FICH-UNL. Programación Orientada a Objetos. 2do Parcial - Tema A - 10/11/2023

Ejercicio 1 (30 ptos)

a) Diseñe una función genérica promedio(...) que reciba una lista (list) de datos de cualquier tipo y calcule el promedio de los elementos.

b) Escriba un programa en C++ que sea un cliente de la función promedio(...). El programa debe invocar la función tres veces para las siguientes listas:

i) Una lista de números flotantes ingresados por el usuario desde el teclado.

ii) Una lista de números enteros generados aleatoriamente en el rango de 1 a 100.

iii) Una lista de números complejos (`struct Complejo { float p_real, p_imag; }`) que deben leerse desde un archivo binario "complejos.dat". Nota: implemente todas las funciones auxiliares que considere necesarias para que el programa funcione.

Ejercicio 2 (30 ptos)

Se dispone de un archivo de textos llamado *"datos.txt"* generado por un equipo que mide una vez al día la altura del río. Para simplificar, suponer que todos los meses tienen 30 días; entonces el archivo tendrá 30*12=360 datos.

a) Escriba un programa C++ que lea el archivo y organice todos los datos en un único contenedor STL.

b) Obtenga el mayor valor de cada mes, y genere con estos 12 valores un nuevo vector donde cada posición contenga una terna de datos: nro de mes, valor máximo de ese mes, y cuántas veces se repitió ese valor máximo en el mes.

c) Ordene el nuevo vector de mayor a menor (según el valor máximo de cada terna) y luego guarde estas 12 ternas en un archivo binario "*máximos.dat".*

Ejercicio 3 (30 ptos)

Escriba una clase Cola para representar una cola de trabajos pendientes. La clase debe ser genérica: el tipo de struct utilizado para representar un trabajo debe ser el argumento *T* de la plantilla. Asuma que el struct *T* con el que se especialice la plantilla siempre tendrá un campo entero id. Por ejemplo, el programa cliente podría definir:

```
struct TrabajoImpresion { int id; bool color; int cant_pags; string archivo; };
Cola<TrabajoImpresion> cola_impresion; // gestiona las impresiones pendiente
```

La clase Cola debe guardar internamente una secuencia de trabajos y tener métodos para:

- Agregar un trabajo: el que se agrega siempre va como último trabajo, y se le asigna un nuevo id. El primer trabajo tendrá el id 1, el 2do el 2, el 3ro el 3, y así sucesivamente. Ayuda: guarde el último id utilizado como atributo en la clase Cola para saber cuál sigue al agregar.
- Obtener el próximo trabajo a realizar: el método debe quitar de la lista al primer trabajo y retornarlo.
- Cancelar un trabajo según su id: debe buscar el trabajo por su id y eliminarlo. Retornar true si lo encontró, false en caso contrario.
- Obtener la cantidad de trabajos.
- Obtener todos los datos de un trabajo según su posición.

Ejercicio 4 (10 ptos) Explique/justifique:

a) ¿Por qué no está disponible el operador [ ] en la clase contenedora list de la STL?

b) Si un aparato de medición genera un archivo de texto y Ud debe procesar los datos creando un programa C++ pero desconoce la disposición de los datos en el archivo, ¿qué sugiere hacer para conocer cómo están dispuestos los datos en las líneas del archivo?

c) ¿Siempre es posible almacenar una instancia de una clase o struct en un archivo binario?

