Avaliação 1 - Lógica de Programação

Recomendação:

Crie um arquivo para cada questão:
Exemplo:
Root/
exercicio1.py
exercicio2.py
exercicio3.py

Exercício 1. Criação de usuário

No sistema de uma empresa, a criação do usuário de uma pessoa é feita unindo as seguintes informações:

- as duas primeiras letras do sobrenome;
- a primeira letra do nome;
- 1 dígito referente ao mês de nascimento da pessoa:
  - se a pessoa nasceu no mês 11, o dígito é 2 (1 + 1);
  - caso ela tenha nascido no mês 12, o dígito é 3 (1 + 2);
  - nos outros casos mantém-se o dígito único;
- e o código “br” no final.

Exemplo:

```
Digite seu nome completo: Lucas Martins Oliveira
Digite sua data de nascimento (dd/mm/aaaa): 12/08/1992

O login solicitado é : oll8br
```

Crie um programa que crie um usuário seguindo o passo a passo acima.
Não é necessário verificar se a pessoa digitou a data de nascimento corretamente; considere que ela sempre estará no formato dd/mm/aaaa.

Exercício 2. Lista de times

Abaixo temos uma lista de times e suas classificações. Utilizando esta lista, crie outras três listas e desenvolva um programa que separe os times entre primeira, segunda e terceira divisão. Eles devem ser salvos nas três novas listas exatamente como aparecem na saída.

```
lista = [
    '1_tigre azul', '2_leões do norte', '1_falcões', '3_tubarões',
    '2_águia real', '3_panteras', '1_dragões', '2_urso polar',
    '3_cobras negras', '2_lobos do sul', '3_raposas douradas'
]
```

Saída:

```
Primeira divisão: ['Tigre Azul', 'Falcões', 'Dragões']
Segunda divisão: ['Leões Do Norte', 'Águia Real', 'Urso Polar', 'Lobos Do Sul']
Terceira divisão: ['Tubarões', 'Panteras', 'Cobras Negras', 'Raposas Douradas']
```

Exercício 3. Dias e meses

Abaixo temos um dicionário com os 12 meses do ano, e quantos dias temos em cada mês. Peça para o usuário digitar um mês, e então mostre a quantidade de dias referente a esse mês.

Crie um dicionário com os seguintes dados:

janeiro – 31
fevereiro – 28/29
março – 31
abril – 30
maio – 31
junho – 30
julho – 31
agosto – 31
setembro – 30
outubro – 31
novembro – 30
dezembro - 31

Importante:

- Use o valor digitado pelo usuário para acessar o dicionário, sem estruturas condicionais:

```
entrada = input("Escolha um mes: ").lower()
```

- Caso o valor seja inválido, use um try-except para tratar o erro apontado. Continue o programa até o usuário digitar uma chave válida;
- Caso o mês escolhido seja fevereiro, pergunte o ano (para saber se o ano é bissexto ou não); trate a entrada até o usuário entrar um valor inteiro e maior que 1900.

Saída:

```
Escolha um mês: 01
Valor inválido. Tente novamente.

Escolha um mês: fevereiro
Digite o ano (maior que 1900): 1800
Valor inválido. Tente novamente.

Digite o ano (maior que 1900): teste
Valor inválido. Tente novamente.

Digite o ano (maior que 1900): 2025
O mês de Fevereiro tem 28 dias.
```

Exercício 4. Estrutura de repetição while

Utilizando a estrutura de repetição while, crie os seguintes loops aninhados:

a. Apresente no console 10 sequências diferentes de valores, a primeira sequência começa em 10 e termina em 1, a próxima sequência não irá conter o maior valor da sequência anterior, a última sequência somente terá o número 1.

```
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7
1 2 3 4 5 6
1 2 3 4 5
1 2 3 4
1 2 3 4
1 2 3
```

```
1 2
1
```

b. Utilizando as funções ord() e chr(), incremente o código anterior para que ele mostre a seguinte sequência de letras:

```
a b c d e f g h i j k
a b c d e f g h i j
a b c d e f g h i
a b c d e f g h
a b c d e f g
a b c d e f
a b c d e
a b c d
a b c
a b
a
```

```
ord("a") # 97
chr(97)  # "a"
```

