

# UT 02 – Estructuras de control – Iterativas – Ejercicios

## Ejercicio 02 – 21

Realiza un programa que use un bucle for y muestre todos los números del 10 al 20, sin incluir el 20. Una línea por cada número.

## Ejercicio 02 – 22

Realiza un programa que use un bucle while y muestre todos los números del 10 al 20, sin incluir el 20. Una línea por cada número.

## Ejercicio 02 – 23

Realiza un programa que:

- Pida al usuario un número inicial (entero)
- Pida al usuario un número final (entero)
- Si el número inicial no es menor o igual al número final, mostrará un mensaje de error.
- En caso contrario, usando un bucle bucle for, muestre todos los números desde el inicial al final, separados por espacios, ambos incluidos.

## Ejercicio 02 – 24

Repite el programa del problema 24, pero usando un bucle while.

## Ejercicio 02 – 25

Haz un programa que permita escribir la tabla de multiplicar de un número que se pregunte al usuario.

Ejemplo de tabla de multiplicar generada por el programa para el número 3:

```
3x1=3
3x2=6
3x3=9
3x4=12
3x5=15
3x6=18
3x7=21
3x8=24
3x9=27
3x10=30
```

## Ejercicio 02 – 26

Algoritmo que pida números hasta que se introduzca un cero. Debe mostrar cada uno de los números introducidos, hasta que el usuario introduzca el cero, a medida que los vaya introduciendo. El cero, que es el número con el que el usuario “corta” el programa, no debe mostrarse.

## Ejercicio 02 – 27

Realizar un programa que:

- Pregunte al usuario cuántos números vamos a procesar.
- Pida al usuario la cantidad de números que ha introducido en el paso anterior, y para cada número indique si el número es menor que cero, cero o mayor que cero.

## Ejercicio 02 – 28

Hacer un programa igual que el del ejercicio 27, pero en lugar de mostrar un mensaje por cada número introducido, mostrará cuántos números positivos, cuántos negativos y cuántos ceros se han introducido.

## Ejercicio 02 – 29

Crea un programa que pida números enteros positivos hasta que se introduzca un cero. Debe calcular la suma y la media de todos los números introducidos. Si el usuario introduce un número menor que cero, debe mostrar un mensaje indicando que no es válido y no tenerlo en cuenta para el cálculo.

## Ejercicio 02 – 30

Haz un programa que muestre la tabla de multiplicar de los números 1,2,3,4 y 5. Hazlo usando un bucle anidado, un bucle dentro de otro.

## Ejercicio 02 – 31

Crea una aplicación que permita adivinar un número. La aplicación genera un número aleatorio del 1 al 99. A continuación, va pidiendo números y va respondiendo si el número a adivinar es mayor o menor que el introducido, además de informarle de los intentos que le quedan (tiene 10 intentos para acertarlo). El programa termina cuando se acierta el número (además te dice en cuantos intentos lo has acertado), si se llega al límite de intentos te muestra el número que había generado.

Para generar un número aleatorio, podemos usar un objeto de la clase Random. También se puede usar el método Math.random(), pero es algo más complicado que con la clase Random.

### Ejercicio 02 – 32

Una persona adquirió un producto para pagar en 20 meses. El primer mes pagó 10 €, el segundo 20 €, el tercero 40 € y así sucesivamente (cada mes, el doble que el mes anterior). Realizar un algoritmo para determinar cuánto debe pagar mensualmente y el total de lo que pagará después de los 20 meses.

### Ejercicio 02 – 33

Escribe un programa que diga si un número introducido por teclado es o no primo. Un número primo es aquel que sólo es divisible entre él mismo y la unidad.

La estrategia más habitual es comprobar si el número es divisible por algún número menor que él, además de el 1. Para optimizar el algoritmo, basta con probar con los números desde 2 hasta la raíz cuadrada del número que estamos probando.

### Ejercicio 02 – 34

Programa que muestra en pantalla los N primeros números primos. Se pide por teclado la cantidad de números primos que queremos mostrar.

