INSTITUTO FEDERAL Espírito Santo Campus Serra	**Curso de Engenharia de Controle e Automação**			
	Componente Curricular: Programação Orientada a Objetos.			
	Professor: Hilário Tomaz Alves de Oliveira			
	Semestre: 2022.1	**Período:** 3°	**Turma:** Noite	**Data de Entrega:** 01/05/2022

Lista de Exercícios 2 – Linguagem de Programação Python (Aulas 6, 7 e 8)

Observações:

- A solução de cada questão que necessita de implementação de código deve estar contida em um arquivo de código na linguagem Python (extensão .py). Para padronizar utilize o seguinte padrão de nomenclatura.
  - l#NumeroListaq#NumeroQuestao.py
    - Nos quais:
      - #NumeroLista deve ser trocado pelo número da lista;
      - #NumeroQuestao deve ser trocado pelo número da questão.
  - **Exemplos:** l1q1.py, l3q4.py, l5q10.py, e assim sucessivamente
- As questões objetivas ou subjetivas podem ser resolvidas em um único arquivo texto contendo de forma explícita o número da questão e sua respectiva resposta.
- Ao final cada aluno deve enviar uma pasta compactada contendo todos os arquivos com suas soluções.

Questão 1. Analise cada um dos trechos de código a seguir e justifique, quando necessário, quais erros eles possuem.

a)
```
dia, temperatura = ('Monday', 87, 65)
```

b)
```
numeros = [1, 2, 3, 4, 5]
```

```
print(numeros[10])
```

c)
```
nome = 'amanda'

nome[0] = 'A'
```

d)
```
numeros = [1, 2, 3, 4, 5]

numeros[3.4] = 20
```

e)
```
tupla_alunos = ('Amanda', 'Blue', [98, 75, 87])

tupla_alunos[0] = 'Ariana'
```

f)
```
('Segunda', 87, 65) + 'Terça'
```

g)
```
'A' += ('B', 'C')
```

h)
```
x = 7

del x

print(x)
```

i)
```
numeros = [1, 2, 3, 4, 5]

print(numeros.index(10))
```

j)
```
numeros = [1, 2, 3, 4, 5]

numeros.extend(6, 7, 8)
```

k)
```
numeros = [1, 2, 3, 4, 5]

numeros.remove(10)
```

l)
```
valores = []

valores.pop()
```

Questão 2. Implemente um script na linguagem Python que leia do usuário uma mensagem de texto e a converta fazendo as seguintes alterações:

- A letra **a** ➔ 4
- A letra **b** ➔ 8
- A letra **e** ➔ 3
- A letra **o** ➔ 0
- A letra **s** ➔ 5
- A letra **t** ➔ 7

Seu programa considerar letras maiúsculas e minúsculas como iguais. Ao final deve-se imprimir na tela a nova mensagem resultante após as trocas realizadas.

Questão 3. Em uma pesquisa realizada no IFES, **N** alunos foram solicitados a avaliar em uma escala de 1 a 5 a qualidade da comida no refeitório, sendo 1 "PÉSSIMO" e 5 "EXCELENTE". Desenvolva um script em Python que leia as avaliações dos alunos com avaliações na escala de 1-5 (somente valores inteiros). Seu programa deve encerrar a votação quando o usuário digitar um valor que esteja fora do intervalo permitido. Seu programa deve ao final determinar e exiba a frequência de cada avalição e as seguintes estatísticas das respostas lidas: mínima, máxima, média, mediana, moda, variância e desvio padrão.

Questão 4. Desenvolva um programa na linguagem Python que leia do usuário duas listas de números inteiros **l1** e **l2** com o mesmo tamanho **n** (esse valor também deve ser definido pelo usuário). Essas listas correspondem a pontuação de dois alunos **a1** e **a2** em um desafio. Seu programa deve exibir a pontuação dos dois alunos (**p_a1** e **p_a2**), que é dado usando as regras explicadas seguir. Por exemplo, dadas as listas **l1 = [4, 6, 7, 2]** e **l2 = [3, 9, 8, 5]**, seu programa deve exibir **p_a1 = 1** e **p_a2 = 3**.

- Se **l1[i] > l2[i]**, o aluno 1 (a1) ganha um ponto;
- Se **l1[i] < l2[i]**, o aluno 2 (a2) ganha um ponto;
- Se **l1[i] = l2[i]**, ninguém ganha ponto.

Questão 5. Crie um programa em Python que leia uma lista **p** contendo **n** números inteiros representando a pontuação de um atleta nos seus jogos. Seu programa deve imprimir na tela a quantidade de vezes que esse atleta bateu seu recorde de pontuação máxima (**p_max**) e mínima (**p_min**). Por exemplo, dada lista **p = [10, 5, 20, 20, 4, 5, 2,**

25, 1] seu programa deve exibir **p_max = 2** e **p_min = 4**. Em verde estão representados os recordes de maior pontuação e em vermelho de menor pontuação.

