UTN FRMDP - TUP - Laboratorio 2 – Primer Parcial – 30 de septiembre de 2022.-

IMPORTANTE:

- **Crear un proyecto con su Apellido y nombre.**
- **Realizar todas las funciones que se indican, añadiendo comentarios a su código identificando con el número de ejercicio e inciso tanto a la/s función/es vinculadas con su resolución, como a la parte del Main correspondiente al ejercicio e inciso. Si no se identifica, la resolución no tendrá validez.**

Dado que nos encontramos próximos al Mundial de Fútbol, vamos a realizar algo relacionado con ello para ir entrando en clima…
Los datos a utilizar se encuentran almacenados en el archivo de jugadores subido en el espacio de entrega del Parcial, el cual hay que descargar y copiar dentro de la carpeta del proyecto del Parcial, y responde a la siguiente estructura de datos:

```
typedef struct {
    int idEquipo;
    char nombreEquipo[30];
    int idJugador;
    int nroCamisetaJugador;
    char nombreJugador[30];
    int puntosAnotados;
} stJugadorEquipo;
```

Tenemos que desarrollar la siguiente funcionalidad:

1. Recorrer el archivo de jugadores y construir con dichos datos una **LISTA DOBLEMENTE ENLAZADA de Jugadores**. Para realizar esto, se deberá:
   - crear la estructura necesaria,
   - inicializar lo que corresponda,
   - y codificar las funciones que sean necesarias, **modularizando de forma correcta.**
   - Los datos deben ser agregados **en el lugar de la lista en el cual resulte más EFICIENTE realizar el proceso de agregado** (al final?, al principio?, o en orden?),
   - y **la función que los agrega a la lista debe utilizar como parámetro un PUNTERO DOBLE a la lista.**

   (20 puntos)

2. Hacer una función que muestre la **LISTA DOBLEMENTE ENLAZADA** obtenida en el punto anterior, **modularizando de forma correcta.**
   - L**a lista deberá ser mostrada AL REVÉS de manera ITERATIVA**. Para ello, pensar qué parámetros deberá recibir.
   - Para ahorrar tiempo, ya se nos ha dado la porción de código correspondiente al mostrado por pantalla de los datos:

```
printf("\n ********** JUGADOR  ********** \n";
printf("\nId del equipo.....: %d ", p.idEquipo);
printf("\nEquipo.....: %s ", p.nombreEquipo);
printf("\nId del Jugador.....: %d ", p.idJugador);
printf("\nNro de camiseta del Jugador.....: %d ", p.nroCamisetaJugador);
printf("\nApellido y nombre del Jugador.....: %s ", p.nombreJugador);
printf("\nPuntos anotados por el Jugador.....: %d \n\n", p.puntosAnotados);
```

   (11 puntos).

3. En relación a la **LISTA DOBLEMENTE ENLAZADA** obtenida en el punto anterior debemos realizar lo siguiente:
    - Una función auxiliar **RECURSIVA** que busque el ultimo nodo de la lista
    - Una función principal que reciba por parámetro un **PUNTERO DOBLE** a la lista, invoque a la anterior, y borre el último nodo de la lista.

(16 puntos)

4. Para poder analizar información de cada equipo por separado, debemos codificar las funciones necesarias para poder **generar una LISTA SIMPLEMENTE ENLAZADA** con **todos los jugadores de la Lista doblemente enlazada que pertenezcan a un equipo a elección del usuario del sistema (sin borrar los mismos de la Lista Doble original)**. Para ello se deberá:
    - crear la estructura necesaria,
    - inicializar lo que corresponda,
    - y codificar las funciones que sean necesarias, **modularizando de forma correcta.**.
    - **Los jugadores deberán ser agregados a la Lista Simple ordenados de menor a mayor** de acuerdo a su cantidad de puntos anotados.

(20 puntos).

5. Hacer una función que muestre la **LISTA SIMPLEMENTE ENLAZADA** obtenida en el punto anterior,
    - **Deberá modularizarse de forma correcta** y teniendo en cuenta el código ya realizado en el punto 2.
    - **La función principal deberá ser RECURSIVA**

(8 puntos)

6. A los jugadores se les abona un plus por cada punto anotado:
    - a cada jugador que supere los 40 puntos anotados, se le abonan $5.000 por punto
    - a cada jugador que NO supere los 40 puntos anotados, se le abonan $2.000 por punto
    - **En una misma función** debemos **recorrer por única vez la lista** y calcular y **retornar por separado** el total a pagar a quienes superan los 40 puntos y el total a pagar a quienes no los superan (pensar cómo devolver ambos valores al Main)

(15 puntos).

7. Hacer una función main()
    - Para hacer esto, cree las variables que considere necesarias e invoque las funciones (de forma directa o indirecta) como corresponde en cada caso.
    - Muestre los resultados cada vez que sea necesario.
    - A fin de identificar cada inciso, comente su código indicando a qué apartado corresponde, por ejemplo: // Ejercicio 3.a

(10 puntos)