UNIVERSIDAD NEBRIJA

PROGRAMACIÓN I - GRUPO C

PRÁCTICA Nº5

19 de diciembre 2023

INSTRUCCIONES

1. Se debe acceder a la **SALA INDIVIDUAL** del campus virtual de la asignatura de Programación I de prácticas, hay una sala para cada integrante.

2. Durante el desarrollo de la sesión práctica se debe seguir el procedimiento de trabajo en la sala asignada para realizar las actividades evaluables. Recordad que es **imprescindible GRABAR toda la sesión** y **COMPARTIR la pantalla completa**. Esto debe hacerlo cada integrante del equipo

3. La entrega de la práctica sólo se admitirá a través de la **actividad** disponible en el campus virtual de la asignatura de **Prácticas de Programación I antes de la hora de finalización de la sesión de prácticas**.

4. Se debe entregar **un único fichero** en **formato .txt o .cpp, sin comprimir**, con el nombre de ***P5GC.***

5. El fichero entregado debe **incluir el nombre de los integrantes del equipo**.

6. Aunque las prácticas se realizan en grupos de dos integrantes, para su evaluación, **ambos deberán hacer la entrega a través de su campus virtual**. En otro caso, la práctica quedará **sin evaluar** y supondrá un **0 en su calificación**.

7. **El incumplimiento de alguna de las instrucciones sobre la realización/entrega de la Práctica supondrá su descalificación.**

EJERCICIO 1 (3,5 puntos)

Escriba un programa en C++11 con el nombre ***P5E1GC*** que, utilizando los **tipos de datos, las operaciones de entrada y salida, las estructuras de control** y **las funciones** adecuadas para estructurar bien el proceso, permita determinar si una matriz solicitada al usuario es ***triangular*** y qué ***tipo*** es, ***triangular superior izquierda*** o ***triangular inferior derecha***.

Importante: El ejercicio debe ser resuelto de **forma algorítmica**, es decir, la solución propuesta tiene que ser general y no particular para unos determinados datos.

Una ***matriz triangular, izquierda o derecha,*** es una **matriz cuadrada** en la que todos los elementos por encima o por debajo de la **diagonal secundaria** son ***iguales a cero***. La ***matriz triangular inferior izquierda*** es una matriz cuadrada cuyos elementos por encima de la diagonal secundaria son ***cero***. La ***matriz triangular superior derecha*** es una matriz cuadrada cuyos elementos por debajo de la diagonal secundaria son ***cero***.

El proceso debe realizar las siguientes tareas:

- Solicitar al usuario números enteros para crear una matriz cuadrada de dimensión definida a través del **valor de una constante declarada** en la función ***main*** del programa. Así, si se decide cambiar la dimensión de la matriz sólo será necesario modificar dicha constante **[1 punto]**.
- Determinar si dicha matriz es **triangular superior izquierda o inferior derecha o nada [1,5 puntos]**.
- Mostrar por pantalla el contenido de la matriz cuadrada indicando si es una matriz **triangular superior izquierda o inferior derecha o no es triangular [1 punto]**.

A continuación, se muestran algunos ejemplos de ejecución:

```
DATOS DE LA MATRIZ
==================
Elemento[1, 1] 1

Elemento[1, 2] 0

Elemento[1, 3] 0

Elemento[2, 1] 0

Elemento[2, 2] 0

Elemento[2, 3] 0

Elemento[3, 1] 0

Elemento[3, 2] 0

Elemento[3, 3] 4


La matriz

        | 1 0 0 |
        | 0 0 0 |
        | 0 0 4 |

 NO es triangular.
```

```
DATOS DE LA MATRIZ
==================
Elemento[1, 1] 0

Elemento[1, 2] 0

Elemento[1, 3] 1

Elemento[2, 1] 0

Elemento[2, 2] 2

Elemento[2, 3] 0

Elemento[3, 1] 3

Elemento[3, 2] 0

Elemento[3, 3] 0


La matriz

        | 0 0 1 |
        | 0 2 0 |
        | 3 0 0 |

 es triangular INFERIOR y SUPERIOR.
```

```
DATOS DE LA MATRIZ
==================
Elemento[1, 1] 0

Elemento[1, 2] 0

Elemento[1, 3] 1

Elemento[2, 1] 0

Elemento[2, 2] 2

Elemento[2, 3] 3

Elemento[3, 1] 4

Elemento[3, 2] 5

Elemento[3, 3] 6


La matriz

        | 0 0 1 |
        | 0 2 3 |
        | 4 5 6 |

es triangular INFERIOR IZQUIERDA.
```

```
DATOS DE LA MATRIZ
==================
Elemento[1, 1] 1

Elemento[1, 2] 2

Elemento[1, 3] 0

Elemento[2, 1] 1

Elemento[2, 2] 2

Elemento[2, 3] 0

Elemento[3, 1] 1

Elemento[3, 2] 0

Elemento[3, 3] 0


La matriz

        | 1 2 0 |
        | 1 2 0 |
        | 1 0 0 |

es triangular SUPERIOR DERECHA.
```

