PythonLab

Prof. Dr. Álvaro Campos Ferreira

idp

Comentários

Variáveis

Tipos das variáveis

Tipos de variáveis

Para cada tipo de variável, as funções tem comportamento diferente.

- Texto (str)
- Número inteiro (int)
- Número real (float)
- Booleano (bool), True ou False

Tipos de variáveis

Ao declarar uma variável, o Python determina seu tipo dinamicamente, uma variável pode mudar de tipo apenas declarando-a novamente:

```
idade = “Trinta e dois” # tipo texto
idade = 32 # tipo inteiro
```

Listas

Dicionários

Controle de fluxo

Laços for ou “para cada”

Com **for**, o bloco irá executar para cada elemento do objeto.

```
for cliente in clientes:
    print(cliente)
```

idp

Laços while ou “enquanto”

O **while** executará o bloco “enquanto” a condição a direita dele resultar em verdadeiro, ou booleano True.

```
while nome == “Álvaro”:
    print(“Olá, professor!”)
```

idp

Blocos if... else ou “se... senão”

O **if** ou “se”, executa o bloco se a condição for satisfeita.

```
if nome == “Álvaro”:
    print(“Bom dia, professor”)
else:
    print(“Bom dia, querido colega”)
```

idp

Funções

Funções

Define-se funções com a palavra-chave **de**
da palavra definir:

```
def somar_numeros(a,b):
    return a + b
```

Documentando funções

A forma padrão de documentar uma função chama-se Docstring e nada mais é que um bloco de comentário.

```
def somar_numeros(a,b):
 """ Soma dois números.
 """
        return a + b
```

Retornando valores

Uma função pode retornar um ou mais valores após a execução.

```
def somar_numeros(a,b):
 """ Soma dois números.
 """

     return a + b
```

Módulos

Módulos de funções

É comum e boa prática gerar arquivos com funções relacionadas para que possam ser importados em outros scripts.

O funcionamento é semelhante a importar bibliotecas.

Módulos de funções

Nesses casos, é útil saber quando um módulo está sendo importado e quando está sendo executado como script e definir comportamentos diferentes dependendo do caso.

Módulos de funções

```
def main():
    print(“Meu módulo é o máximo.”)
if __name__ == ‘__main__’:
    main()
```

Bibliotecas

Vetores e Matrizes

Vetores e Matrizes

Arrays são construídos com listas:

```
import numpy as np
minha_lista = [1, 4, 7]
meu_vetor = np.array(minha_lista)
```

$$b = \begin{Bmatrix} 1 \\ 4 \\ 7 \end{Bmatrix} \qquad x = \begin{Bmatrix} x \\ y \\ z \end{Bmatrix}$$

idp

Vetores e Matrizes

Matrizes são construídas com listas de listas:

```
import numpy as np
minha_lista = [[1, 2, 3], [4,5,6], [7,8,9]]
minha_matriz = np.array(minha_lista)
```

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

idp

Vetores e Matrizes

O Numpy tem funções para construir matrizes especiais, como a matriz identidade.

```
import numpy as np
meusZeros = np.zeros(10,10)
meusUns = np.ones(10,10)
minhaIdentidade = np.eye(10,10)
```

Manipulação gráfica

Manipulação gráfica

É possível importar imagens diretamente como arrays utilizando a biblioteca skimages.

```
from skimage import io
foto = io.imread('eixos.png')
foto.shape
```

Manipulação gráfica

Os dados são armazenados em arrays em que cada elemento possui um canal de cor para o pixel.

```
import matplotlib.pyplot as plt
plt.imshow(foto)
plt.imshow(foto[:,:,0])
plt.imshow(foto + 100)
plt.imshow(foto * 100)
```

Programar é especificar

Pensamento computacional

- Decomposição
- Reconhecimento de padrões
- Abstração
- Algoritmo

Especificação

Ao especificar, é necessário decompor e abstrair a sua ideia, reconhecer os padrões para por fim poder determinar o algoritmo e de fato desenvolver a aplicação.

Especificação

Isso pode ser realizado formalmente através de um documento de especificação de requisitos.

Para problemas pequenos, a especificação é simples e não precisa ser formalizada, mas sempre existe.

Especificação e planejamento

Especificação e planejamento

A especificação vai permitir planejar o curso do projeto pois será possível estimar o tempo para a implementação de cada funcionalidade especificada.

Metodologias de desenvolvimento

Metodologia Cascata (Waterfall)

Metodologia de gerenciamento de projetos em que os requisitos dos clientes e partes interessadas são coletados no início do projeto. Então um plano **sequencial** é criado para acomodar todos os requisitos.

Metodologia Ágil (Agile)

Metodologia de gerenciamento de projetos que procura o desenvolvimento e entrega contínua do produto para o cliente, encorajando desenvolvimento e testes paralelos a partir de times pequenos trabalhando em funcionalidades diferentes.

Metodologia Ágil (Agile)

Valores da metodologia ágil:

- Indivíduos e interações sobre processos e ferramentas
- Software funcionando sobre documentação completa
- Colaboração com o cliente sobre seguir um plano

Ciclo de vida

Cascata:

- Ciclo de vida longo
- Poucos ajustes

Ágil:

- Ciclo de vida curto
- Adaptável

Estágios de monitoramento Ágil

1) Requerimentos
2) Design
3) Desenvolvimento
4) Integração e teste
5) Implementação e deployment
6) Review

Tipos de monitoramento ágil

Scrum

- Projetos grandes
- Sprints

Kanban

- Projetos rápidos
- Fluxo de trabalho

Produção, Homologação e Desenvolvimento

INSTITUTO BRASILEIRO DE ENSINO,
DESENVOLVIMENTO E PESQUISA