Exercício 5. Palavra

Crie uma função que inverte a primeira metade de uma palavra, ou seja:

Input: batata

bat ata -> tab ata -> tabata

output:

```
tabata
```

input: busca

bu sca -> ub sca -> ubsca

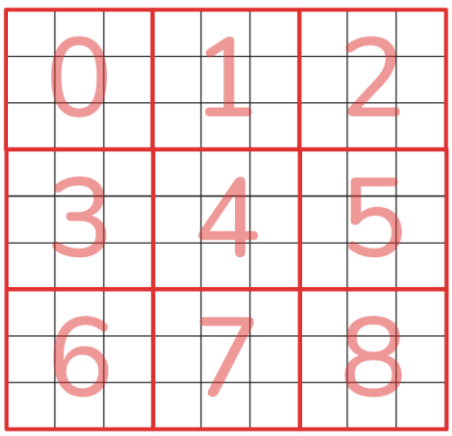
output:

```
ubsca
```

Exercício 6. Sudoku

Sudoku é um jogo baseado na combinação lógica de números. Seu objetivo é colocar números de 1 a 9 em cada uma das células vazias, o jogo é constituído numa grade de 9x9 com subdivisões de 3x3, na qual as linhas da grade principal e as subdivisões deve conter todos os números de 1 a 9 sem se repetir na linha ou grade.

As subdivisões são da seguinte forma:



Portanto:

Dentro de uma divisão não pode haver número repetido.
Na mesma linha não pode haver número repetido.
Na mesma coluna não pode haver número repetido.

a. Crie uma função que recebe o nome de um arquivo txt dos templates, e retorna uma matriz do jogo presente dentro do arquivo.

b. Crie uma função chamada 'verificar_jogada()', que recebe por parâmetro a matriz do jogo, uma posição 'x', uma posição 'y' e um valor 'v', a função com base nesse valor e nessa posição deve validar se o valor pode ou não ser inserido na posição passada, o retorno da função deve ser um booleano, True caso possa ser inserido, False caso não possa.

c. Crie uma função chamada 'verificar_jogo()' que recebe por parâmetro a matriz do jogo, com isso ela deve validar se esse jogo está em conformidade com as regras, se todos os valores presentes no jogo estão válidos ou não.

Exercício 7. For

Crie um módulo chamado *trig.py*, onde você construirá duas funções: a função **seno** e a função **cosseno**. Para isso você tem as seguintes informações:

- O ângulo digitado pelo usuário estará em graus, e no cálculo é usado o ângulo em radianos; use a seguinte fórmula para fazer a conversão:

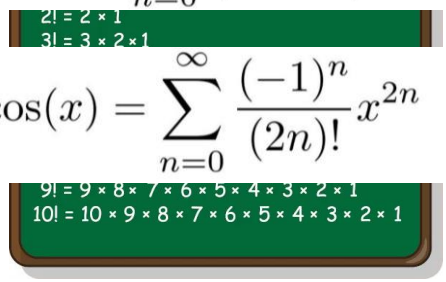
```
from math import pi

rad = graus*pi/180
```

- Para implementar as funções, construa uma função **fatorial**, que recebe um valor inteiro e retorna o fatorial desse número. Relembrando:

$$\sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1}$$

$$\cos(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n}$$



- Agora implemente as funções **seno** e **cosseno**, que são os dois somatórios apresentados a seguir.
  - Faça o somatório indo de 0 a 5;
  - **x** é o **ângulo em radianos**.

- Por fim crie um outro arquivo que importa o módulo *trig.py* e aplique as funções criadas.

Saída:

```
Digite um ângulo em graus: 60
Seno: 0.86603
Cosseno: 0.50000
```