REGRAS:

- Todos os exercícios devem funcionar com qualquer valor de entrada. Faça testes.
- Todas as variáveis, exceção das constantes e quando o valor é informado, devem ter seu valor solicitado ao usuário (input).

EXERCÍCIOS

1) Peça 10 números e conte quantos são múltiplos de 3. Use for.

```
403  #Atividade de revisão
404  for numero in range(0,11,3):
405      print(numero)
```

```
Run    exercícios
C:\Users\51536116807\PycharmProjects\
0
3
6
9
```

2) Crie um programa que simule o uso de senha com tentativas infinitas até digitar a senha correta (use while True).

```
senha = "Python123*"
while True:
    tentativa = input("Digite a senha: ")
    if tentativa == senha:
        print("Senha correta")
        break
    else:
        print("Senha incorreta")
```

```
exercícios
C:\Users\51536116807\PycharmProjects\PythonPr
Digite a senha: Python
Senha incorreta
Digite a senha: 123
Senha incorreta
Digite a senha: Python123*
Senha correta
```

3) Monte um sistema que repita um menu até o usuário escolher sair. Use while e break.

```python
while True:
    print("Menu:")
    print("1. Salgados")
    print("2. Doces")
    print("3. Pratos de comida")
    print("4. Bebidas")
    print("5. Sobremesas")
    print("6. Sair")

    opcao = input("Escolha uma opção: ")

    if opcao == "1":
        print("Escolha: salgado")
    if opcao == "2":
        print("Escolha: Doces")
    if opcao == "3":
        print("Escolha: Pratos de comida")
    if opcao == "4":
        print("Escolha: Bebidas")
    if opcao == "5":
        print("Escolha: Sobremesas")
    if opcao == "6":
        print("Escolha: sair")
        break
```

```
Run  exercícios
C:\Users\51536116807\PycharmProject
Menu:
1. Salgados
2. Doces
3. Pratos de comida
4. Bebidas
5. Sobremesas
6. Sair
Escolha uma opção: 3
Escolha: Pratos de comida
Menu:
1. Salgados
2. Doces
3. Pratos de comida
4. Bebidas
5. Sobremesas
6. Sair
Escolha uma opção: 6
Escolha: sair

Process finished with exit code 0
```

4) Crie um programa que peça dois números inteiros e exiba todos os números entre eles que são primos. Use for.

5) O usuário tem 3 tentativas para acertar a senha. Se errar todas, o acesso é bloqueado. Use while.

```python
senha = "Python123*"
tentativas = 3
while True:
    tentativa = input("Digite a senha: ")
    if tentativa == senha:
        print("Senha correta")
        break
    else:
        tentativas -=1
        print(f'Senha incorreta. Tentativas restantes: {tentativas}')
    if tentativas == 0:
        print("Acesso bloqueado!")
```

```
C:\Users\51536116807\PycharmProjects\Pyth
Digite a senha: 123
Senha incorreta. Tentativas restantes: 2
Digite a senha: Python123*
Senha correta

Process finished with exit code 0
```

6) Peça 10 números e separe em duas listas: pares e ímpares. Mostre as duas no final.

```python
#6
pares = []
impares = []

for i in range(10):
    num = int(input(f"Digite o {i + 1}° número: "))

    if num % 2 == 0:
        pares.append(num)
    else:
        impares.append(num)

print( pares)
print( impares)
```

```
Run  exercícios
Digite o 2° número: 2
Digite o 3° número: 3
Digite o 4° número: 4
Digite o 5° número: 5
Digite o 6° número: 6
Digite o 7° número: 7
Digite o 8° número: 8
Digite o 9° número: 9
Digite o 10° número: 10
[2, 4, 6, 8, 10]
[1, 3, 5, 7, 9]
```

7) Peça uma frase e conte quantas vogais há nela. Mostre o total de cada uma (a, e, i, o, u).

```
frase = input("Digite uma frase: ").lower()
vogais = {'a': 0, 'e': 0, 'i': 0, 'o': 0, 'u': 0}

for letra in frase:
    if letra in vogais:
        vogais[letra] += 1

for v, qtd in vogais.items():
    print(f"{v}: {qtd}")
```

```
Digite uma frase: amo o senai
a: 2
e: 1
i: 1
o: 2
u: 0
```

8) Simule o lançamento de uma moeda até sair "cara" três vezes seguidas. (Dica: usar random.choice(["cara", "coroa"]) e while).

```
import random

contador = 0
tentativas = 0

while contador < 3:
    resultado = random.choice(["cara", "coroa"])
    print(resultado)
    tentativas += 1
    if resultado == "cara":
        contador += 1
    else:
        contador = 0

print(f"Total de lançamentos: {tentativas}")
```

```
coroa
coroa
coroa
cara
coroa
coroa
cara
cara
coroa
cara
cara
coroa
coroa
coroa
cara
coroa
cara
cara
cara
Total de lançamentos: 19
```

9) Crie um programa que leia uma sequência de números e determine quantos números são menores que a média.

```
numeros = list(map(float, input("Digite números separados por espaço: ").split()))
media = sum(numeros) / len(numeros)
menores = [n for n in numeros if n < media]

print(f"Média: {media}")
print(f"Números menores que a média: {menores}")
```

```
Digite números separados por espaço: 5 6 9 2
Média: 5.5
Números menores que a média: [5.0, 2.0]

Process finished with exit code 0
```

