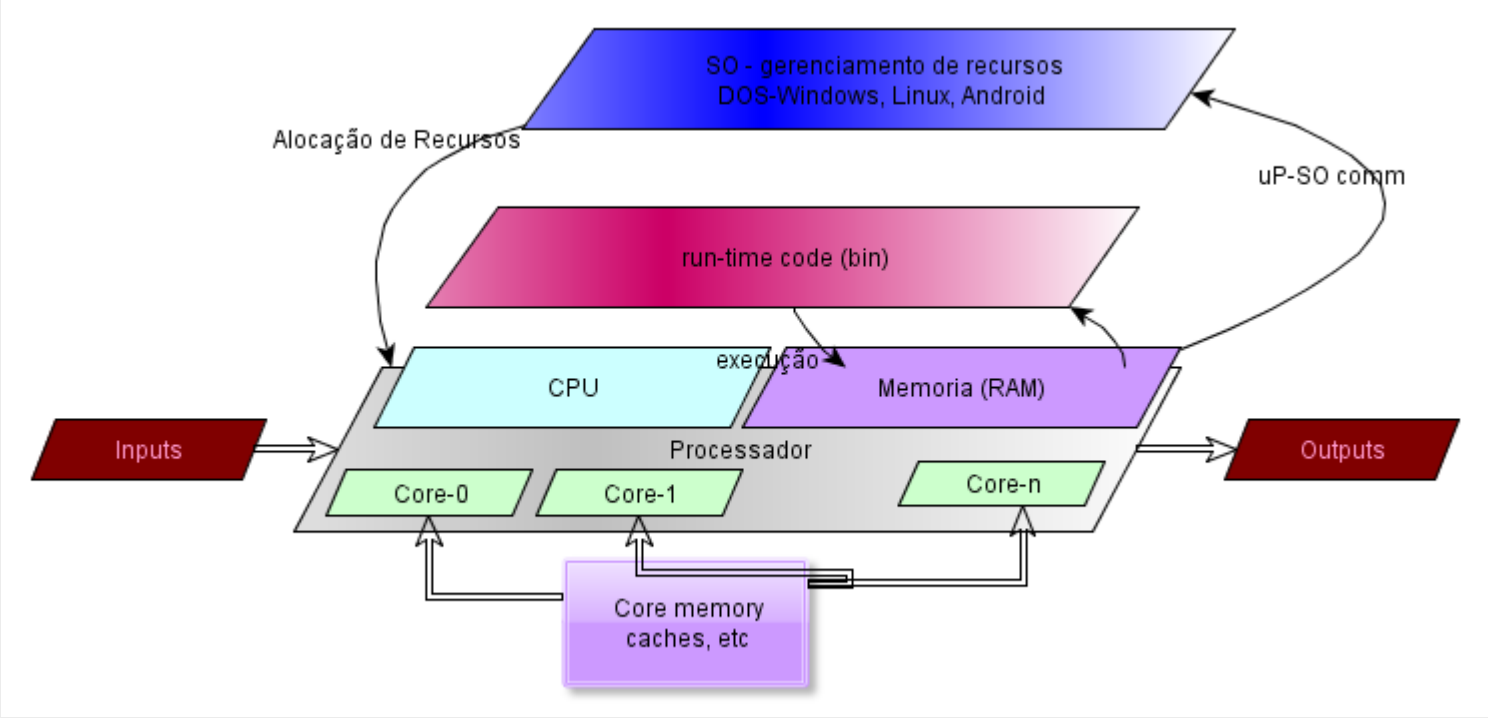


Execução de um Programa

- Programas são normalmente armazenados na memória do sistema de computação, em formato a ser executado pela máquina
- O computador digital só consegue executar programas em código possível de ser interpretado pela CPU, i.e., via código **binário**.
- A execução do programa é realizada de forma sequencial, via leitura dos comandos de execução, i.e. do código armazenado em memória

Execução:

Programa armazenado na memória (ex., RAM)



A Linguagem de Programação Python

- Python é um exemplo de **linguagem de programação de alto nível**.
- O computador executa programas em código binário, via **linguagens de baixo nível** (“linguagens de máquina” ou “linguagens assembly”).
- Programas escritos em linguagens de alto nível (i.e., alto nível de abstração), precisam ser processados (interpretados, compilados) para ser executados pela máquina (micro-controlador, processador, computador, ou sistema de computação).

