XVI Semana de Sistemas de Informação da UNIVAS – Out/2022

Lista de Exercícios Mini Curso de Python

1. **[Introdução]** Utilizando a linguagem Python implemente um programa que exiba na tela do usuário o texto: ***Hello, World!!!***

2. **[Input/Output]** Crie um programa em Python capaz de solicitar ao usuário dados pessoais para cadastro. Utilizando a função *built-in* ***input()*** solicite o *nome*, *idade* e se possuir curso superior. Armazene estes dados em tipos adequados respectivamente: ***str***, ***int*** e ***boolean***. **Obs.:** será necessário fazer a conversão de tipos, pois o retorno de *input()* é sempre texto, ou seja, ***str***.

3. **[Input/Output]** Crie um programa em Python que solicite um valor qualquer ao usuário e exiba o tipo do dado fornecido que o usuário digitou no teclado. Para implementar esta solução sigas as instruções:

    - Ao capturar o valor ao usuário, utilize a função *built-in input()*, inserido na função *eval()*, para converter o tipo conforme o dado informado: ***eval(input("Digite um valor qualquer: "))***;
    - Para obter o tipo da variável utilize a função *built-in type()*, como: ***print(type(valor_usuario))***;

4. **[Modules]** Aproveitando o exemplo hello_world.py crie no mesmo diretório outro módulo: ***utils.py***. Neste novo módulo crie a seguinte variável: ***curso = "Introduction Python".***

    No módulo *hello_world.py* faça o *import* de *utils.py*, refatore a implementação da mensagem ***"Hello, World!!!"***, acrescentando a mensagem ***"Welcome to Introduction Python"***, utilizando pra isso o conteúdo da variável definida em utils.py.

5. **[Modules]** Utilize o exercício acima e implemente a execução principal (***main()***) em Python para o seu programa.

6. **[Biblioteca Padrão]** Implemente um programa em Python para calcular os dias de vida de um usuário. Solicite ao usuário o ano, mês e dia de seu aniversário, realize o cálculo e exiba o resultado.

7. **[IF/ELSE]** Crie um programa em Python capaz solicitar ao usuário um número e elevá-lo ao cubo. Após o cálculo, informe se o valor obtido é maior ou menor que 100. Para esta implementação considere:
    - Utilize a função *input* para armazenar o retorno de uma variável:
        - ***val = input("mensagem")*** *-> exige conversão para o tipo numérico.*
        - ***Obs.: val = int(val)*** *-> realiza a conversão para o tipo de dado inteiro.*
    - Utilize a função *math.pow* para ele o valor ao cubo:
        - Faz-se necessário importar a biblioteca: ***import math***
    - Utilize a estrutura de controle: ***if else*** para descobrir se é < ou > que 100:
        - ***Obs.:*** atente-se a correta indentação para cada estrutura de controle.
    - Para exibir o resultado ao usuário utilize ***f-Strings***, como no exemplo:
        - ***name = "Eric"***
        - ***age = 74***
        - ***print(f"Hello, {name}. You are {age}.")***

8. **[FOR]** Faça um programa que gere e escreve os números ímpares dos números entre 100 e 200.

9. **[WHILE]** Crie um programa que forneça as operações básicas de uma calculadora: adição, subtração, divisão e multiplicação. Ao iniciar, o programa deve questionar qual operação o usuário deve realizar e solicitar os dois valores que serão calculados. Deve-se oferecer a opção Sair para o usuário. Deve-se utilizar o comando While, para que a sua calculadora seja executada até que o usuário selecione a opção Sair. Após cada cálculo executado deve-se exibir o resultado para o usuário.

10. **[DEF]** Faça um programa para calcular o IMC de uma pessoa. O IMC é calculado dividindo o peso pela altura elevada ao quadrado. O cálculo do IMC deve estar numa função. Dê um nome intuitivo e amigável para a função. O peso e a altura da pessoa devem ser solicitados, convertidos para float e passados como parâmetros da função. Exiba o resultado para o usuário.

11. **[DEF]** Crie um programa para gravar dados em arquivo. Deve conter um módulo com duas funções:

    - init(): esta função deve solicitar ao usuário os seguintes dados: nome do arquivo, nome, cpf, endereço do usuário.
    - write_data(file_name, name, cpf, address): gravar todos os dados em um arquivo conforme nome definido pelo usuário. Confira exemplo de código a seguir.

    Obs.: Insira a verificação do valor __name__ para permitir sua execução pela linha de comando.

```python
#Criar arquivo e escrever
new_file = open('meu_arquivo.txt', 'w')
new_file.write('Linha 1\n')
new_file.write('Linha 2\n')
new_file.write('Linha 3\n')
new_file.close()
```

12. **[STR]** Nome na vertical: solicite o nome do usuário e imprima-o na vertical.

13. [STR] Solicite ao usuário o seu nome e sobrenome separados por espaço. Escreva um programa que divida o valor informado em duas variáveis. Exiba o nome e sobrenome informado com o seu valor alterado através do método *swapcase()* do objeto string.

14. **[TUPLE]** Faça um programa que contenha uma função capaz de receber uma tupla como parâmetro. Esta função deve varrer todos os elementos da tupla recebida e caso encontre algum dado do tipo string, informe ao usuário: "String encontrada: [valor]". Dicas:
    - Utilize a função *built-in type()* do Python.
    - Crie os valores das tuplas direto no código *(hard code)*.

15. **[LIST]** Faça um programa que leia uma lista contendo 10 caracteres, e diga quantas consoantes foram lidas.

16. [LIST] Faça um Programa que peça a idade e a altura de 5 pessoas, armazene cada informação no seu respectivo vetor. Imprima a idade e a altura na ordem inversa a ordem lida. Utilize o método *reverse()* da lista.

17. **[DICT]** Crie um dicionário e armazene nele os seus dados: nome, idade, telefone, endereço. Imprima todos os dados usando o padrão chave: valor.

18. [DICT] Crie um programa que, usando dicionário, crie uma agenda de tamanho fornecido inicialmente pelo usuário. Leia os dados de todos os contatos do usuário de forma que a agenda fique completa e por fim imprima todos os contatos. O dicionário que representa um usuário deve contem nome e telefone.

19. **[VENV]** Faça um programa Python capaz de validar endereços CEP, fazendo uso da API pública https://viacep.com.br. Utilize a biblioteca: https://pypi.org/project/requests/ para realizar as requisições HTTP à esta API. O usuário deve informar o valor do CEP e o programa informar se o CEP é válido ou não.

    Obs.: realize a seguinte instalação numa Virtual Env Python: ***pip install requests***.