### Ejercicio 02 – 35

Realizar un ejemplo de menú, donde podemos escoger las distintas opciones hasta que seleccionamos la opción de “Salir”. El menú podría ser algo similar a esto:

1. Opción A
2. Opción B
3. Opción C
4. Opción D
9. Salir

Cuando el usuario elige una opción, el programa mostrará “Se ha elegido A”, o “Se ha elegido B”, etc. Cuando elija la opción “Salir” el programa mostrará “Gracias por usar este menú sin utilidad”.

### Ejercicio 02 – 36

Escribe un programa que pida el límite inferior y superior de un intervalo (números enteros). Si el límite inferior es mayor que el superior lo tiene que volver a pedir, y seguir haciéndolo hasta que el usuario lo haga correctamente. A continuación, se van introduciendo números enteros hasta que introduzcamos el 0. Entonces, el programa dará las siguientes informaciones:

- La suma de los números que están dentro del intervalo. Intervalo abierto, no incluye los límites.
- Cuántos números están fuera del intervalo. Como el intervalo es abierto, los números fuera del intervalo incluyen los límites.

- Informa si hemos introducido algún número igual a los límites del intervalo (sólo si ha habido alguno o no).

### Ejercicio 02 – 37

Escribe un programa que:

- Pregunte al usuario una frase, formada por varias palabras.
- Pregunte al usuario la palabra que quiere buscar en la frase.
- Localice TODAS las apariciones de la palabra buscada, teniendo en cuenta que no se deben distinguir mayúsculas y minúsculas.
- Por cada aparición de la palabra buscada, mostrará la posición en la frase.
- Si la palabra buscada no aparece ninguna vez, no mostrará nada.
- Antes de salir, mostrará “Fin del programa”.

### Ejercicio 02 – 38

Escribe un programa que pregunte una frase al usuario. Mostrará cuántas vocales hay en la frase introducida por el usuario. Se considerarán vocales las siguientes letras:

- Vocales minúsculas y mayúsculas: a, e, i, o, u, A, E, I, O, U.
- Vocales minúsculas y mayúsculas con tilde: á, é, í, ó, ú, Á, É, Í, Ó, Ú.
- La u con diéresis, tanto minúscula como mayúscula: ü, Ü.

Aunque hay varias formas de hacerlo, unas pistas para una de las formas más flexibles, que permite incorporar ciertos cambios prácticamente sin tocar código:

- Se puede recorrer la cadena introducida por el usuario con un bucle, extrayendo caracteres con `charAt`.
- Se puede crear una constante con todas las letras que consideramos vocales válidas.
- Se puede usar `indexOf` en la constante para saber si un carácter es una vocal válida.

### Ejercicio 02 – 39

Crea un programa que pida una cadena de texto al usuario. El programa creará otra cadena de texto en el que estarán todas las letras de la cadena original, excepto las vocales. Se consideran vocales las mismas letras que en el problema 04-38.

### Ejercicio 02 – 40

Realizar un programa que muestre todos los números del 1 al 120, en filas de 10 números. Algo parecido a esto:

```
1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60
```

```
61 62 63 64 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90
91 92 93 94 95 96 97 98 99 100
101 102 103 104 105 106 107 108 109 110
111 112 113 114 115 116 117 118 119 120
```

En esta matriz de números se deben generar con un color distinto los números primos. Ya hicimos código para saber si un número es primo o no en otro problema.

Para escribir en consola con un color distinto, puedes mirar los siguientes artículos:

- <https://www.campusmvp.es/recursos/post/como-cambiar-los-colores-de-la-consola-con-java-y-system-out-println.aspx> (en castellano)
- <https://www.baeldung.com/java-log-console-in-color> (en inglés)

También tendrás que buscar cómo escribir los números con una longitud igual, todos de tres caracteres de ancho y con relleno de blancos delante, para que se alineen correctamente.