Trabajo Practico 1: Parte 2

Fecha de entrega: Definido Según comisión

Objetivos:

En el trabajo práctico integrador de este año, cada grupo deberá desarrollar una aplicación que permita a usuarios registrados acceder a gestionar un medallero olímpico. Esta aplicación tendrá como objetivo principal registrar y mostrar las medallas obtenidas por los países participantes en diversas disciplinas deportivas, asi como también el cálculo de estadísticas asociadas.

En la Parte 1, cada grupo ha especificado la estructura de menús de la interfaz de usuario, definiendo claramente las opciones para registrar medallas, consultar el medallero y generar reportes. En esta Parte 2, se procederá a definir y codificar los algoritmos necesarios para la gestión eficiente de la información del medallero.

La aplicación deberá ser implementada utilizando los conceptos desarrollados en la asignatura, tales como:

- Estrategia de descomposición modular descendente.
- Funciones y recursividad.
- Paso de parámetros.
- Tipos de datos abstractos.
- Archivos de textos y directos para el almacenamiento persistente de los datos del medallero.

Además, se espera que los alumnos mantengan el desarrollo de una aplicación amigable, que minimice la posibilidad de ingreso de datos erróneos, con mensajes adecuados que faciliten la experiencia del usuario. La aplicación deberá proporcionar una interfaz de usuario intuitiva y funcional.

El desarrollo de este medallero olímpico no solo reforzará el entendimiento de los conceptos teóricos vistos en clase, sino que también proporcionará una experiencia práctica valiosa en el manejo de datos y la resolución de problemas complejos.

ESTRUCTURAS DE DATOS A UTILIZAR

En base a lo desarrollado en la Parte 1, en esta nueva iteración se trabajará con un conjunto de requerimientos definidos en base a las estructuras de datos propuestas en este apartado. Los grupos NO PODRÁN agregar nuevas estructuras de datos en sus soluciones.

Países Participantes en las Competencias

La estructura `competidores` almacena los ID de los países que se disputan una medalla en una disciplina específica. Como mínimo, deben existir 4 países para que la disciplina se realice. Como máximo, pueden participar 16 países en una misma disciplina.

```
struct competidores{
        int paises[16];
        int tl;
};
```

Se sabe que existen 87 disciplinas, por lo que la configuración de la competencia olímpica se almacena en un vector `competencia` definido de la siguiente manera:

```
competidores competencia[87];
```

Medallas por Deporte

La estructura `deporte_medallas` almacena el ID del país que ha ganado la medalla oro-plata-bronce para cada uno de los 87 deportes propuestos.

```
int deporte_medallas[87][3];
```

INFORMACIÓN BRINDADA POR LA CÁTEDRA

Junto con este enunciado se brinda un proyecto ZinjaI en el cual se provee la implementación de funciones que deben utilizarse para resolver este enunciado.

Particularmente, se tiene:

- Para los Países: Se dispone de las funciones `listarPaises()` e `imprimirPais(int ID)`. La primera lista los países indicando su ID y su nombre. La segunda imprime por pantalla el nombre del país con identificador ID. Los ID de países van de 1 a 196.
- Para los Deportes: Se dispone de las funciones `mostrarDeportesIndividuales()` y `mostrarDeportesColectivos()`. En ambos casos, las funciones listan deportes indicando su ID y si nombre. Los ID de deporte van de 1 a 97.

I. REQUISITOS FUNCIONALES DE LA APLICACIÓN

Sobre la base del menú desarrollado en la Parte 1, se debe actualizar el contenido para visualizar las siguientes opciones:

MENÚ Principal

```
Menú Principal
==========================
1.- Generar Competencia
2.- Cargar Medallas por Deporte
3.- Mostrar Medallero
X.- Salir de la aplicación
Ingrese una opción:
```

En este menú, el funcionamiento solicitado es el siguiente:

- La *opción 1* debe realizar la carga aleatoria del arreglo competencia, pudiendo ejecutarse una única vez a lo largo de la ejecución del programa. Para la carga de datos tener en cuenta que un mismo país no puede participar más de una vez en una disciplina.
- La *opción 2* dirige al usuario al submenú correspondiente únicamente si se ha generado una competencia con anterioridad (es decir, se ha ejecutado con éxito la opción 1).
- La *opción 3* dirige al usuario al submenú correspondiente únicamente si se han cargado medallas por deporte (es decir, se ha ejecutado al menos una vez la opción 2).

