Lecture 01

1. Criar um código que exemplifique a impressão de uma mensagem simples na tela.

2. Criar um código que exemplifique a impressão do type() para variáveis de diversos tipos.

3. Criar um código que exemplifique o uso dos diversos tipos de aspas do Python em strings.

4. Criar um código que exemplifique a criação de função das seguintes maneiras:

    1. Sem parâmetros;
    2. Com parâmetros;
    3. Com retorno;

5. Criar um código que exemplifique a utilização de operadores matemáticos: +=, *=, -= …

6. Criar um código que exemplifique a utilização de operadores lógicos e dos valores True e False, diretamente no print().

7. Criar um código que exemplifique a conversão de tipo de uma variável e a impressão do type() dessa variável.

8. Criar um código que exemplifique a entrada de dados para uma variável, nas seguintes formas:

    - Entrada simples de dados;
    - Conversão de tipo na entrada dos dados;

Observação: *Todos os exemplos devem conter tratamento dos dados que estão entrando.*

Lecture 02

1. Criar uma string contendo um “Lorem Ipsum”, e imprima essa string, bem como o tamanho desta mesma string.

2. Criar um código que exemplifique a utilização de caracteres especiais como estes: \n, \t, \\, \', \", \r, \b, \f, \v e \0.

3. Criar um código que exemplifique a utilização de caracteres na impressão de uma string a partir de um início e fim dessa mesma string utilizando slicing [início:fim].

4. Criar um código que exemplifique a utilização de funções específicas para strings, como:

   - find(), endswith(), replace(), count(), capitalize(), upper(), lower().

5. Criar um código que exemplifique a utilização do if…, elif e else.

Lecture 03

Lista

1. Criar um exemplo de uma lista de tipos diversos.

2. Criar um código que exemplifique a alteração de algum dado dessa lista mediante o seu índice.

3. Criar um código que exemplifique a impressão de uma lista usando o print().

4. Criar um código que mostre o tamanho da lista de dados anteriormente criada.

5. Criar um código que tente ordenar a lista de dados diversos anteriormente criada com sort().

6. Criar uma lista de números.

7. Criar um código que exemplifique a utilização dos métodos abaixo na lista de números anteriormente criada:

    - sort(), sort(reverse=True), reverse(), append(), insert(), remove(), pop().

Tupla

1. Criar uma tupla de números.

2. Criar um código para imprimir a tupla anteriormente criada.

3. Criar um código para imprimir o type() da tupla criada.

4. Criar um código que exemplifique a impressão de um dado da tupla utilizando o seu índice.

5. Criar um código que exemplifique a impressão de dados de uma tupla mediante um índice inicial e final de dados.

6. Criar um código que exemplifique a execução das diversas funções referentes ao uso de tupla:

   - index(), count(), append(), sort(), sort(reverse=True),
   - e verificar se funcionam:
     reverse(), insert(), remove(), pop().

7. Criar um código que exemplifique a utilização do if… else para verificar se uma string é um palíndromo, utilizando as letras da palavra "RADAR" (e outras palavras) em uma lista, com o uso das funções copy() e reverse().

Lecture 04

Dictionary

1. Criar um dicionário com dados de tipos diversos, inclusive contendo listas e tuplas, com chaves que podem conter, como dados, também dicionários aninhados.

2. Criar um código que exemplifique a impressão desse dicionário de dados.

3. Criar um código que exemplifique a impressão, por meio de sua chave, de algum dado do dicionário de dados anteriormente criado, seja esse dado não aninhado ou aninhado.

4. Criar um código que exemplifique a alteração de algum dado do dicionário anteriormente criado.

5. Criar um código que exemplifique a impressão dos seguintes dados do dicionário anteriormente criado: chaves, valores e items.

6. Criar um código que exemplifique a impressão de um dado do dicionário de dados anteriormente criado, usando a chave desse dado e a função get().

7. Criar um código que exemplifique a alteração de dados do dicionário utilizando update().

Set

1. Criar um set de números.

2. Criar um código que exemplifique a execução de operações no set criado anteriormente, com os seguintes métodos:

   - add(), remove(), pop() e clear().

3. Criar dois sets de números e executar as seguintes funções:

   - union() e intersection().

4. Criar um set com um número inteiro e, em seguida, inserir o mesmo número, porém em ponto flutuante, e imprimi-lo.

Lecture 05

Uso do looping em Python: **While** e **For**.

While

1. Criar um código que exemplifique a execução de um looping simples.

2. Criar um código que exemplifique a execução de um looping infinito com break.

3. Criar uma lista de números.

4. Criar um código que exemplifique a impressão de cada número mediante seu índice.

5. Criar uma lista de nomes de pessoas.

6. Criar um código que exemplifique a impressão de cada nome mediante seu índice, nas seguintes formas:

   - De forma ascendente;
   - De forma descendente;

7. Criar uma tupla de números.

8. Criar um código que exemplifique a impressão de cada número pelo seu índice.

For

1. Criar uma tupla de números.

2. Criar um código que exemplifique a impressão dos números da tupla anteriormente criada através do seu índice.

3. Criar uma tupla de nomes de pessoas.

4. Criar um código que exemplifique a impressão de cada nome através do seu índice da seguinte forma:

   - De forma ascendente;
   - De forma descendente;

5. Criar uma string com um nome.

6. Criar um código que exemplifique que, mediante iteração sobre cada caractere, um caractere específico seja alterado utilizando um condicional.

