Ficha nº 5 – Instruções de repetição

Tópicos abordados

- Instruções *for*, *while*

Exercícios

1. Reescreva o seguinte segmento de código utilizando o ciclo while

```
for i in range(20):
    if i == 10:
            print("\n")
    else:
            print(i)
```

2. Números de 1 a 100

   a) Escreva um programa que imprima no ecrã todos os números inteiros de 1 a 100 utilizando o ciclo *while*
   b) Reescreva o programa usando o ciclo *for*.
   c) Altere o programa para que imprima os números por ordem inversa

3. Números pares entre 1 e N

   Escreva um programa que apresente no ecrã os números pares entre 1 e N, sendo N um número inteiro positivo e fornecido pelo utilizador. Proceda à respetiva validação dos dados de entrada.

4. Múltiplos de 3 entre m e M

   Implemente um programa que imprima no ecrã os múltiplos de 3 entre m e M, sendo m e M fornecidos pelo utilizador e em que m tem de ser inferior a M. Proceda à respetiva validação dos dados de entrada.

5. Soma dos números naturais

   Escreva um programa que calcule a soma dos N primeiros números naturais, sendo N fornecido pelo utilizador. Proceda à respetiva validação dos dados de entrada.

6. Fatorial

   a) Escreva um programa que permita calcular o fatorial de um número N inteiro positivo, fornecido pelo utilizador. Proceda à respetiva validação dos dados de entrada.
   b) Altere o programa de modo a que este permita calcular o fatorial de todos os números entre 1 e N, sendo N um número inteiro positivo fornecido pelo utilizador.
   Exemplo da saída de dados:
   O fatorial de 1 é 1
   O fatorial de 2 é 2
   O fatorial de 3 é 6
   …

7. Números primos

   a) Escreva um programa que dado um número inteiro positivo N, fornecido pelo utilizador, verifique se o mesmo é ou não um número primo.
   Nota: Um número primo só é divisível por 1 e por si mesmo.
   b) Altere o programa de modo a que este permita apresentar no ecrã a lista de todos os números primos existentes entre 1 e 100.
   c) Modifique o programa para que imprima simplesmente os números primos gémeos, i.e., os números primos que diferem entre si de duas unidades (ex: 3 e 5, 11 e 13, ...).

8. Máximo divisor comum

   Implemente um programa que permita calcular o máximo divisor comum – mdc(m,n) – entre dois números inteiros positivos m e n fornecidos pelo utilizador. Proceda à respetiva validação dos dados de entrada.

9. Jogo da adivinha

   a) Escreva um programa que deverá deixar o utilizador adivinhar um número inteiro introduzido previamente. A cada tentativa o programa deverá indicar se o número é maior, menor ou se acertou no número. No caso de o utilizador acertar, o programa deve indicar quantas tentativas necessitou para tal.
   b) Altere o programa de modo a que seja possível controlar o número máximo de tentativas. O programa deverá pedir inicialmente esse valor, efetuando a respetiva validação de entrada.

10. Dígitos de um número

    a) Desenvolva um programa que, dado um número inteiro fornecido pelo utilizador, calcule o número de vezes que surge o algarismo zero nesse número.
    Exemplo: se o utilizador introduzir o número 3010, o programa deverá indicar que existem 2 zeros no número fornecido.
    b) Altere o programa anterior de modo a que este, para além de indicar quanto zeros existem no número, forneça também a indicação da posição dos mesmos.
    Exemplo: se o utilizador introduzir o número 3010, o programa deverá indicar que existem 2 zeros no número fornecido, nas posições 1 e 3.

11. Pirâmide de Asteriscos

Escreva um programa que coloque no ecrã meia árvore de Natal com asteriscos. O número de ramos deverá ser introduzido pelo utilizador.
Exemplos com 3, 4 e 5 ramos:

```
*
**
***
```

```
*
**
***
****
```

```
*
**
***
****
*****
```

12. Pirâmide de números

a) Implemente um programa que imprima no ecrã a seguinte pirâmide de números:

```
1
1 2 1
1 2 3 2 1
1 2 3 4 3 2 1
1 2 3 4 5 4 3 2 1
```

b) Altere o programa de modo a que o número de linhas da pirâmide seja fornecido pelo utilizador, efetuando a devida validação dos dados da entrada.