Submenú Carga de Medallas por Deporte (Opción 2 del Menú Principal)

Sobre la base del menú desarrollado en la Parte 1, se debe actualizar el contenido para visualizar las siguientes opciones:

```
Menú Carga Medallas por Deporte
==========================
1.- Carga por Deporte Individual
2.- Carga por Deporte Colectivo
X.- Volver al menú principal

Ingrese una opción:
```

En este menú, el funcionamiento solicitado es el siguiente:

- *Opción 1*: Presenta el listado de los deportes Individuales usando la función `mostrarDeportesIndividuales()` que ha sido proporcionada por la cátedra. El usuario elije un deporte individual de la lista desplegada y la aplicación inicia el proceso de carga de las medallas (oro, plata y bronce) para dicho deporte. Para esto, se presenta al usuario el listado de países que participan en deporte seleccionado según la competencia generada. Luego, se solicita que indique el ID del país que ha ganado cada medalla. Esta información debe almacenarse en la estructura `deporte_medallas`. En cualquier momento el usuario puede abandonar la carga. Si al abandonar la carga no se ha completado la carga del podio, no debe almacenarse registro alguno del deporte.
- *Opción 2*: Presenta el listado de los deportes colectivos, usando la función `mostrarDeportesColectivos()` que ha sido proporcionada por la cátedra. El usuario elije un deporte individual de la lista desplegada y la aplicación inicia el proceso de carga de las medallas (oro, plata y bronce) para dicho deporte. Para esto, se presenta al usuario el listado de países que participan en el deporte seleccionado según la competencia generada. Luego, se solicita que indique el ID del país que ha ganado cada medalla. Esta información debe almacenarse en la estructura `deporte_medallas`. En cualquier momento el usuario puede abandonar la carga. Si al abandonar la carga no se ha completado la carga del podio, no debe almacenarse registro alguno del deporte.
- *Opción X*: Vuelve al usuario volver al menú principal.

Submenú Mostrar Medallero

Sobre la base del menú desarrollado en la Parte 1, se debe actualizar el contenido para visualizar las siguientes opciones:

```
Menú Medallero
==========================
1.- Medallero por País
2.- Medallero por Deporte
3.- Top Medallero
X.- Volver al menú pricipal

Ingrese una opción:
```

En este menú, el funcionamiento solicitado es el siguiente:

- *Opción 1*: Se presenta al usuario el medallero olímpico ordenado por importancia (oro, plata, bronce) con el total de medallas obtenidas por los 187 países. Si hay países que no han obtenido medallas, no deben visualizarse.
- *Opción 2*: Se muestra al usuario el listado de deportes en los que se cargaron medallas, indicando los nombres de los países que han conformado el podio.
- *Opción 3*: El usuario ingresa el valor TOP a visualizar. Por ejemplo, si ingresa 10, se presenta el TOP 10 de los países que más medallas han ganado (en total).

PAUTAS DE ENTREGA

CONFORMACIÓN DE GRUPOS

Se deben mantener los grupos de la Parte 1. En caso de que algún grupo deba cambiar su conformación, DEBE CONSULTAR LA SITUACIÓN CON LOS PROFESORES. No se aceptarán entregas de grupos que no respeten esta pauta y no hayan sido debidamente autorizados por los profesores.

EVALUACIÓN

La solución entregada por cada grupo será revisada y evaluada por parte de los profesores, como así también una prueba de funcionamiento. Si la misma cumple con el funcionamiento requerido, el TP se considerará Aprobado, caso contrario podrán solicitarse correcciones específicas y la reentrega del mismo en plazo determinado por la cátedra. El código entregado no solamente debe compilar, sino también ajustarse a las buenas prácticas de codificación aprendidas a lo largo del curso.

Los trabajos no entregados en tiempo y forma, NO SERÁN EVALUADOS y se considerarán NO APROBADOS. Lo mismo ocurrirá con los trabajos entregados por grupos que no respeten la conformación de grupos.

POLÍTICA ANTIPLAGIO

Las entregas realizadas por los grupos son individuales. Los archivos entregados serán analizados con herramientas de software específicas para detectar plagios de código.

Los trabajos que no cumplan con las pautas de autoría establecidas, NO SERÁN EVALUADOS y se considerarán NO APROBADOS.

CONSULTAS

Las consultas de los grupos serán atendidas por los docentes mediante emails, en el Canal virtual empelado por cada comisión y por mensajes en el Campus Virtual.

DOCUMENTACIÓN A ENTREGAR

Las entregas se realizarán exclusivamente por medio del Campus Virtual en la tarea destinada para tal fin por un único integrante del grupo.

Cada grupo debe entregar:

- Archivo “Integrantes.txt”: Archivo de texto que contendrá el apellido, nombre y correo electrónico de cada uno de los integrantes del grupo (uno por línea).
- El programa C++ (archivo/s .cpp), que debe estar comentado, junto con su respectivo archivo ejecutable (archivo .exe).
- Estos archivos deben comprimirse en un único archivo ZIP/RAR nombrado con el siguiente formato *AEDD_GRUPO_apellido1_apellido2_apellido3*.
    - GRUPO: hace referencia a identificación que los docentes le asignaron (Nombre/Número)