INF-134 Estructuras de Datos, 2022-2

Tarea 1

Profesores: Gabriel Carmona, Javier Robledo
mailto:gabriel.carmonat@sansano.usm.cl, javier.robledo@usm.cl

Ayudantes:
Joaquín Gatica (joaquin.gatica@sansano.usm.cl)
Rodrigo Flores (rodrigo.floresf@sansano.usm.cl)
Gabriel Escalona (gabriel.escalona@usm.cl)

Fecha de entrega: 16 de septiembre, 2022.
Plazo máximo de entrega: 5 días.

1. Reglas del Juego

La presente tarea debe hacerse en grupos de 3 personas. Toda excepción a esta regla debe ser conversada con los ayudantes (usando los correos indicados en el encabezado de esta tarea). No se permiten de ninguna manera grupos de más de 3 personas. Debe usarse el lenguaje de programación C++. Al evaluarlas, las tareas serán compiladas usando el compilador `g++`, usando la línea de comando `g++ archivo.cpp -o output -Wall`. **No se aceptarán variantes o implementaciones particulares de g++, como el usado por MinGW**. Se deben seguir los tutoriales que están en Aula USM, cualquier alternativa explicada allí es válida. Recordar que una única tarea en el semestre puede tener nota menor a 30. El no cumplimiento de esta regla implica reprobar el curso. No se permite usar la biblioteca *stl*, así como ninguno de los *containers* y algoritmos definidos en ella (e.g. `vector`, `list`, etc.). Está permitido usar otras funciones de utilidad definidas en bibliotecas estándar, como por ejemplo `math.h`, `string`, `fstream`, `iostream`, etc.

2. Objetivos

Comprender y familiarizarse con las estructuras y tipos de datos básicos que provee el Lenguaje de Programación C++. Entre los conceptos mas importantes, se encuentran:

- Paso de parámetros por valor.
- Paso de parámetros por referencia.
- Asignación de memoria dinámica.
- Manipulación de punteros.
- Manejo de Archivos.

Además se fomentará el uso de las buenas prácticas y el orden en la programación de los problemas correspondientes.

3. Problemas a Resolver

En esta sección deben implementarse funciones y clases en C++, de acuerdo a las siguientes descripciones. Se debe entregar cada uno de los problemas en archivos `.cpp` separados (y correspondientes `.hpp` de ser necesario). Para cada problema, entregue una función `main` adecuada que permita probar sus programas. Sin embargo, tenga en cuenta que durante la corrección se probarán otras funciones `main`.

3.1. Fastest Typist en INF-134

Los profesores de Estructuras de Datos desean que sus estudiantes incrementen su velocidad de tecleo durante la asignatura. Para promover esta actividad, le aconsejan a sus cursos ejercitar durante 30 minutos diarios durante un mes y registrar sus progresos. La velocidad de tecleo se mide en ppm (palabras por minuto), el cual es un valor entero. Además se mide la precisión de tecleo como un porcentaje que permite decimales (número real).
El registro de la información de estudiantes se mantiene en un archivo de texto, llamado `estudiantes.txt`, en donde por cada linea se tiene el rol, nombre, apellido y paralelo separados por espacio. A continuación un ejemplo del archivo:

```
202173571-K Pedro Alvarez 201
202173501-9 Antonia Soto 200
202173504-2 Fernanda Buendia 201
```

A su vez se tiene un archivo binario, llamado `registro.dat`, con n registros con la información de velocidad y precisión de cada estudiante en un día, mes y año determinado. Para este archivo considere la siguiente estructura:

```
struct Registro {
    int dia;
    int mes;
    int anio;
    char rol[12];
    int ppm;
    float precision;
};
```

El primer dato en el archivo binario es un entero que indica la cantidad de registros en el archivo binario.

Su programa debe leer la información de ambos archivos y recibir ciertas consultas por entrada estándar, estas consultas corresponden a preguntar dado una fecha y un atributo del registro cual fue el mejor tecleador/a. Esta sigue el siguiente formato:

- Un entero Q, que consiste a la cantidad de consultas a leer por entrada estándar.
- Cada consulta consiste del siguiente formato:

  `t d m a`

  - Un entero t, el cuál valdrá 0 si se quiere preguntar por precisión, o valdrá 1 si se quiere preguntar por ppm.
  - Un entero d, el cuál valdrá -1 o entre 1 al 31. Si vale -1 Significa que se deberá ignorar el día, sino será el valor que aparece.
  - Un entero m, el cuál valdrá -1 o entre 1 al 12. Si vale -1 Significa que se deberá ignorar el mes, sino será el valor que aparece.
  - Un entero a, el cuál valdrá entre 1 al 3000.

- Nota: m no puede valer -1 si d no vale -1.

El formato de la respuesta de cada consulta debe ser una línea con el nombre completo del alumno/a. Un posible input de la tarea es:

```
3
1 -1 2 2022
0 20 10 2
1 1 1 2598
```

El posible output de la tarea con el ejemplo anterior es:

```
Pedro Alvarez
Antonia Soto
Fernanda Buendia
```

4. Entrega de la Tarea

Entregue la tarea enviando un archivo comprimido `tarea1-apellido1-apellido2-apellido3.zip` o `tarea1-apellido1-apellido2-apellido3.tar.gz` (reemplazando sus apellidos según corresponda) a la página aula.usm del curso, a más tardar el día 16 de septiembre, 2022, a las 23:59:00 hs (Chile Continental), el cual contenga:

- Los archivos con los códigos fuentes necesarios para el funcionamiento de la tarea. ¡Los archivos deben compilar!
- **nombres.txt**, Nombre, ROL, Paralelo y qué programó cada integrante del grupo.
- **README.txt**, Instrucciones de compilación en caso de ser necesarias, y la forma de compilación que usó (debe ser alguna de las indicadas en los tutoriales entregados en Aula USM).

5. Restricciones y Consideraciones

- Por cada día de atraso en la entrega de la tarea se descontarán 10 puntos en la nota.
- El plazo máximo de entrega es 5 días después de la fecha original de entrega.
- **Las tareas que no compilen no serán revisadas y serán calificadas con nota 0**.
- Debe usar **obligatoriamente** alguna de las formas de compilación indicadas en los tutoriales entregados en Aula USM.
- Por cada *Warning* en la compilación se descontarán 5 puntos.
- Si se detecta **COPIA** la nota automáticamente sera 0 (CERO), para todos los grupos involucrados. El incidente será reportado al jefe de carrera.
- La prolijidad, orden y legibilidad del código fuente es obligatoria. Habrá descuentos si alguno de estos items no se cumple.

6. Consejos de Programación

El código fuente del programa debe estar estructurado adecuadamente en archivos (separados de ser necesario). Si el código fuente está desordenado, se pueden descontar hasta 20 puntos de la nota.

Cada función programada debe tener comentarios de la siguiente forma:

```
/*****
*   TipoFunción NombreFunción
******
*   Resumen Función
******
*   Input:
*       tipoParámetro NombreParámetro : Descripción Parámetro
*        .......
******
*   Returns:
*       TipoRetorno, Descripción retorno
*****/
```

Por cada comentario faltante, se restarán 5 puntos.

Por último, la indentación (1 TAB o 4 espacios), es muy importante. Por **cada bloque mal indentado, se quitarán 10 puntos**.