FICH-UNL. Programación Orientada a Objetos. 2do Parcial - Tema B - 10/11/2023

Ej 1 (30 ptos) Se tiene un archivo de textos "*listado.txt*" con un nombre y apellido en cada línea de los alumnos inscriptos a mátemática. En una revisión se determinó que el total de datos superaban los 300 (cantidad que debían ser) debido a que estos datos fueron pasados por distintas personas y se habían pasado varias veces los datos de un mismo alumno. Se desea procesar el listado de tal forma que no haya datos repetidos. **a)** Escriba un programa que cargue los datos en un vector de string y elimine los repetidos. **b)** Además se les quiere proporcionar email institucional a cada uno, el cual se debe generar a partir de la primer letra del nombre, el apellido completo y el dominio "@fich.unl.edu.ar" (suponer que en cada string inicial hay un solo nombre y un solo apellido separados por espacio). **c)** Por último, debe generar 5 archivos texto llamados "comision1.txt", "comision2.txt",.... "comision5.txt", con los datos de 60 alumnos cada uno de la siguiente manera: apellido, nombre, correo (uno por linea, las tres partes separadas por coma). Por ejemplo si el alumno se llama "Juan Perez", en el archivo se deberá guardar "Perez, Juan, jperez@fich.unl.edu.ar".

Ej 2 (30 ptos) Un estudio de juegos *indie* te contrata como programador. Tu primer trabajo es programar una clase llamada TDP (abreviatura de "Tabla De Puntajes") que le será de utilidad al estudio para sus proyectos actuales y próximos. El estudio te solicita que la clase maneje la TDP de un juego en particular, la cual se guardará en la base de datos del estudio como un archivo binario con el nombre "puntuaciones_juego.dat" donde "juego" se reemplaza con el nombre del juego. Pero hay un problema: los distintos juegos de la empresa tienen distintos criterios de puntaje. Mientras que el juego *SPACE SHOOTERS* guarda sus puntajes como una lista de enteros, el juego *FROG RACING* guarda sus puntajes como floats, y es posible que los próximos lanzamientos de la empresa utilicen distintos criterios. Se sabe que no importa el criterio que usen, siempre se pueden comparar los puntajes para ver qué jugador obtuvo el mejor puntaje, y que los mismos se identifican a través de un atributo *nom* (que no es más que un arreglo de 5 chars, e identifica a un único jugador, por lo que dos puntajes se consideran iguales si tienen el mismo *nom*). Entonces, para demostrarle a la empresa que hicieron bien en contratarte:

- **a)** Implemente la clase **genérica** TDP que resuelva este problema para la empresa. La clase recibe en su constructor el nombre del juego, y debe cargar en memoria los datos de la tabla de puntuaciones del juego.
- **b)** También debe ser posible agregar puntajes. Si el jugador a agregar ya tenía un puntaje, debe actualizarlo (se guarda el último puntaje, no el mejor).
- **c)** Y debe poder mostrar una lista ordenada de los *nom*s de menor a mayor.
- **d)** Finalmente, la clase debe poseer una forma de actualizar el archivo binario de la base de datos (y debe actualizarse automáticamente cuando la clase se destruye).

Ej 3 (30 ptos) Escribir una clase genérica llamada OperaLista que posea como atributo una lista dinámica (list). a) La clase debe poseer métodos para construir la lista a partir de un vector que se pasa como parametro u métodos para b) Agregar un elemento c) Obtener la cantidad de elementos d) Ordenar la lista en forma decreciente y retornar cuantas veces se repite el mayor elemento. e) Eliminar un elemento de la lista cuya posición se pasa como parámetro f) un método para mostrar por pantalla los elementos de la lista.
Defina un programa cliente con 2 instancias de OperaLista empleando enteros y strings. Los datos deben leerse de los archivos de texto *listaenteros.txt* y *listastring.txt* respectivamente. El programa debe mostrar ambas listas, y cuantas veces se repite el mayor de cada una.

Ej 4 (ptos) a) Qué consideraciones hace al elegir una secuencia de la STL para organizar los datos de un programa? Es lo mismo usar vector que list?. b) Es posible abrir un archivo binario como archivo de texto? y viceversa?. explique. **c)** ¿Qué es un iterador en la STL? Si x está declarada como instancia de alguna de las secuencias de la STL. ¿Siempre es posible desplazarse con x.begin()+n para poder acceder al elemento de la posición n (n+1 ésimo) de x?