Gerente:

Secretário: **Analista:**

Módulo 1: Estrutura de Repetição por Condição

Resumo:
Esta é uma atividade de aprendizagem orientada a processos (POGIL) que deverá ocorrer em equipes com o auxílio de um facilitador. Você e sua equipe deverão examinar imagens, gráficos, trechos de códigos ou textos para então passar por um conjunto de perguntas que irão guiá-los por um ciclo de exploração, criação de conceitos e aplicação.
Ao final dessa atividade, os estudantes deverão ser capazes de:

Conteúdo:

- Explicar as etapas de um loop
- Explicar a sintaxe de um loop **while**

Habilidades de Processo:

- Empatia
- Pensamento Crítico
- Trabalho em equipe

1. Examine o programa em Python a seguir:

```
#Codigo disponivel no CodeBench
#Imprime o nome inserido 20x
nome = input("Insira o seu nome: ")
contador = 0
while(contador < 20):
      print(nome)
      contador = contador + 1
```

Um loop é uma estrutura que permite que um bloco de código seja repetido uma ou mais vezes. No código acima o loop é representado pela palavra chave "**while**".

   a. No código acima, circule tudo que está associado ao **loop while**.

b. Teste o código no CodeBench. O que a linha de código `contador=contador + 1` faz ?

c. Como o interpretador do Python sabe quais linhas pertencem ao loop?

d. Toda estrutura de um loop requer três ações. Identifique a linha de código no programa que corresponde a cada uma das três ações.

- Inicialize uma variável usada na condição de teste:
- Inclua uma condição de teste que faça com que o loop termine quando a condição for falsa:
- Dentro do loop, atualize a variável que está sendo usada na condição de teste:

2 Teste o código a seguir no CodeBench. Explique o que cada linha de código faz:

```
#Codigo disponivel no CodeBench
1     numero = int(input("Digite um número: "))
2     contador = 1
3     while(contador <= numero):
4           if(contador % 10 == 0):
5                 print(contador)
6           else:
7                 print(contador, end = " ")
8           contador = contador + 1
```

3 O código a seguir deveria imprimir os números de 1 a 10, mas não imprime nada. Corrija o problema.

```
#Codigo disponivel no CodeBench
numero = 1
while numero >= 10:
      print(numero)
      numero = numero + 1
```

4 Teste o código a seguir no Codebench:

```
#Codigo disponivel no CodeBench
numero = 0
while numero <= 10:
      print(numero)
      numero = numero - 1
```

a. Explique o resultado do código acima:

b. O programa termina? Justifique sua resposta:

5 Teste o código a seguir:

```
#Codigo disponivel no CodeBench
numero = 1
while numero <= 10:
      if numero % 2 == 0:
            print(numero, end = " ")
      numero =  numero + 1
```

a. Qual é a saída do código:

b. O que faz com que a saída seja apenas em uma linha?

c. Qual estrutura de controle foi utilizada no código?

6 Utilize as instruções a seguir para criar um programa no CodeBench que solicite um número entre 1 e 10. Caso o número inserido esteja fora do intervalo, o programa deve solicitar novamente ao usuário um número válido. Conclua as etapas a seguir para escrever esse código.

a. Escreva uma linha de código que solicite ao usuário um número entre 1 e 10:

b. Escreva uma expressão para testar o número que o usuário digitou está dentro do alcance.

c. Use a expressão que foi criada anteriormente para escrever um **loop while** que será executado quando a entrada do usuário está fora do alcance. O corpo do loop deve informar ao usuário que ele digitou um número inválido e solicitar um número válido novamente.

d. Escreve o código que imprime uma mensagem informando ao usuário que ele digitou um número válido.

e. Coloque os códigos das etapas anteriores juntos. Teste o código. Funciona corretamente? Caso contrário, corrija e tente novamente:

f. Quantas vezes o loop foi executado?

Uma estrutura de loop para a qual sabemos o número de vezes que será executado é conhecido como **loop controlado por contagem**.

7 Às vezes não sabemos quantas vezes os dados devem ser inseridos. Por exemplo, suponha que você queira criar um programa que adiciona uma quantidade desconhecida de números positivos que serão inseridos pelo usuário. O programa para de adicionar números quando o usuário digita um número zero ou negativo. Em seguida, o programa imprime o total. Antes de criar este programa, revise as três ações necessárias para todo o loop:

a. Inicialize uma variável que será usada na condição de teste: o que será testado para determinar se o loop é executado? Escreva uma linha de código que inicialize uma variável a ser usada na condição de teste do loop para este programa. A variável deve conter um valor inserido pelo usuário.

b. Inclua uma condição de teste que faça com que o loop termine quando a condição for falsa: Qual é a condição de teste para o loop **while** usado neste programa?

c. Dentro do corpo do loop, atualize a variável usada na condição de teste: Escreva o código para o corpo do loop. Inclua o código para atualizar a variável na condição de teste.

d. Este é um loop controlado por contagem? Justifique sua resposta:

e. Complete o programa e teste no CodeBench. Funciona corretamente?

Os atalhos para operadores aritméticos fornecem uma maneira concisa de criar instruções de atribuição quando a variável no lado esquerdo da instrução de atribuição também está no lado direito. O operador de atalho aritmético (**+=**) geralmente é usado para incrementar uma variável

8 Teste o código a seguir no CodeBench:

```
#Codigo disponivel no CodeBench
número = 1
numero += 3
print(numero)
```

a. O que o operador de atalho "**+=**" faz?

b. O código `x += 5` é equivalente a qual das seguintes opções?

i. x = 5
ii. x = y + 5
iii. x = x + 5
iv. y = x + 5

c. Substitua o operador "**+=**" pelos operadores de atalho a seguir e execute o código no CodeBench. Explique o que cada operador faz.

i. **-=**

ii. ***=**

9 O código a seguir deveria imprimir números começando do 100 e terminando com 0. No entanto, está faltando uma linha de código. Adicione a linha ausente, usando operador aritmético de atalho. Reescreva o código com a linha inserida.

```
#Codigo disponivel no CodeBench
contagem = 100
while contagem >= 0:
      print(contagem)
```

10. Escreva um trecho de código no CodeBench que solicite ao usuário um número par. Se o número não for par, o usuário deverá receber uma mensagem solicitando um número par novamente.

11. Escreva um trecho de código no CodeBench que solicite do usuário números inteiros positivos. Os números digitados pelo usuário devem ser somados e armazenados em uma variável. O programa deve ser finalizado quando a soma dos números inseridos pelo usuário for igual ou maior a 15: