

# Algoritmo e Implementação

Prof. Maurício Gagliardi Palma

[mauricio.palma@ufscar.br](mailto:mauricio.palma@ufscar.br)

## Algoritmo

Uma sequência **finita** de instruções **bem definidas** e **não ambíguas** para resolver um problema ou realizar algum objetivo

**Cada instrução deve ser possível de ser cumprida.**

## Programa / Software

Sequência finita de instruções em uma linguagem **compreendida pelo computador**

A solução de um problema usando computador envolve:

- Entendimento do problema (análise)
- Criação de uma sequência de operações que, quando executadas, produzem a solução para o problema
  - Projeto da solução - algoritmo
  - Implementação (codificação) - em uma linguagem de programação
- **Execução desta sequência de operações (feita pelo computador)**
- Verificação da adequação da solução

Para executar os algoritmos, o computador utiliza:

- Estruturas e instruções para armazenamento e manipulação das informações
- Estruturas de controle, para manipulação do fluxo de execução das instruções
  - Estrutura sequencial
  - Estruturas de seleção (condicional)
  - Estruturas de repetição

- Podem ser classificados em:
  - **Dados de entrada:** informações que fornecemos para o computador
  - **Dados intermediários:** gerados pelo computador durante a execução do programa, que não interessam como resultado final, mas que são necessários para a obtenção do resultado final
  - **Dados de saída:** informações que o computador fornece como resultado após manipular os dados de entrada

- **Numéricos**: números que podem ser inteiros ou reais
  - Ex: valor inicial do investimento, taxa, período, notas
    - **Inteiro** = {..., -1, 0, 1, 2, ....., 5527 ...}
    - Real (**ponto flutuante**) = { ..., -1.0, 0.0, 1.0, 2.5, ....}
- **Literais**: caractere ou sequência de caracteres (normalmente indicamos dados desse tipo entre `“”` ou `’ ’` )
  - Ex: nome, endereço, sintoma do paciente
    - **Caractere** = ‘a’, ‘b’, ‘B’, ‘1’, ‘#’, ‘\$’, ‘\*’
    - Cadeira de caracteres (**String**) = “Mônica Souza”, “20/03/2016”, “1234”, “mauricio.palma@ufscar.br”, “2+3”
- **Lógicos**: verdadeiro ou falso (True / False)

## Constante

É um valor fixo que **não se modifica** durante a execução do programa

## Variáveis

Representa informações cujo valores podem ser modificados ao longo do tempo

- **Nome ou Identificador:** nome dado à variável para possibilitar sua manipulação
- **Tipo de dado:** tipo dos dados que a variável vai armazenar
- **Conteúdo:** valor atual da variável

**Obs.:** Embora uma variável possa assumir diferentes valores, ela só pode armazenar um valor a cada instante

- Calcular a área de uma circunferência:

$$\text{Area} = \text{pi} * r$$

- **pi** tem valor **constante**, ou seja, independente de qual seja a circunferência (vale para todas as vezes em que a área é calculada)
- **r** e **Area** são **variáveis**, ou seja, a cada execução do algoritmo podemos ter um raio ( **r** ) distinto, gerando um valor diferente para a área (**Area**)



- Para colocar dados nas variáveis podemos:
  - Ler um valor do teclado e armazená-lo na variável fazendo uma operação de entrada de dados - **comando de leitura**
  - Atribuir um valor para a variável usando um **comando de atribuição**
  - Para mostrar o valor da variável fazemos uma operação de saída de dados, por exemplo mostrando o valor na tela - **comando de escrita**

1. Faça o algoritmo de um programa que, dado quatro números inteiros, calcule a soma desses números.
2. Faça o algoritmo de um programa que, dado três notas, calcule a média aritmética entre elas.
3. Faça o algoritmo de um programa que, dado o salário de um funcionário e o percentual de aumento, calcule o valor do aumento e o novo salário.
4. Faça o algoritmo de um programa que calcule a área de um triângulo.

# Python

Variáveis

Objetos

Atribuição

- Python é um exemplo de **linguagem de programação de alto nível**.
- O computador só consegue executar programas escritos em **linguagens de baixo nível** (“linguagens de máquina” ou “linguagens assembly”).
- Programas escritos em linguagens de alto nível precisam ser processados antes que possam rodar.

- 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.



- **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**.

- Existem duas maneiras de usar o interpretador python
  - **linha de comando** (“shell mode”)
  - modo de **script** (“program mode”)
- **Linha de comando**: você digita comandos em Python e o interpretador mostra o resultado
  - Só digitar “python” no terminal, ou abrir o python na busca do windows

```
$ python3.5
Python 3.5.2 (default, Nov 12 2018, 13:43:14)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
```

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

programa1.py

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

Interpretador

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



- 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.

```
print("Ola turma de Int. a Comp.")  
print("Vamos programar em Python")
```

```
print("Ola turma de Int. a Comp.") print("Vamos programar em Python")
```

A grey curved arrow points from the second code block to the explanatory text.

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


```
print("Ola turma de Int. a Comp.")  
print("Vamos programar em Python")
```



```
print("Ola turma de Int. a Comp.") print("Vamos programar em Python")
```



```
print("Ola turma de Int. a Comp."); print("Vamos programar em Python")
```



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.

- 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
```

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

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

**Variáveis**

**Memória/Objetos**

altura



10

largura



3

a



29

- Pode conter letras, dígitos (números) e o \_
- Não pode começar por números
- Não podem conter espaços em branco e outros caracteres como +, \*, /, ), @, etc
- Não pode ter acentuação
- Case sensitive (diferencia maiúsculas de minúsculas)
- Palavras reservadas da linguagem não podem ser usadas



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		

### Identificadores válidos

A123, SomaTotal, Nota\_final, ALFA, x

### Identificadores Inválidos

123a, +total , X-Y, Nota Final, (x)

# 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 a ele, mais de um nome ou nenhum nome.
- 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
```

- 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
```

- Alguns dos tipos básicos de objetos do Python são:
  - **int**: Corresponde aos números inteiros. Ex: 10, -24.
  - **float**: Corresponde aos números racionais. Ex.: 2.41, 3.141592
  - **str** ou **string**: Corresponde a textos. Ex: “Ola turma”. Um literal do tipo string deve estar entre aspas simples ou aspas duplas.
  - **bool**: falso (False) e verdadeiro (True).

- Uma variável em Python possui o tipo correspondente ao objeto que ela está associada **naquele instante**
- Python não possui tipagem forte.
  - 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 linguagens de tipagem forte, cria-se variáveis de tipos específicos e elas só podem armazenar valores daquele tipo para o qual foram criadas. Exemplos de linguagem com tipagem forte: C/C++, Java, Fortran.

```
>>> 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)
```

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)
```

# Obrigado!