A Linguagem de Programação Python

- Dois tipos de programas processam linguagens de alto nível, traduzindo-as para linguagens de baixo nível: **interpretadores** e **compiladores**.
- **Interpretador:** lê um programa escrito em linguagem de alto nível e o executa, ou seja, faz o que o programa diz.

A Linguagem de Programação Python

- **Compilador:** lê o programa e o traduz completamente antes que o programa comece a rodar.

- O programa traduzido é chamado de **código objeto** ou **executável**.
- O Python usa ambos os processos, mas ela é em geral considerada uma linguagem interpretada.

A Linguagem de Programação Python

- Existem duas maneiras de usar o interpretador: no modo **linha de comando** (“shell mode”) e no modo de **script** (“program mode”).

  Linha de comando: você digita comandos em Python e o interpretador mostra o resultado.

```
$ python3
Python 3.6.1 |Anaconda 4.4.0 (64-bit)| (default, May 11 2017, 13:09:58)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

A Linguagem de Programação Python

- Existem duas maneiras de usar o interpretador: no modo **linha de comando** (“shell mode”) e no modo de **script** (“program mode”).

  Script: você pode escrever um programa inteiro em um arquivo e usar o interpretador para executar o conteúdo do arquivo como um todo.

```
$ python programa1.py
Meu primeiro programa soma os numeros 2 e 3:
5
```

```
print("Meu primeiro programa soma os numeros 2 e 3:")
print(2 + 3)
```

A Linguagem de Programação Python

- Por convenção, arquivos que contém programas em Python tem nomes que terminam com a extensão **.py**, ex: programa1.py

```
$ python programa1.py
Meu primeiro programa soma os numeros 2 e 3:
5
```

```
print("Meu primeiro programa soma os numeros 2 e 3:")
print(2 + 3)
```

A Linguagem de Programação Python

- Mais uma maneiras de usar o interpretador
- **Jupyter Notebook**: você digita comandos em Python e o interpretador mostra o resultado.

```
$ jupyter notebook
[I 10:26:52.652 NotebookApp] Serving notebooks from local directory: /home/
sandra
[I 10:26:52.652 NotebookApp] 0 active kernels
[I 10:26:52.652 NotebookApp] The Jupyter Notebook is running at: http://localhost:
8888/?token=1f73b8c2188d8d828b2ec376cad7a390ba4b975d1023a8fc
```

A Linguagem de Programação Python

arquivo.ipynb

Estrutura Básica de um Programa em Python

- Um **programa** é uma sequência de comandos que serão executados pelo interpretador.

```
comando 1
comando 2
…
comando n
```

- 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

```
print("Ola turma de MC102")
print("Vamos programar em Python \o/")
```

```
print("Ola turma de MC102") print("Vamos programar em Python \o/")
```

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

Estrutura Básica de um Programa em Python

```
print("Ola turma de MC102")
print("Vamos programar em Python \o/")
```

```
print("Ola turma de MC102") print("Vamos programar em Python \o/")
```

```
print("Ola turma de MC102"); print("Vamos programar em Python \o/")
```

Você pode usar um ponto e vírgula ao final de cada comando para usar vários comandos em uma mesma linha.

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 5 do tipo **int** em Python.

Objetos

```
print(type("Ola turma de MC102"))
print(type(5))
```

```
<class 'str'>
<class 'int'>
```

5 é um **inteiro**, do tipo **int**

"Ola turma de MC102" é uma **string ou texto cadeia de caracteres**, do tipo **str**

Objetos

```
print(type("5"))
```

```
<class 'str'>
```

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

Variáveis

- Variáveis são uma forma de se associar um nome dado pelo programador com um objeto.
- No exemplo abaixo associamos os nomes **altura**, **largura** e **a** com os valores 10, 3, e 29, respectivamente.

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

Variáveis: Diagrama de referência

- Diagrama de referência mostra o estado de cada variável em um **instante de tempo particular**.


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

Variáveis: Regras para Nomes

- **Deve** começar com uma letra (maiúscula ou minúscula) ou underscore(_). **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:
  { (+ - * / \ n ; . , ?
- Letras maiúsculas e minúsculas são diferentes: c = 4 C = 3

Variáveis: Regras para Nomes

```
76trombones = "grande parada"
mais$ = 1000000
class = "MC102"
```

O nome `76trombones` é ilegal pois **não começa com uma letra**.

`mais$` é ilegal pois contém um **caractere ilegal**, o símbolo de cifrão.

Mas o que está errado com `class`?

Variáveis: Regras para Nomes

- Ocorre que `class` é uma das **palavras reservadas** (keywords) de Python.
- As palavras reservadas definem a sintaxe da linguagem e sua estrutura e **não podem ser usadas como nomes de variáveis**.
- Python tem pouco mais de trinta palavras reservadas (e uma vez ou outra melhorias em Python introduzem ou eliminam uma ou duas).

Variáveis: Palavras Reservadas

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 do comando.
- Um objeto pode ter um nome associado com ele, mais de um nome ou nenhum nome.

Atribuição

- No exemplo abaixo, após todos comandos serem executados o objeto 10 terá duas variáveis associadas com ele, o objeto 20 uma, e 11 nenhuma.