Questão 6. Desenvolva uma função em Python que recebe por parâmetro uma matriz n x m, calcula e retorna sua transposta. Teste sua função com alguns exemplos.

Questão 7. Uma matriz quadrada de números inteiros é um quadrado mágico se o valor da soma dos elementos de cada linha, de cada coluna e da diagonal principal e da diagonal secundária é o mesmo. Além disso, a matriz deve conter todos os números inteiros do intervalo [1 ... n × n]. Exemplo:

$$\begin{bmatrix} 15 & 8 & 1 & 24 & 17 \\ 16 & 14 & 7 & 5 & 23 \\ 22 & 20 & 13 & 6 & 4 \\ 3 & 21 & 19 & 12 & 10 \\ 9 & 2 & 25 & 18 & 11 \end{bmatrix}$$

A matriz acima é um quadrado mágico, cujas somas valem 65. Escreva uma função em Python que, dada uma matriz quadrada, verifique se ela é um quadrado mágico. Teste sua função com alguns exemplos.

Questão 8. Dado uma lista de números inteiros de tamanho **N** (a_1, a_2, a_3, ..., a_n) e uma variável soma (s). Construa uma função em Python que verifique se existe um par (a_i, a_j) com i ≠ j, cuja soma seja igual a s ($a_i + a_j$ = s). Sua função deve receber por parâmetro uma lista e valor da soma. Teste sua função com alguns exemplos.

Exemplos:

- (1, 2, 3, 4) e soma = 8 ➔ False
- (1, 2, 3, 4, 4) e soma = 8 ➔ True (4, 4)

Questão 9. Uma palavra que é escrita de forma idêntica quando lida da esquerda para direita e vice-versa, por exemplo 'radar', é chamada de palíndroma. Escreva uma função em Python chamada de *eh_palindroma* que recebe uma string por parâmetro e retorna

True se for um palíndroma e *False* caso contrário. Sua função deve ser *case-insensitive*, ou seja, letras maiúsculas e minúsculas devem ser consideradas iguais. Teste sua função com alguns exemplos.

Questão 10. Escreva a função *reverte_*agenda, que aceite como entrada uma agenda, ou seja, um dicionário mapeando nomes (as chaves) a números de telefone (os valores). A função deverá retornar outro dicionário representando a agenda reversa, mapeando números de telefone (as chaves) aos nomes (os valores). Teste sua função com alguns exemplos.

Questão 11. Desenvolva uma função em Python chamada de *codificar*, que recebe uma string como parâmetro e retorna uma string criptografia definida usando as regras a seguir. Cada caractere em uma posição ímpar *i* no alfabeto será criptografado com o caractere na posição *i* + 1, e cada caractere em uma posição par *i* será criptografado com o caractere na posição *i* – 1. Por exemplo, a letra 'a' é criptografada com 'b', a letra 'b' com 'a', 'c' com 'd', 'd' com 'c' e assim por diante. Caracteres minúsculos deverão permanecer minúsculos, e caracteres maiúsculos deverão permanecer assim. Teste sua função com alguns exemplos.

Questão 12. Implemente uma função em Python chamada *sublista* que receba duas listas por parâmetro *lista_1* e *lista_2* e retorna *True* se a *lista_1* é uma sublista da *lista_2*, ou *False* caso contrário. Uma lista 1 é considerada uma sublista de uma lista 2, se todos os elementos da lista 1 aparecem na lista 2 na mesma ordem em que eles ocorrem na lista 1. Crie uma função principal e teste sua função com alguns exemplos.

Exemplos:

- [15, 1, 100] é uma sublista de [20, 15, 30, 50, 1, 100]
- [15, 50, 20] não é uma sublista de [20, 15, 30, 50, 1, 100]

Questão 13. Desenvolva uma função na linguagem Python que recebe por parâmetro um texto, calcula e retorna um valor representando a sua pontuação com base nas seguintes regras:

- A pontuação de uma palavra é 2 se a palavra contém um número ímpar de vogais, caso contrário, o escore da palavra é igual a 1. Letras maiúsculas e minúsculas

devem ser consideradas iguais. Considere que o texto de entrada não possui nenhum símbolo de acentuação. A lista de vogais é: [‘a’, ‘e’, ‘i’, ‘o’, ‘u’]

- A pontuação total do texto é dada pela média das pontuações das suas palavras, ou seja, soma as pontuações das palavras individualmente e divide pelo total de palavras do texto.
- Fragmente o texto por espaço em branco para identificar as suas palavras.

