UNLaM
Dto. Ingeniería e Investigaciones Tecnológicas

(0612) PROGRAMACIÓN II
(1110) PROGRAMACIÓN
FINAL
FEBRERO 2022 - MESA A
Sábado 26 /02/2022 09 hs.

Apellido y Nombre:

DNI:

Calificación :

Ejercicio C

Archivo 1:

- *Nombre*: “mat1.txt”
- *Tipo*: Texto
- *Contenido*: los elementos no nulos de una matriz dispersa (Ver NOTA 1).
- *La primera línea* del archivo indica las dimensiones de la matriz de la forma:
  [filas][columnas]
  Los corchetes aparecen de forma literal en el archivo de texto.
- *El resto de las líneas:* indica los valores no nulos de la matriz, el formato es el siguiente;
  [fila][columna] valor no nulo
  Estas líneas no tienen ningún orden.

Archivo 2:

- *Nombre*: “mat2.txt”
- *Tipo*: Texto
- *Contenido*: los elementos no nulos de una matriz dispersa (Ver NOTA 1).
- *La primera línea* del archivo indica las dimensiones de la matriz de la forma:
  [filas][columnas]
  Los corchetes aparecen de forma literal en el archivo de texto.
- *El resto de las líneas:* indica los valores no nulos de la matriz, el formato es el siguiente;
  [fila][columna] valor no nulo
  Estas líneas no tienen ningún orden

Archivo 3:

- *Nombre*: “suma.txt”
- *Tipo*: Texto
- *Contenido*: los elementos no nulos de una matriz dispersa (Ver NOTA 1).
- *La primera línea* del archivo indica las dimensiones de la matriz de la forma:
  [filas][columnas]
  Los corchetes aparecen de forma literal en el archivo de texto.
- *El resto de las líneas:* indica los valores no nulos de la matriz, el formato es el siguiente;
  [fila][columna] valor no nulo
  Estas líneas no tienen ningún orden

Matriz Dimensiones:

- 0 < filas < 1.000
- 0 < columnas < 1.000

NOTA: Leer la nota al final del archivo

Objetivo propuesto: desarrollar una aplicación, main y la/s función/es necesarias, para sumar los elementos de 2 matrices dispersas (Archivo 1 y Archivo 2). Luego mostrar la matriz resultado, mostrando tanto los elementos nulos como los no nulos. Guardar el resultado en otro archivo (Archivo 3). Su solución **debe optimizar el uso de memoria** y hacer uso de las siguientes funciones:

Funciones:
main: debe llevar a cabo el objetivo propuesto, optimizando el uso de memoria y hacer uso de las funciones indicadas a continuación. En caso de necesitar más funciones puede agregarlas.

verificar_mat: esta función **verifica si es posible alcanzar el objetivo propuesto**. Esta función debe invocarse antes de cargar las matrices en memoria.

cargar_mat: función para cargar una matriz en memoria.
Carga en memoria una matriz dispersa de forma tal de poder realizar la suma de matrices. Usted debe seleccionar la estructura de datos más conveniente. Debe optimizar el uso de memoria.

sumar_mat: función que realiza la suma de dos matrices. Optimizar el uso de memoria.

imprimir_mat: recibe una matriz dispersa y la muestra completa (con todos los valores, incluyendo los valores nulos).

guardar_mat: recibe una matriz dispersa y la guarda en un archivo de texto, respetando el formato solicitado. Luego de ejecutada esta función ya no estará disponible la matriz.

NOTA:
- Una **matriz rala** o **matriz dispersa** o **matriz hueca** es una matriz de gran tamaño en la que la mayor parte de sus elementos son cero.
- En caso de necesitar alguna otra función puede hacerlo, pero DEBE DESARROLLARLA completa.
    -

Ejercicio C++

Implemente la clase VectorEnteros en C++ tal que el main que se encuentra al pie sea válido.
Analice la salida por consola para determinar el funcionamiento de los operadores.
A tener cuenta:

1. Sea cuidadoso con el manejo de la memoria.
2. Aunque su compilador lo permita, no utilice VLE (Variable Length Array).
    a. Ej: “int vec[x]” donde x es determinado en tiempo de ejecución.
3. Sea cuidadoso de su tiempo y no desarrolle ningún método que no sea necesario para que el main propuesto funcione correctamente.
4. No suponga ningún tamaño máximo para los vectores.
5. Informe cualquier situación anormal con una excepción.

```
int main()
{
    int ve1[]       = {1,2,3,4,5,6,7,8};
    int ve2[]       = {9,10,11,12,13};
    VectorEnteros v1;
    VectorEnteros v2(ve2, sizeof(ve2)/sizeof(ve2[0]));
    VectorEnteros v3;
    cout<<v1.agregar(ve1, sizeof(ve1)/sizeof(ve1[0]))<<endl;
    v3 = v1+v2;
    v3 = v3+14;
    cout<<v3<<endl;
    return 0;
}
```

Salida esperada:

```
[1, 2, 3, 4, 5, 6, 7, 8]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]

Process returned 0 (0x0)   execution time : 0.106 s
Press any key to continue.
```

EVALUACIÓN PRESENCIAL

CONDICIÓN PARA APROBAR

- Para alcanzar el 4 debe:
  - **EJERCICIO DE C**
    - Desarrollar **correctamente** la función **cargar_mat,** de forma de hacer un uso óptimo de la memoria.
    - Desarrollar correctamente la función **imprimir_mat.**
    - **Desarrollar un main que únicamente cargue la matriz dispersa y la muestre.**
  - **EJERCICIO DE C++**
    - Debe desarrollar correctamente al menos el 60% del ejercicio completo.

NOTA GENERAL

- Desarrolle cada ejercicio en un proyecto separado.
- **Incluya en el encabezado de cada archivo, // apellido_nombre_DNI**
- Recuerde antes de comprimir, eliminar las carpetas bin y obj de cada proyecto.
- **Entregue cada proyecto compactado en un zip, "Ej_nro_apellido_nombre_DNI.zip".**
- Entregue el parcial usando prácticas de MIEL, cada proyecto en la práctica correspondiente C o C++.
- NO Enviar a los tutores.

¡El mayor de los éxitos!