UNIVERSIDAD NEBRIJA

ALUMNO:

EPS -Ingeniería Informática

Asignatura: G0460006 Programación I

Curso: 2022/2023 Examen: Final Fecha: 16-1-2023
Semestre: 1º Convocatoria: Ordinaria

Parte Práctica (10 puntos; 70% nota final) - Tiempo: 2h40m

Los ficheros correspondientes a los ejercicios se deben entregar en la actividad correspondiente a través del campus antes de la finalización del tiempo establecido. Cada fichero se llamará EjercicioX, donde la X será el número de dicho ejercicio. Los únicos formatos válidos serán **txt** o **cpp**, siempre **sin comprimir**.

Criterios generales de evaluación

Funciones/Métodos: Si no se usa el paso por referencia constante cuando las variables de los parámetros de entrada no son de tipo simple.	40%
Tipos de datos y variables: • Uso de variables globales (fuera del ámbito de una función). • Si no se usan los tipos contenedor vistos en clase (std::array; std::vector; std::set; std::string, etc.) para las variables que lo necesiten.	 0% 0%
El **programa no compila** o **no se asemeja** a lo pedido**.**	0%
Si no se cumplen los **criterios de entrega** indicados en la **actividad/examen.**	0%

Criterios particulares de evaluación

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%

IMPORTANTE:

- Todos los ejercicios del examen deberán ser resueltos de forma **algorítmica**, es decir, la **solución** propuesta tendrá que ser **general** y **no particular** para unos determinados datos/valores.

- Todos los ejercicios resueltos sin utilizar funciones cuando sea apropiado se valorarán con una nota máxima del 60% sobre la calificación prevista.

UNIVERSIDAD NEBRIJA

- Se recomienda una primera lectura del examen completo para planificar la realización del examen. Y una segunda lectura detallada antes de la realización de cada uno de los ejercicios propuestos.

Ejercicio 1 (3 puntos)

Escriba un programa en C++11, ***Ejercicio1***, utilizando los tipos de datos, las **estructuras de control** y las **funciones** necesarias para determinar si dos números enteros mayores que cero solicitados al usuario son **amigos.**

Se dice que dos números enteros mayores que cero son ***números amigos*** si la suma de los **divisores propios** de cada uno es igual al otro número. Por ejemplo, los números 220 y 284 son amigos:

- Divisores propios de **220**: 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 y 110 => suma divisores: 284
- Divisores propios de **284**: 1, 2, 4, 71 y 142 => suma divisores: 220

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:

- Solicitar al usuario dos números enteros validando que son mayores que cero y distintos entre ellos **(0, 5 puntos)**.
- Determinar si dichos números son **amigos** realizando las siguientes operaciones:
    - Determinar la relación de divisores propios de cada número **(0,75 puntos)**.
    - Calcular la suma de los divisores propios de cada número **(0,25 puntos)**.
    - Comprobar si los números son amigos **(0,25 puntos)**.
- Mostrar por pantalla la relación de divisores propios de cada número y un mensaje indicando si los números son amigos o no **(0,75 puntos)**.
- Finalmente, se preguntará al usuario si desea repetir el proceso con otros dos números o finalizar la ejecución **(0,5 puntos)**.

A continuación, se muestran algunos ejemplos de ejecución:

```
Introduzca un numero entero positivo y mayor que 0: 284


Introduzca un numero entero positivo y mayor que 0: 215


Los numeros 284 y 215 NO son AMIGOS.
Los divisores del numero 284 son:  1 2 4 71 142
Los divisores del numero 215 son:  1 5 43

Desea repetir el proceso con otros datos (S/N)? s
```

```
Introduzca un numero entero positivo y mayor que 0: 220


Introduzca un numero entero positivo y mayor que 0: 110


El numero 110 debe ser mayor que el numero 220
Por favor, introduzca el segundo numero de nuevo.

Introduzca un numero entero positivo y mayor que 0: 284


Los numeros 220 y 284 son AMIGOS.
Los divisores del numero 220 son:  1 2 4 5 10 11 20 22 44 55 110
Los divisores del numero 284 son:  1 2 4 71 142

Desea repetir el proceso con otros datos (S/N)? s
```

```
Introduzca un numero entero positivo y mayor que 0: -26

El valor del numero no es valido. Vuelva a intentarlo.

Introduzca un numero entero positivo y mayor que 0: 284


Introduzca un numero entero positivo y mayor que 0: 284


El numero 284 debe ser distinto al primero.
Por favor, introduzca el segundo numero de nuevo.

Introduzca un numero entero positivo y mayor que 0: 220


Los numeros 284 y 220 son AMIGOS.
Los divisores del numero 284 son:  1 2 4 71 142
Los divisores del numero 220 son:  1 2 4 5 10 11 20 22 44 55 110

Desea repetir el proceso con otros datos (S/N)? s
```