Exemplos:

a) **Entrada:** Python e muito fácil
   Saída: 1.75
b) **Entrada:** Ola Mundo
   Saída: 1.00

Questão 14. Escreva um programa em Python para armazenar uma agenda de telefones em um dicionário. Cada pessoa pode ter um ou mais telefones e a chave do dicionário é o nome da pessoa. Seu programa deve ter as seguintes funções:

- **incluir_novo_nome** – essa função acrescenta um novo nome na agenda, com um ou mais telefones. Ela deve receber como argumentos o nome e os telefones.
- **incluir_telefone** – essa função acrescenta um telefone em um nome existente na agenda. Caso o nome não exista na agenda, você deve perguntar se a pessoa deseja incluí-lo. Caso a resposta seja afirmativa, use a função anterior para incluir o novo nome.
- **excluir_telefone** – essa função exclui um telefone de uma pessoa que já está na agenda. Se a pessoa tiver apenas um telefone, ela deve ser excluída da agenda.
- **excluir_nome** – essa função exclui uma pessoa da agenda.
- **consultar_telefone** – essa função retorna os telefones de uma pessoa na agenda.

Seu programa deve ter dois arquivos de código, um arquivo somente com as implementações das funções e um outro arquivo contendo um menu com uma opção para que o usuário execute cada uma das funções listadas anteriormente. Seu programa deve ficar repetindo a exibição do menu e respondendo as ações do usuário até que ele digite a opção 6 – Sair.

------ Menu Agenda ------

1 – Incluir novo nome

2 – Incluir telefone

3 – Excluir telefone

4 – Excluir nome

5 - Consultar telefone

6 - Sair

Questão 15. O cenário ilustrado nesta questão é fictício e não reflete a realidade. O número que compõe o Cadastro Nacional da Pessoa Jurídica (CNPJ) segue o padrão xx.xxx.xxx/xxxx-xx, onde x é um dígito. Esse número é composto por três segmentos de dígitos, sendo o primeiro o número da inscrição propriamente dito, o segundo (após a barra) o número de filiais e o terceiro representados pelos últimos dois valores que são os dígitos verificadores. Os dois dígitos verificadores do CNPJ são gerados a partir dos 12 primeiros dígitos usando as regras apresentadas a seguir. Desenvolva uma função na linguagem Python que recebe por parâmetro uma string contendo um número de CNPJ no formato completo e retorna *True* se o CNPJ for válido, caso contrário retorne *False*. Teste sua função com alguns exemplos.

Para ilustrar os cálculos de validação dos dígitos verificadores vamos usar este CNPJ fictício **11.222.333/0001-81.**

1. A geração do 1º dígito verificador é realizada multiplicando os 12 primeiros dígitos do CNPJ da esquerda para direita pela seguinte sequência de números: **5, 4, 3, 2, 9, 8, 7, 6, 5, 4, 3 e 2.** Usando o CNPJ de exemplo teremos o seguinte cálculo:
   a) **1** * 5 + **1** * 4 + **2** * 3 + **2** * **2** + 2 * 9 + **3** * 8 + **3** * 7 + **3** * 6 + **0** * 5 + **0** * 4 + **0** * 3 + **1** * 2 = 102
   b) O próximo passo é calcular o resto da divisão de 102 por 11, cujo resultado é 3.

c) Caso o resto da divisão seja menor do que 2, o digito verificador é igual a 0. Caso contrário, o 1º digito verificar é igual a 11 menos o valor do resto da divisão do passo b).

i. No nosso exemplo, como 3 é maior do que 2, então calcula-se 11 – 3 = 8.

ii. Logo, o 1º dígito verificador deve ser igual a 8, que é igual ao do nosso CNPJ de exemplo.

2. A geração do 2º dígito verificador é semelhante ao do 1º digito e é realizada multiplicando os 12 primeiros dígitos mais o 1º digito verificador do CNPJ da esquerda para direita pela seguinte sequência de números: **6, 5, 4, 3, 2, 9, 8, 7, 6, 5, 4, 3 e 2.** Usando o CNPJ de exemplo teremos o seguinte cálculo:

a) **1** * 6 + **1** * 5 + **2** * 4 + **2** * 3 + 2 * 2 + **3** * 9 + **3** * 8 + **3** * 7 + **0** * 6 + **0** * 5 + **0** * 4 + **1** * 3 + **8** * 2 = 120

b) O próximo passo é calcular o resto da divisão de 120 por 11, cujo resultado é 10.

c) Caso o resto da divisão seja menor do que 2, o digito verificador é igual a 0. Caso contrário, o 2º digito verificar é igual a 11 menos o valor do resto da divisão do passo b).

i. No nosso exemplo, como 10 é maior do que 2, então calcula-se 11 – 10 = 1.

ii. Logo, o 2º dígito verificador deve ser igual a 1, que é igual ao do nosso CNPJ de exemplo, o que indica que o CNPJ é válido.