```
a = 10
b = 11
c = 10
b = 20
```

Atribuição

- Se uma variável for usada sem estar associada com nenhum objeto, um erro ocorre.
- No exemplo abaixo não podemos usar a variável c, pois esta não foi definida (associada com algum objeto).

```
>>> a = 10
>>> b = 10
>>> a = a+b
>>> a
20
>>> a = a + c
```

Tipos de Objetos em Python

- Python possui os seguintes tipos básicos que veremos nesta aula:
  - **int**: Corresponde aos números inteiros. Ex: 10, -24.
  - **float**: Corresponde aos números racionais. Ex: 2.4142, 3.141592.
  - **str** ou **string**: Corresponde a textos. Ex: "Ola turma".
- Os tipos básicos booleanos, bytes, listas, tuplas, conjuntos e dicionários serão vistos ao longo do curso.

Tipo Inteiro

- Objetos do tipo **int** armazenam valores inteiros.
- Literais do tipo **int** são escritos comumente como escrevemos inteiros.
- Exemplos: 3, 1034, e -512.
- O tipo **int** possui precisão arbitrária (limitado a memoria do seu computador).

Tipo Ponto Flutuante

- Objetos do tipo **float** armazenam valores “reais”.
- Literais do tipo **float** são escritos com um ponto para separar a parte inteira da parte decimal. Exemplos: 3.1415 e 9.8.
- Possuem problemas de precisão pois há uma quantidade limitada de memória para armazenar um número real no computador.

Tipo Ponto Flutuante

- Notem no exemplo abaixo o erro de precisão:

```
>>> 1/10.0
0.1
>>> 0.1 + 0.2
0.30000000000000004
```

Dica :
0.1 poderia ser representado internamente pelo computador, por uma expressão matemática, do tipo: 1/16 + 1/32 + 1/256 + 1/512 + 1/4096 ...

Tipo Ponto Flutuante

```
>>> print(42000)
42000
>>> print(42,000)
42 0
>>> print(42.000)
42.0
```

Tipo String

- Objetos do tipo **string** armazenam textos.
- Um literal do tipo **string** deve estar entre aspas simples ou aspas duplas. Exemplos de **strings**:
  - 'Ola Brasil!' ou "Ola Brasil".
- Veremos posteriormente neste curso diversas operações que podem ser realizadas sobre objetos do tipo **string**.

Tipagem em Python

- Uma variável em Python possui o tipo correspondente ao objeto que ela está associada **naquele instante**.
- Python não possui tipagem forte 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 fraca**.
  - Em outras linguagens cria-se variáveis de tipos específicos e elas só podem armazenar valores daquele tipo para o qual foram criadas.
  - Estas últimas linguagens possuem **tipagem forte**.

Tipagem em Python

```
>>> a = 3
>>> print(a)
3
>>> a = 90.45
>>> print(a)
90.45
>>> a = "Ola voces!"
>>> print(a)
Ola voces!
```

Exercício

- Qual o valor armazenado na variável **a** no fim do programa?

```
d = 3
c = 2
b = 4
d = c + b
a = d + 1
a = a + 1
print(a)
```

Exercício

- Você sabe dizer qual erro existe neste programa?

```
d = 3.0
c = 2.5
b = 4
d = b + 90
e = c * d
a = a + 1
print(a)
print(e)
```

Referências hyperlinks

- https://ic.unicamp.br/~mjara.perez (links, ... TBD)

```
MC102, IC, Unicamp > https://ic.unicamp.br/~mc102
Python international community >  https://www.python.org/
Versão Brasileira>  https://python.org.br/
Wikipedia >  https://pt.wikipedia.org/wiki/Python
Python Tutorial > https://docs.python.org/3/tutorial/
Django Girls (Tutorial em português): https://
tutorial.djangogirls.org/pt/python_introduction/
Jupyter Python : http://jupyter.org/about
Jupyter Python em português
https://www.concrete.com.br/2017/07/21/introducao-ao-python-com-
jupyter/
```