UNIVERSIDAD NEBRIJA

Ejercicio 2 (3 puntos)

Diseñar e implementar un programa C++11 que a partir de dos cadenas de texto compruebe si la segunda está contenida en la primera, sin usar el método ***find()*** del tipo ***std::string***.

Por ejemplo: para la cadena "**Estoy enamorado del mar**" y "**amor**". La segunda cadena está contenida en la primera ya que **enamorado** contiene la cadena "**amor**".

El programa imprimirá por pantalla si la **segunda cadena se encuentra contenida en la primera o no**, y en caso afirmativo indicará la **posición de la primera letra** y la **posición de la última letra**.

Para este ejercicio las calificaciones parciales serán asignadas en función de los siguientes subapartados:

- Leer un texto cualquiera (conjunto de palabras) y leer una palabra almacenando la información en las variables del tipo que consideres oportuno **(0,5 puntos)**
    - Alternativamente se podrán definir dos variables tipo ***std::string*** con las cadenas con la información asignada en la declaración. En ese caso este subapartado tendrá una puntuación de 0 puntos.
- Si el programa localiza la cadenada adecuadamente **(2 puntos)**
- Si el programa identifica las posiciones adecuadamente **(0,5 puntos)**

UNIVERSIDAD NEBRIJA

Ejercicio 3 (4 puntos)

En este ejercicio deberás desarrollar un programa en C++11 para calcular y visualizar elementos de dos series numéricas $\{a_0\ a_1\ a_2\ a_3\ldots\}$ y $\{b_0\ b_1\ b_2\ b_3\ldots\}$.

Dicho programa debe preguntar al usuario de forma cíclica por el número de términos que desea visualizar y por sus dos primeros valores. Teniendo en cuenta que para el cálculo de los términos de ambas series utilizaremos los **mismos valores iniciales** ($\mathbf{a_0=b_0}$ y $\mathbf{a_1=b_1}$), y que el **número de términos será idéntico para ambas**.

Las series cuyos valores se desean visualizar por el terminal son:

- $a_n = a_{n-1}+a_{n-2}$
- $b_n = n*((b_{n-1})/2 + b_{n-2})$

De tal manera que si, por ejemplo, los valores iniciales para el término a_0 y b_0 fuera 0; el de a_1 y b_1 fuera 1 y quisiéramos obtener 8 valores de ambas series tendríamos los siguientes resultados.

n	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**	**[6]**	**[7]**
Serie a	0	1	1	2	3	5	8	13
Serie b	0	1	1	4,5	13	55	243	1235,5

Para ello **es necesario que escojas "*UNA*" de las dos opciones siguientes**. Teniendo en cuenta la **penalización en la nota del ejercicio que supone la segunda opción frente a la primera** (en lugar de 4 puntos, el ejercicio puntuará sobre 2 puntos):

- **Opción utilizando punteros**: Implementar una función que devuelva un puntero a una variable contenedor del tipo que consideres oportuno que almacene los términos calculados de ambas series **(2 puntos)**
- **Opción sin punteros**: Implementar una función que devuelva una variable contenedor del tipo que consideres oportuno que almacene los términos calculados para ambas series **(1 punto)**

En la función principal deberás implementar el código necesario para llevar a cabo la tarea indicada en el comienzo del enunciado hasta que el usuario decida terminar la ejecución del programa. La puntuación de este subapartado será:

- En caso de implementar la **opción con punteros (2 puntos)**
- En caso de implementar la **opción sin punteros (1 punto)**