7. Criar um código que exemplifique a criação de range(limite) de números e a iteração sobre esse mesmo range, fazendo o seguinte:

   - Apenas imprimindo;
   - Imprimindo mediante condição;

8. Criar um código que exemplifique a criação de range(início, limite) de números e a iteração sobre esse mesmo range, fazendo o seguinte:

   - Apenas imprimindo;
   - Imprimindo mediante condição;

9. Criar um código que exemplifique a criação de range(início, limite, passos) de números e a iteração sobre esse mesmo range, fazendo o seguinte:

   - Apenas imprimindo;
   - Imprimindo mediante condição;

10. Criar um range(início, limite, passos) e utilizar o WHILE para iterar sobre o mesmo fazendo o seguinte:

    - Apenas imprimindo;
    - Imprimindo mediante condição;

Lecture 06

Uso de funções.

1. Criar um código que exemplifique a criação de uma função com uma mensagem de saudação, fazendo o seguinte:

    - Impressão simples;
    - Impressão utilizando o end=;

2. Criar um código que exemplifique a criação de uma função com valor pré-definido e mensagem de saudação, fazendo o seguinte:

    - Impressão simples;
    - Impressão utilizando o end=;

3. Criar um código que exemplifique uma função que mostre o tamanho de uma coleção a ser passada.

4. Criar um código que exemplifique uma função que receba uma coleção e imprima seus itens separados por vírgula.

5. Criar um código que exemplifique uma função que realize recursão (ou seja, que chame a si mesma).

6. Criar um código que exemplifique uma função que realize o cálculo de um "fatorial" entre dois números (início e fim) no formato de soma.

7. Criar um código que exemplifique uma função que realize o cálculo de um "fatorial" entre dois números (início e fim) no formato de multiplicação.

8. Criar um código que exemplifique uma função que receba um valor em dólar e o converta para real.

9. Crie uma lista de nomes de pessoas.

10. Criar um código que exemplifique uma função que imprima esses nomes utilizando o conceito de recursão.

Lecture 07

Conceitos sobre o uso de arquivo em Python.

1. Criar um código que exemplifique a execução da leitura de um arquivo de texto utilizando os métodos read() e readline().

2. Criar um código que exemplifique a execução da escrita em um arquivo de texto utilizando buffer e codificação para esse mesmo arquivo.

3. Criar um código que exemplifique a execução da leitura de um arquivo de texto, utilizando buffer e codificação para esse mesmo arquivo.

4. Criar um código que exemplifique a abertura de um arquivo, exibindo seu conteúdo e os dados como codificação, tipo (usando type()) e outros metadados disponíveis.

5. Criar um código que exemplifique a abertura de um arquivo e o uso do read() diretamente dentro da função print().

6. Criar uma lista de nomes de pessoas.

7. Criar um código que exemplifique a inclusão desses nomes, um por um e separados por espaço, em um arquivo.

8. Criar um código que exemplifique a leitura dos N primeiros caracteres de um arquivo, com o valor de N informado diretamente no print().

9. Criar uma classe que contenha métodos para trabalhar com arquivos, contendo os seguintes métodos: escrita, leitura, verificação se uma determinada palavra existe, e exclusão do arquivo. Todos os métodos devem conter tratamento de exceção.

Lecture 08

Conceito sobre orientação a objetos - parte 01.

1. Criar uma classe, por exemplo, Person (Pessoa), que contenha o seguinte:

    - Atributo estático;
    - Construtores: sem parâmetros e com parâmetros;
    - Impressão do dado referente ao __getstate__ para esta classe;
    - Criação de um método estático usando o decorador @staticmethod;
    - Sobrescrição do método __str__ para fornecer uma representação legível do objeto;
    - Implementação da possibilidade de a classe ser utilizada dentro de uma declaração with.

Lecture 09

Conceito sobre orientação a objetos - parte 02.

1. Criar uma classe que exemplifique o uso de:
    - Atributos privados;
    - Métodos privados;

2. Criar um objeto da classe criada anteriormente e, após sua utilização, deletá-lo, demonstrando a tentativa de acesso a um item dessa classe após a deleção do objeto.

3. Criar uma classe, por exemplo, Person (Pessoa), com atributos privados e implementar os métodos getter e setter utilizando o decorador @property.

4. Criar um método de classe utilizando @classmethod para a classe criada anteriormente e demonstrar sua utilização.

Sobrecarga de Operadores

5. Criar uma classe Vector com os atributos x e y.

6. Sobrecargar os operadores matemáticos para permitir, por exemplo, a soma dos atributos x e a soma dos atributos y de dois objetos Vector.

7. Sobrescrever o método __str__ para exibir o vetor de forma legível.

8. Criar uma classe, por exemplo, Product (Produto), com os atributos name e price.

9. Sobrecargar o operador __gt__ para verificar se o preço de um objeto é maior ou menor que o preço de outro objeto.