



Nesta aula veremos:

- 1- Identação e os comandos if, else, print, input, % e while.
- 2- Import.
- 3- Def, for, range, len, // e [].

- 1- Identação e os comandos if, else, print, input, % e while.

```
2- ano = 0
3- print('Digite o ano (9999 para sair)')
4-
5- while ano != 9999:
6-
7-     ano = int(input('Ano: '))
8-
9-     if (ano%4==0 and ano%100!=0) or (ano%400==0):
10-         print('Ano Bissexto')
11-     else:
12-         print('Ano Nao e bissexto')
13-
14- print('Programa encerrado!')
```

2- Comando import, existem muitos pacotes para serem importados em sua aplicação:

matemática	<ul style="list-style-type: none">mathstatisticsrandom	Processamento de texto	<ul style="list-style-type: none">stringrereadline
Sistemas de arquivos	<ul style="list-style-type: none">fileinputgzipzipfile	Tipos de dados	<ul style="list-style-type: none">collectionsarraydatetimecalendar
Formatos de arquivos	<ul style="list-style-type: none">configparsercsv	Sistemas operacionais	<ul style="list-style-type: none">osplatform

O **NumPy** é um dos pacotes mais populares do Python. Isso porque ele é rápido e oferece muitas funções e operações que podem contribuir para a execução simples e fácil de cálculos numéricos pelos desenvolvedores.

O **Pillow** é o amigável fork da PIL – Python Imaging Library (Biblioteca de Imagens do Python, em português). Para alguns especialistas, é como se o Pillow fosse uma versão aprimorada do PIL

Matplotlib é um pacote muito utilizado para a criação de gráficos que possam ser empregados na melhor visualização de dados.

O **OpenCV** (Open Source Computer Vision Library) é um pacote desenvolvido pela Intel e incorporado a muitas linguagens de programação, como é o caso do Python.

O **Delorean** é um pacote que fornece melhorias na linguagem para manipular o tempo, isto é, o DateTime (data e hora).

O **SciPy** é um outro pacote básico da linguagem Python. Sua base é o NumPy, uma vez que ele precisa lidar com grandes quantidades de números. Por isso, é muito utilizado na computação científica e técnica.

O pacote **Pandas** possui como característica principal a sua capacidade de realizar traduções de operações complexas com dados em um ou dois comandos. Por isso mesmo ele é ideal para a ciência de dados.

O **Requests** tem como principal finalidade tornar as solicitações HTTP mais simples, responsivas e amigáveis. Com ele, é possível personalizar, controlar, autorizar e configurar solicitações HTTP de forma muito fácil.

O **Fire** é um pacote capaz de gerar automaticamente uma interface de linha de comando (CLI) de qualquer objeto Python. O **Google**, por exemplo, utiliza-o para criar uma linha de comando e diferentes ferramentas de gerenciamento de experiências.

O Natural Language ToolKit, ou simplesmente **NLTK** (Kit de Ferramentas de Linguagem Natural, em português) é um pacote que desenvolve algoritmos capazes de compreender a linguagem humana.

O **Arrow** é um pacote focado em datas e horas. Isso quer dizer que ele oferece uma abordagem mais amigável para criar, manipular, formatar, remover e converter datas e horas com menos importações e menos linhas de código.

O **FlashText** tem como finalidade pesquisar e substituir palavras em documento de forma fácil. Para fazer isso, ele cria uma string atualizada e, ao realizar a pesquisa, recorre a uma lista de palavras-chave para realizar a substituição.

Por fim, um outro importante pacote Python é o **PyGame**. Como o próprio nome sugere, ele consiste em um conjunto de funções e classes destinadas à criação de videogames e de algumas outras aplicações multimídia. Para isso, ele oferece módulos compostos por elementos gráficos e sonoros.

Importando um pacote:

```
import math
print(math.sqrt(25))
```

Importando apenas um modulo do pacote:

```
from math import sqrt
print(sqrt(25))
```

```
import random

numero = random.randrange(1,100)
wtenta = 0

while wtenta < 10:
    seunum = input("Digite seu palpite : ")
    wtenta = wtenta + 1
    # wtenta += 1

    if seunum == numero:
        print("Parabens voce acertou em ", wtenta, "tentativas.")
        wtenta = 11
    else:
        if seunum < numero:
            print("Tente um valor maior.")
        else:
            print("Tente um valor menor.")

if wtenta == 10:
    print("Voce utrapassou o numero de tentativas, tente novamente!")
```

3- Def, for, range, len, // e [].

Funções são definidas e chamadas por: função().

Funções podem receber argumentos como parâmetro.

```
def verifica(saque):
    lista = []
    notas = 0
    valor = saque
    qtd = saque // 100
    lista += ["          "]
    if (qtd > 0):
        nota_100 = 'Quantidade de notas de 100 : ' + str(qtd)
        lista.append(nota_100)
        notas = notas + qtd
    saque=saque % 100
    qtd = saque // 50

    if (qtd > 0):
        nota_50 = 'Quantidade de notas de 50 : ' + str(qtd)
        lista.append(nota_50)
        notas = notas + qtd
```

```
saque=saque % 50
qtd = saque // 20

if (qtd > 0):
    nota_20 = 'Quantidade de notas de 20 : ' + str( qtd)
    lista.append(nota_20)
    notas = notas + qtd
saque=saque % 20
qtd = saque // 10

if (qtd > 0):
    nota_10 = 'Quantidade de notas de 10 : ' + str(qtd)
    lista.append(nota_10)
    notas = notas +(qtd)
saque=saque % 10
qtd = saque // 5

if (qtd > 0):
    nota_5 = 'Quantidade de notas de 5 : ' + str(qtd)
    lista.append(nota_5)
    notas = notas +(qtd)
saque=saque % 5
qtd = saque // 2

if (qtd > 0):
    nota_2 = 'Quantidade de notas de 2 : ' + str(qtd)
    lista.append(nota_2)
    notas = notas + qtd
lista += ["          "]
lista += ["_____"]
lista += ["Seu saque no valor de " +str(valor) + " tem " +str(notas) + " notas!!!"]
lista += ["          "]

for i in range(len(lista)):
    print(lista[i])

return lista[0 : 8]

def main():
    global saque
    saque = int(input('Digite o valor do saque : '))

    if ( 10 <= saque <= 999999):
```

```
verifica(saque)
else:
    print('Valor nao pode ser sacado')
```

```
main()
```