Módulo 7 - Strings

Aula 7.1 - Introdução - Exercícios

Instruções para submissão: Para cada questão, crie no VSCode um arquivo chamado **`aula1_questaoX.py`**, sendo **`X`** o número da questão. Faça commit de todos os arquivos para a pasta **`modulo7`** do seu repositório da disciplina no GitHub.

1) Escreva um programa que solicita o nome do usuário e o imprime em forma de escada, como indicado no exemplo a seguir.

```
Digite seu nome: Fulano
F
Fu
Ful
Fula
Fulan
Fulano
```

2) Escreva um programa que solicite ao usuário inserir seu primeiro nome e sobrenome separadamente. Em seguida, concatene essas duas strings e exiba a mensagem de boas-vindas.

```
Digite seu primeiro nome: Alice
Digite seu sobrenome: Silva
Bem-vinda, Alice Silva!
```

3) Escreva um script que dado uma frase conta os espaços em branco.

```
Digite a frase: Meu amor mora em Roma e me deu um ramo de flores
Espaços em branco: 11
```

4) Faça um programa que leia um número de celular e, caso o número tenha apenas 8 dígitos, acrescente o 9 na frente. Caso o número já tenha 9 dígitos, verifique se o primeiro dígito é 9. Adicione o separador `"-"` na sua impressão.

```
Digite o número: 97651234
Número completo: 99765-1234
Digite o número: 980876543
Número completo: 98087-6543
```

5) Implemente um código que leia uma string do usuário e imprima quantas vogais existem na frase e quais os seus índices da string. Dica: **`letra in "aeiou"`**. Exemplo:

```
Digite uma frase: Meu amor mora em Roma e me deu um ramo de
flores
19 vogais
Índices [1, 2, 4, 6, 10, 12, 14, 18, 20, 22, 25, 28, 29, 31, 35, 37,
40, 44, 46]
```

6) Dada uma string e uma palavra objetivo, encontre todos os anagramas da palavra objetivo. Anagramas são palavras com os mesmos caracteres rearranjados.

```
Digite uma frase: Meu amor mora em Roma e me deu um ramo de
flores
Digite a palavra objetivo: amor
Anagramas: ["amor", "mora", "ramo", "Roma"]
```

7) Crie a função **`encrypt()`** que recebe uma lista de strings e retorna os nomes criptografados, bem como a chave da criptografia. Regras:

   - Chave de criptografia: gere um valor n aleatório entre 1 e 10
   - Substitua cada caracter **`c`** pelo caracter **`c + n`**. Trabalharemos apenas com o intervalo de caracteres visíveis (entre 33 e 126 na tabela Unicode)

```
nomes = ["Luana", "Ju", "Davi", "Vivi", "Pri", "Luiz"]
chave_aleatoria = 5
nomes_cript = ['Qzfsf', 'Oz', 'If{n', '[n{n', 'Uwn', 'Qzn!']
```

8) Desenvolva um validador de CPF. Solicite do usuário um CPF na forma XXX.XXX.XXX-XX (lido como string) e imprima "Válido" ou "Inválido".

   O primeiro passo é calcular **o primeiro dígito verificador.** Separamos os primeiros 9 dígitos do CPF (ex: 111.444.777) e multiplicamos cada um dos números, da direita para a esquerda por números crescentes a partir do número 2, como no exemplo abaixo:

CPF	1	1	1	4	4	4	7	7	7
Multiplicador	10	9	8	7	6	5	4	3	2
Resultado	10	9	8	28	24	20	28	21	14

   Em seguida somamos o resultado: **`10+9+8+28+24+20+28+21+14 = 162`**

   Pegamos o resultado e dividimos por 11: **`162 / 11 = 14 com resto 8`**

   - Se o resto da divisão for menor que 2, então o dígito é igual a 0 (Zero).

- Se o resto da divisão for maior ou igual a 2, então o dígito verificador é igual a 11 menos o resto da divisão (11 - resto).

No nosso exemplo temos que o resto é 8 então faremos `11-8 = 3`. Logo o primeiro dígito verificador é 3. Então sabemos que o CPF deve ser: `111.444.777-3X`

Para calcular o segundo dígito vamos usar o primeiro dígito já calculado. Vamos montar a mesma tabela de multiplicação usada no cálculo do primeiro dígito. Só que desta vez usaremos na segunda linha os valores 11,10,9,8,7,6,5,4,3,2 já que estamos incluindo mais um dígito no cálculo:

CPF	1	1	1	4	4	4	7	7	7	**3**
Multiplicador	11	10	9	8	7	6	5	4	3	**2**
Resultado	11	10	9	32	28	24	35	28	21	**6**

Somamos: `11 + 10 + 9 + 32 + 28 + 24 + 35 + 28 + 21 + 6 = 204`
Dividimos o total do somatório por 11 e consideramos o resto da divisão.
`204 / 11 = 18 e resto 6`

Aplicamos a mesma regra que utilizamos para obter o primeiro dígito:
- Se o resto da divisão for menor que 2, então o dígito é igual a 0 (Zero).
- Se o resto da divisão for maior ou igual a 2, então o dígito é igual a 11 menos o resto da divisão (11 - resto).

`11-6 = 5`, logo 5 é o nosso segundo dígito verificador e o CPF completo é:

111.444.777-35

O CPF de entrada deve ser considerado válido se os dígitos fornecidos pelo usuário forem os mesmos dígitos calculados através do processo acima.

Exemplos válidos	**Exemplos inválidos**
545.315.761-52	545.315.761-12
473.063.662-70	473.063.662-98
775.682.566-77	775.682.566-13