EJERCICIO 2 (3,5 puntos)

Escriba un programa en C++11 con el nombre ***P5E2GC*** que, utilizando los **tipos de datos, las operaciones de entrada y salida, las estructuras de control** y **las funciones** adecuadas para estructurar bien el proceso, permita determinar los números enteros mayores que cero que son **perfectos en el intervalo [1..500].** Concretamente, se pide obtener la relación de números perfectos en el intervalo indicado almacenando en el tipo contenedor que considere mejor, elementos de tipo ***struct*** que incluyan el número entero perfecto y junto a sus divisores propios.

Importante: El ejercicio debe ser resuelto de **forma algorítmica**, es decir, la solución propuesta tiene que ser general y no particular para unos determinados datos.

Se dice que un ***número entero es perfecto*** si la suma de sus ***divisores propios*** es igual al propio número. mayores que cero son ***números amigos*** si la suma de los **divisores propios** de cada uno es igual al otro número. Por ejemplo, el número 6:

- Divisores propios de **6**: 1, 2 y 3 => suma divisores: **6**

Un **divisor propio** de un número es un factor positivo de dicho número que no sea el propio número. Por ejemplo, los divisores propios del 6 son 1, 2 y 3, pero no el 6.

El proceso debe realizar las siguientes tareas:

- Procesar todos los números del intervalo [1..500] para determinar si un número es **perfecto** realizando las siguientes operaciones **[0,5 puntos]**:
  - Determinar la relación de divisores propios de un número y los almacene en algún tipo de contenedor **[0,5 puntos]**.
  - Calcular la suma de los divisores propios de un número almacenados en algún tipo de contenedor **[0,5 puntos]**.
  - Comprobar si el número es perfecto **[0,5 puntos]**.
  - Almacenar el elemento **struct** con el número perfecto y sus divisores propios en el contenedor **[0,5 puntos]**.
- Mostrar por pantalla la relación de divisores propios de cada número perfecto encontrado en el intervalo [1..500] **[1 punto]**.

A continuación, se muestran algunos ejemplos de ejecución:

```
Los divisores del numero 6 son:  1 2 3
Los divisores del numero 28 son:  1 2 4 7 14
Los divisores del numero 496 son:  1 2 4 8 16 31 62 124 248
```

RUBRICA DE CALIFICACIÓN

- El 30% de la nota de las prácticas, se obtendrá del trabajo en clase (resolución de preguntas, dominio de la materia, etc.). **Esta nota será individual**. Se consideran aspectos como:
  - Defensa **individual** a cuestiones que el/la profesor/a plantea a cada integrante del equipo.
  - Debate en **equipo** sobre la estrategia a seguir, realizando un **boceto de la estructura** que tendrá cada ejercicio propuesto en la práctica (a papel, **dicho boceto debe entregarse al final de la práctica**).
- El 70% restante será la implementación de los ejercicios de la práctica (**misma nota para cada integrante del equipo**), siguiendo la rúbrica de evaluación general y la siguiente rúbrica para cada apartado:

Concepto	Puntos
Ejercicio nº1.-	
- Función para solicitar los números enteros para crear la matriz cuadrada con la dimensión establecida como constante.	1 punto
- Función para determinar si es una matriz triangular superior o inferior o nada.	1,5 puntos
- Función para mostrar por pantalla la matriz y el mensaje correspondiente.	1 punto
Ejercicio nº2.-	
- Función para procesar los números del intervalo indicado.	0,5 puntos
- Función que:	
o Determine la relación de divisores propios de un número.	0,5 puntos
o Determine la suma de divisores propios de un número.	0,5 puntos
o Compruebe si el número es perfecto.	0,5 puntos
o Almacene cada número perfecto y sus divisores propios como elemento de un contenedor.	0,5 puntos
- Funciones para mostrar por pantalla el contenido del contenedor de números perfecgtos.	1 punto

Se valorará la escritura correcta del código, con las **tabulaciones** adecuadas, **nombres significativos** para las variables, **comentarios**, etc.

Además, cada apartado de la parte práctica, se evaluará considerando el siguiente baremo:

	% máx. (*)
El elemento evaluable no compila o no se asemeja a lo que se pide	0%
El elemento evaluable no se aproxima suficientemente a lo pedido	40%
El elemento evaluable se aproxima suficientemente a lo pedido	60%
El elemento evaluable funciona correctamente y las estrategias y elementos de código elegidos son adecuados.	100%

(*) El porcentaje (% max.) representa el valor máximo sobre el que se evalúa el elemento indicado.