Objetivo:

Aprender como utilizar o comando de repetição while:

Sintaxe:

while condição**:**
{Bloco de comandos}

Ferramentas:

Exemplo 1: Imprime de 1 a 10

```
print("Início do programa!")
print("Imprime de 1 a 10 \n")
cont = 1
while cont <= 10:
        print(cont)
        cont = cont + 1
print("Fim do programa!")
```

Exemplo 2: Par ou ímpar

```
x = 1
while x != 0:
        x = int(input("Digite um número inteiro ou zero para sair: "))
        if x % 2 == 0:
            print("%d é par" % x)
       else:
            print("%d é impar" % x)
```

Exemplo 3: Primo Fast com else

```
#primo fast
N = int(input("Digite N: "))
i = 2
while i < N:
    R = N % i
    if R == 0:
        print("{} não é primo!".format(N))
        break
    i += 1
else:
    print("{} é primo!".format(N))
```

Exemplo 4: Primo Slow

```
#primo slow
N = int(input("Digite o número: "))
cont = 0
i = 2
while i < N:
    R = N % i
    if R == 0:
        cont += 1
    i += 1
if cont == 0:
    print("{} é primo!".format(N))
else:
    print("{} não é primo!".format(N))
```

Exemplo 5: Fatorial

```
#fatorial
n = int(input("Fatorial de: ") )
result=1
count=1
while count <= n:
    result *= count # equivale a result = result * count
    count += 1 # equivale a count = count + 1
print(result)
```

Exemplo 6: Fatorial Function from

```
# fatorial
# Exceptions ( negative number )
from math import factorial
n = int(input("Digite o valor n = "))
fat = factorial(n)
print ("O fatorial de {} é: {}".format(n, fat))
```

Exemplo 7: Fatorial Function import

```
# fatorial
# Exceptions ( negative number )
import math
n = int(input("Digite o valor n = "))
fat = math.factorial(n)
print ("O fatorial de {} é: {}".format(n, fat))
```

Exemplo 8: Pesquisa na Lista

```
print("Pesquisa sequêncial")
n = int(input("Digite n: "))
l = list(range(2, n+1, 2)) #função list + tipo range (class range(start, stop, step))
print("lista gerada",l)
x = int(input("Digite um número: "))
while x != 0:
    if x in l:
        print("{0} está na lista".format(x))
    else:
        print("{0} não está na lista".format(x))
    x = int(input("Digite um número: "))
print("Fim do Programa!")
```

Exemplo 9: Pesquisa sequencial

```
print("Pesquisa sequêncial")
n = int(input("Digite n: "))
l = list(range(2, n+1, 2)) #função list + tipo range (class range(start, stop, step))
print("lista gerada",l)
print("Número de elementos na lista: ",len(l)) #lenght
x = int(input("Digite um número: "))
while x != 0:
    i = 0
    while i < len(l) and l[i] != x:
        i+=1
    if i < len(l):
        print("{0} está na lista".format(x))
    else:
        print("{0} não está na lista".format(x))
    x = int(input("Digite um número: "))
print("Fim do Programa!")
```

Exemplo 10: Comando continue

```
x = 1
while x != 0:
    x = int(input("Digite um valor: "))
    if x <= 0:
        continue
    print("oi")
print("tchau!")
```

Exemplo 11- Comando break

```
while True:
    x = int(input("Digite um valor: "))
    if x == 0:
        break
    print("oi")
print("tchau!")
```

Pontos chave:

- Funcionamento do comando repetição while;
- Análise do comando while completo (break, continue e else);
- Uso de listas e depuração de código.

Curiosidade: A linguagem Python não implementa o comando do while encontrado em diversas linguagens de programação. Para realizar a análise da condição ao término de um bloco de comandos basta utilizar a estrutura abaixo:

while True**:**
{Bloco de comandos}
If condição**:**
break

Tarefas:

1. Elabore um programa em linguagem Python que leia um número inteiro N e, em seguida, mostre na tela os N primeiros termos da sequência de Fibonacci. Faça o programa de modo que o valor de N seja no mínimo 2.

A sequência de Fibonacci é uma sequência de números inteiros que têm as seguintes regras de formação: os dois primeiros termos são 0 e 1; do terceiro termo em diante cada termo é a soma dos dois anteriores.

Exemplo: Se N = **10**, então: **0, 1, 1, 2, 3, 5, 8, 13, 21, 34**
O código deve ser entregue no seguinte link:

https://forms.gle/uFsrEd1bVBpZZ396A

Observações:

- Trabalho deve ser realizado em duplas;
- Cada dupla entregará apenas um código;
- O nome dos integrantes deve constar na forma de comentário multiline no corpo do código;
- Cada dupla deverá entregar o arquivo fibonacci_RM1_RM2.py onde o RM é o (Registro de Matrícula) de cada um dos alunos.