FHO | FUNDAÇÃO HERMÍNIO OMETTO

Programação de Computadores II

Apresentação da Disciplina

Plano de Ensino

Cronograma

Revisão Programação de Computadores I

Porque a engenharia deve conhecer programação?

- Desenvolvimento de Raciocínio Lógico;
- Representação de Processos a nível Abstrato;
- Interpretação e representação da informação do mundo real;
- Capacidade de solucionar problemas;
- Habilidade de Interpretação de diferentes linguagens.

Conteúdo Programado

- Revisão de Programação de Computadores II;
- Revisão de Programação de Computadores II;
- Dicionário;
- Matriz;
- Módulo Numpy;
- Função;
- Módulo Matplotlib;
- Exercícios de Fixação

Manipulando Tupla

- Para contar quantos elementos tem em uma tupla precisamos utilizar a função count(), e passar a posição como parâmetro.

```
'''

AMBIENTE DE PROGRAMAÇÃO

'''

tupla = (1,"ola",1,1,5)

print(tupla.count(1))
```

- Resutado será: 3
- pois o comando irá contas quantos elementos tem na posição desejada. no caso a palavra “ola".

Manipulando Tupla

- Para saber qual o índice da tupla em que esta a informação basta utilizar a função index() e o parâmetro deve ser a informação.

```
'''

AMBIENTE DE PROGRAMAÇÃO

'''

tupla = (1,"ola",1,1,5)

print(tupla.index("ola"))
```

- Resultado será: 1
- repare que o parâmetro dentro da função foi a palavra “ola” e o retorno foi o número do índice.

Manipulando Tupla

- Para verificar a existência de determinado elementos em uma tupla podemos utilizar o comando in.

```
'''

AMBIENTE DE PROGRAMAÇÃO

'''

tupla = (1,"ola",1,1,5)

print("ola" in tupla)
```

- Resultado será: True
- Caso o elemento a ser procurado existir dentro da tupla será retornado true caso contrário será retornado false.

Manipulando String

- Em python uma string é considerada uma cadeia de caracteres. Esses caracteres mesmo formando uma palavra ou uma frase podem ser selecionados individualmente, assim como podemos fazer com uma lista.

Python
3. *nome_palavra* [*índice*]

Manipulando String

- Assim como as variáveis que recebem números, podemos criar variáveis com palavras, onde seu tipo é **string**. O comando imput retorna um valor **string**, portanto sempre que vamos trabalhar com variáveis numéricas precisamos converter elas para números.
- A partir de agora iremos detalhar como manipular as **strings**, e como tratar uma variável string em números.

Comando Importante

- Quando temos uma frase ou uma palavra dentro de uma variável em formato string podemos fazer o fatiamento dessa frase ou dessa palavra.
    - frase= **(“Programação de Computadores II”)**
    - frase[3] = exibe apenas o índice 3;
    - frase[3:14] = vai exibir do índice 3 até o índice 13, o ultimo valor não é incluso na exibição;
    - frase[9:15:2] = vai apresentar de 9 à 15 pulando de 2 em 2 caracteres;
    - frase[:15] = vai apresentar do começo da string (caracter 0).
    - frase[5:] = vai apresentar do caracter 5 até o final da string;
    - frase[9::3] = vai apresentar do caracter 9 até o final pulando de 3 em 3 caracteres.

Comando Importante

- Existem comando que podemos utilizar para analizar a string.
  - **Comando len** - determina o comprimento da estrutura de caracteres.
    - print(len(frase))
  - **count: frase.count(“o”)** - vai contar quantas vezes a letra o aparece.
  - **count(“o”,0,17)** - conta quantas vezes a letra o pararece no intervalo determinado.
  - **replace: frase.replace**(‘Programação’,’Thiago') reposiciona as palavras mais não altera o texto inicial.

Comando Importante

- **Title : frase.title()** - coloca a primeira letra de cada frase em maiúsculo.
- **Captalize : frase.captalize()** - coloca a primeira letra da frase em maiúsculo.
- **Upper : frase.upper()** - coloca todos as letras da frase em maiúsculo.
- **Lower : frase.lower()** - coloca todas as letras da frase em minúsculo.
-

Comando Importante

- **Strip** : Remove espaços desnecessários.
  - **rstrip** - remove os espaços extras que estão a direita (nome.rstrip);
  - **lstrip** - remove espaços extras que estão a esquerda (nome.lstrip).
- Quando queremos concatenar frases ou palavras utilizamos o sinal de adição (+).

Comando Importante

- Qualquer frase pode ser decomposta em sub-palavras e assim gerar várias listas. Para isso utilizamos o comando split.

    - frase.split()

Conversão de string para número

- Quando utilizamos o comando imput, automaticamente ele retorna uma string, independente do que foi solicitado ou o usuários digitado. Portanto quando queremos trabalhar com números precisamos converter essa string em números inteiros ou reias, com os seguintes comandos.

  - **variavel = int(variavel)** - para converter em números inteiros.
  - **variavel = float(variavel)** - para converter em números reais.