10) Crie um programa que leia uma sequência de números e determine o segundo maior número.

```
numeros = list(map(int, input("Digite números separados por espaço: ").split()))
unicos = list(set(numeros))  # Remove duplicatas
unicos.sort(reverse=True)

if len(unicos) >= 2:
    print(f"Segundo maior número: {unicos[1]}")
else:
    print("Não há segundo maior número.")
```

```
Digite números separados por espaço: 10 2 7 4
Segundo maior número: 7
```

11) Mostre a tabela de teste de mesa preenchida referente ao programa abaixo:

```
a = 1
b = 2
c = 0
for i in range(1, 6):
    if i % 2 == 0:
        a += i
    else:
        b *= i
    c = a + b
print(a, b, c)
```

i	a	b	c	print(a, b, c)
1	1	2	3	A=7, b=30, c= 37

2	3	2	5	
3	3	6	9	
4	7	6	13	
5	7	30	37	

12) Mostre a tabela de teste de mesa preenchida referente ao programa abaixo:

```
x = 3
y = 2
z = 0
while x < 8:
   for j in range(2):
      z += y
      y += 1
   x += 1
print(x, y, z)
```

xx	y	z	j	print(x, y, z)
4	2	0	z=5, y=4	x=8, y=12,z=65
5	4	5	z=14, y=6	
6	6	14	z=27, y=8	
7	8	27	z=44, y=10	
8	10	44	z=65, y=12	

DESAFIO - obrigatório

1) Simulação de Populações de Coelhos

Crie um programa que simule o crescimento de uma população de coelhos ao longo de várias gerações. Os coelhos se reproduzem a uma taxa fixa a cada geração, e uma porcentagem deles morre a cada geração. O programa deve solicitar ao usuário a taxa de reprodução, a taxa de mortalidade e o número inicial de coelhos. Use um loop for ou while para simular várias gerações e exiba a população de coelhos após um número de gerações especificado pelo usuário.

```
populacao = int(input("Digite o número inicial de coelhos: "))
taxa_reproducao = float(input("Taxa de reprodução por geração (ex: 0.5 para 50%): "))
taxa_mortalidade = float(input("Taxa de mortalidade por geração (ex: 0.1 para 10%): "))
geracoes = int(input("Número de gerações: "))
for i in range(geracoes):
    nascimentos = populacao * taxa_reproducao
    mortes = populacao * taxa_mortalidade
    populacao = populacao + nascimentos - mortes
    print(f"Geração {i+1}: {int(populacao)} coelhos")
```

```
Digite o número inicial de coelhos: 10
Taxa de reprodução por geração (ex: 0.5 para 50%): 0.4
Taxa de mortalidade por geração (ex: 0.1 para 10%): 0.1
Número de gerações: 5
Geração 1: 13 coelhos
Geração 2: 16 coelhos
Geração 3: 21 coelhos
Geração 4: 28 coelhos
Geração 5: 37 coelhos

Process finished with exit code 0
```

2) Jogo da Forca. Crie um jogo da forca, onde:

- **Palavra oculta**: A palavra é escolhida aleatoriamente de uma lista de palavras pré-definidas. A palavra deve ser exibida com espaços (_) representando cada letra. O jogador deve tentar adivinhar as letras da palavra.
- **Feedback dinâmico**:
  - O jogo deve mostrar a palavra com as letras corretas já adivinhadas a cada tentativa.
  - O jogo também deve mostrar as **letras erradas** que o jogador já tentou, para evitar que ele repita a mesma letra.
  - Caso o jogador tente uma letra que já tenha sido usada (correta ou incorreta), o jogo deve **informar que ele já tentou essa letra**, pedindo que ele tente outra.
- **Número de tentativas**: O jogador tem um total de **6 tentativas** para errar antes de perder o jogo. A cada erro, o número de tentativas diminui.
- **Mensagens de vitória ou derrota**:
  - O jogo deve informar ao jogador quando ele **ganhar**, revelando a palavra completa.

- Caso o jogador **perca**, o jogo deve revelar a palavra e informar que ele perdeu

```python
import random
palavras = ["banana", "computador", "python", "floresta", "universo"]
palavra = random.choice(palavras)
letras_certas = set()
letras_erradas = set()
tentativas = 6
while tentativas > 0:
    exibicao = [letra if letra in letras_certas else "_" for letra in palavra]
    print("Palavra:", " ".join(exibicao))
    print("Erradas:", " ".join(letras_erradas))
    if "_" not in exibicao:
        print("Parabéns! Você acertou a palavra:", palavra)
        break
    tentativa = input("Tente uma letra: ").lower()
    if tentativa in letras_certas or tentativa in letras_erradas:
        print("Você já tentou essa letra. Tente outra.")
        continue
    if tentativa in palavra:
        letras_certas.add(tentativa)
    else:
        letras_erradas.add(tentativa)
        tentativas -= 1
        print(f"Letra incorreta. Tentativas restantes: {tentativas}")
if tentativas == 0:
    print("Você perdeu! A palavra era:", palavra)

```

```
Palavra: _ _ _ _ _ _
Erradas:
Tente uma letra: y
Palavra: _ y _ _ _ _
Erradas:
Tente uma letra: p
Palavra: p y _ _ _ _
Erradas:
Tente uma letra: t
Palavra: p y t _ _ _
Erradas:
Tente uma letra: h
Palavra: p y t h _ _
Erradas:
Tente uma letra: o
Palavra: p y t h o _
Erradas:
Tente uma letra: n
Palavra: p y t h o n
Erradas:
Parabéns! Você acertou a palavra: python

Process finished with exit code 0
```