

# Design de Software

Aula 4 – Python (loops while)

2016 – Engenharia

Fábio Ayres <[fabioja@insper.edu.br](mailto:fabioja@insper.edu.br)>

Raul Ikeda <[rauligs@insper.edu.br](mailto:rauligs@insper.edu.br)>

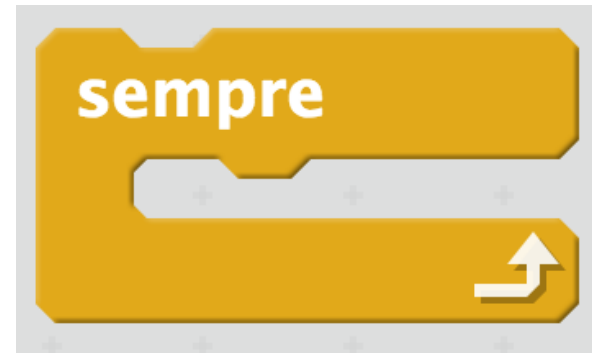
# Objetivos de Aprendizado

- Entender como e por que aplicar repetições do tipo *while*
- Resolver problemas que combinem testes condicionais e repetições
- Praticar leitura e análise de código

# Loops ou laços: *while*

- Um loop extremamente simples

```
1 while True:  
2     print("A")  
3
```



# While – mais um exemplo

- Um exemplo um pouco mais complexo.  
O que ele faz?

```
1 valor = 1
2 while valor!=0:
3     valor = int(input("Digite o valor"))
4     print("Você digitou %d"%valor)
5
```

# While – mais um exemplo

- Qual o resultado esperado?

```
1 i = 0
2 while i < 10:
3     print(i, end="")
4     i += 1
5
```

Evita pular para a próxima linha após o if

# Break e continue

```
1 while True:
2     valor = int(input("Digite um valor positivo (ou zero para terminar): "))
3
4     # Se o valor for negativo, volte para o início do LAÇO 'while'
5     if valor < 0:
6         print("Eu não gosto de números negativos, vamos tentar de novo.")
7         continue
8
9     # Se o valor for zero, pule para fora do LAÇO 'while'
10    if valor == 0:
11        print("Ok, parando...")
12        break
13
14    # Se chegou até aqui, imprima o valor
15    print("Você digitou {0}".format(valor))
16
17 print("Até mais!")
```

# Exercícios

1. Escreva um programa que imprima todos os números inteiros de -5 a 5.
2. Escreva um programa que peça um número  $n$  para o usuário e imprima a contagem regressiva de  $n$  até 0.
3. Escreva um programa que imprima todos os números de 0 a 50 que são múltiplos de 3.
4. Escreva um programa que imprima todos os números de 0 a 1000 que sejam múltiplos de 2, 3 ou 7.

# Exercícios

5. Escreva um programa que peça um número  $n$  para o usuário e diga se o número é primo ou não.
6. Escreva um programa que lê uma sequência de números positivos e imprime a soma deles. O programa deve parar de pedir números quando um número negativo for digitado.



# Exemplos mais complexos

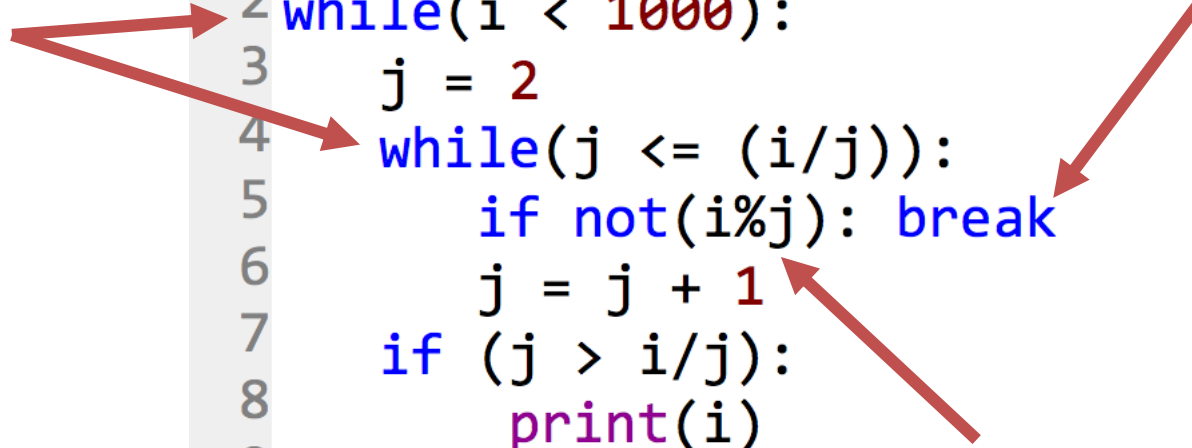
O que o código abaixo faz?

Loops aninhados?

```
1 i = 2
2 while(i < 1000):
3     j = 2
4     while(j <= (i/j)):
5         if not(i%j): break
6         j = j + 1
7     if (j > i/j):
8         print(i)
9     i += 1
10
```

break em código?

Negando um inteiro num if?



# Teste de Mesa

- Desenvolver Exemplo de Teste de Mesa

## Exercício 1 – `guess_number.py`

Uma habilidade muito importante para um programador é saber compreender rapidamente programas já existentes.

Esta é a realidade em muitos projetos.

Execute o programa `guess_number.py` e o analise

# Exercício 1 - Guess\_number.py – ler e depurar

```
1  # -*- coding: utf-8 -*-
2  from random import randint
3
4  print("""
5      Como vai a sua PARANORMALIDADE?
6      Vamos ver quantos chutes você leva
7      para acertar um número de 1 a 10
8
9      """)
10 number = randint(1,10)
11 guess = -1  #numero a adivinhar
12 steps = 0   #chutes
13
14 while number != guess:
15     guess = int(input("Escolha um numero de 1 a 10 \n"))
16
17     if guess < number:
18         print("Muito baixo")
19     elif guess > number: # elif means else if
20         print ("Muito alto")
21
22 if steps == 1:
23     print("Parabens! Voce é um médium")
24 elif steps > 1 and steps < 4:
25     print ("Você tem potencial mas precisa praticar")
26 elif steps > 4:
27     print ("Desculpe, você é um muggle")
28
```

# guess\_number.py – Atividades

1. O que faz a função randint?
2. Como você determina se o número gerado por randint inclui 0 e 10 ou não?
3. O programa funciona? Como ele é diferente de um programa que funciona? O que falta para que ele funcione?
4. Como você o consertaria? Conserte
2. Como você faria para dar ao usuário a opção de jogar mais um round, fazendo o sorteio de um novo número?

# Estutura de dados

- **Estruturas de dados** são formas de armazenar dados de forma eficiente no computador
- Armazenar dados numa sequencia é uma dessas formas e o Python possui seis tipos nativos (built-in types).
- Listas (**Lists**) são uma das formas mais comuns de se organizar dados.

# Criação de Listas

Para criar uma lista, basta colocar os seus dados entre colchetes e separados por vírgula. Exemplos:

```
# Série de Fibonacci
lista1 = [0,1,1,2,3,5,8,13,21,34,55,89,144,233,377,610,987]

# Planetas do Sistema Solar
lista2 = ["Mercúrio","Vênus","Terra","Marte","Júpiter","Saturno","Urano","Netuno"]

# Números de Bernoulli
lista3 = [1,-1/2,1/6,0,-1/30,0,1/42,0,-1/30,0,5/66]

# Cores do Arco-íris
lista4 = [0o77600000, 0xF7941D, [0.17, 1.00, 0.50], "verde", 255, "anil", 0x662D91]
```

# Acesando a Lista

Da mesma forma que String usa o índice, podemos fazer o *slice* da forma:

**Lista[início, fim, passo]**

Exemplo:

```
lista1 = [0,1,1,2,3,5,8,13,21,34,55,89,144,233,377,610,987]
lista2 = ["Mercúrio","Vênus","Terra","Marte","Júpiter","Saturno","Urano","Netuno"]
lista3 = [1,-1/2,1/6,0,-1/30,0,1/42,0,-1/30,0,5/66]
lista4 = [0o77600000, 0xF7941D, [0.17, 1.00, 0.50], "verde", 255, "anil", 0x662D91]
```

```
>>> print(lista1[2:4])
[1, 2]
>>> print(lista2[-1])
Netuno
>>> print(lista3[::2])
[1, 0.1666, -0.0333, 0.0238, -0.0333, 0.0757]
```



# Atualizando Listas

```
>>> lista = ["A","B","C"]  
>>> lista[1]="b"  
>>> print(lista)
```



['A', 'b', 'C']

```
>>> lista = ["A","B","C"]  
>>> del lista[1]  
>>> print(lista)
```



['A', 'C']

```
>>> lista = ["A","B","C"]  
>>> lista.append("D")  
>>> print(lista)
```



['A', 'B', 'C', 'D']

OU: >>> lista += ["D"]

# Soma e Multiplicação com Listas

- $p = [\text{None}] * 100$
- $q = [1] + [2, 3]$

# Métodos Úteis

**list.index(x)**

Retorna o índice do primeiro item cujo valor é x.

**list.count(x)**

Retorna o número de vezes que x aparece na lista.

**list.insert(i, x)**

Insere o item x na posição i da lista.

**list.remove(x)**

Remove o elemento x da lista

Mais detalhes das estruturas de dados em:

<https://docs.python.org/3/tutorial/datastructures.html>

# Resolva o problema com Listas

- Crie um programa que pergunta o número do mês e imprime o nome do mês.
- Crie um programa que pergunta o nome do mês e imprime o número do mês.

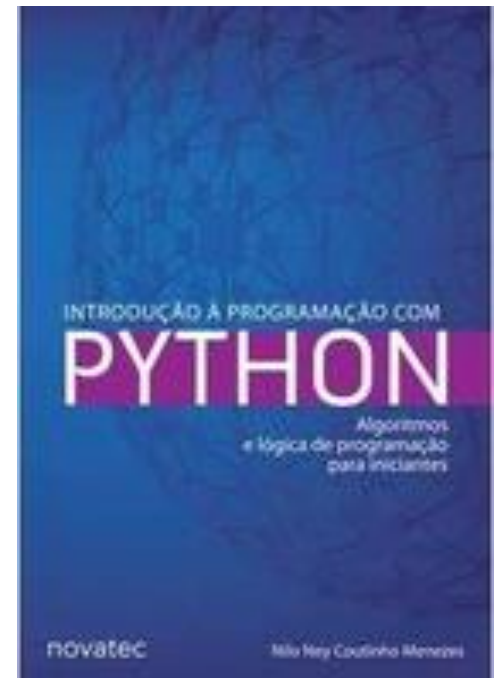
# Resolva o problema com Listas

- Crie um programa que pergunte palavras ao usuário e preencha um lista. O programa deve parar com a palavra "fim". Ao final somente as palavras em que a primeira letra seja "a" é que são impressas.

# Para a próxima aula

- Repetições (Cap. 5)
- Listas (Cap. 6)

Capítulos 5 e 6 do livro  
**Introdução à Programação com Python**,  
de Nilo Ney Menezes.  
(disponível na biblioteca)



# Insper

[www.insper.edu.br](http://www.insper.edu.br)

# Fun: Shooting Yourself in the Foot

Python: You create a gun module, a gun class, a foot module and a foot class. After realizing you can't point the gun at the foot, you pass a reference to the gun to a foot object. After the foot is blown up, the gun object remains alive for eternity, ready to shoot all future feet that may happen to appear.

Java: You find that Microsoft and Sun have released incompatible class libraries both implementing Gun objects. You then find that although there are plenty of feet objects implemented in the past in many other languages, you cannot get access to one. But seeing as Java is so cool, you don't care and go around shooting anything else you can find.

—Mark Hammond