(3640) Algoritmos Y Estructuras de Datos
Final MESA B

28 / 07 / 2025

Apellido y Nombre:

DNI:

Calificación:

Ejercicio 1: Una compañía de videojuegos está desarrollando un TDA “tabla de puntos” donde mantiene los puntajes de los videojuegos al estilo de la década del 80’, guardando 3 iniciales y un puntaje y solo manteniendo el top N de puntajes. El mismo utiliza internamente una lista simplemente enlazada genérica y tiene el .h detallado más abajo.
Desarrolle las primitivas de lista correspondientes necesarias para completar la funcionalidad anteriormente solicitada, recuerde que **la lista no puede tener más de N nodos al finalizar cada operación de inserción**.
Desarrolle al menos 3 lotes de prueba donde simula insertar una cantidad M de puntajes en la tabla de puntos utilizando un archivo de texto, donde se especifica en la primera línea el tamaño de la tabla y en las líneas siguientes los puntajes para cada inicial.

tabla_de_puntos.h

```
#ifndef PUNTAJE_H_INCLUDED
#define PUNTAJE_H_INCLUDED
#include<lista.h>

typedef struct{
    char iniciales[4];
    unsigned puntaje;
}t_puntaje;

typedef struct{
    t_lista tabla;
    unsigned tam;
}t_tabla_puntos;

void crear_tabla_puntos(t_tabla_puntos* p, unsigned tam);
int agregar_puntaje(t_tabla_puntos* p, const t_puntaje* dato);
void imprimir_tabla_puntos(t_tabla_puntos* p);
void vaciar_tabla_puntos(t_tabla_puntos* p);

#endif //PUNTAJE_H_INCLUDED
```

Lote de pruebas

```
5
AAA 500
BBB 700
DDD 540
EEE 320
FFF 630
GGG 520
HHH 710
III 620
```

Salida

```
1. HHH …… 710
2. BBB …… 700
3. FFF …… 630
4. III …… 620
5. DDD …… 540
```

Ejercicio 2: Una institución que se dedica a organizar competencias a nivel mundial, está creando un TDA “Podio” para incluir en sus programas de gestión de competencias donde se almacenan un listado de competidores en formato de podio de N posiciones. Tenga en cuenta que los podios admiten empates y solo en caso de empate puede suceder que el podio contenga más de N participantes en la misma posición e incluso en todo el listado. El mismo utiliza internamente una lista simplemente enlazada genérica y tiene el .h detallado más abajo.
Desarrolle las primitivas de lista correspondientes necesarias para completar la funcionalidad anteriormente solicitada, recuerde que **la lista no puede tener más de la cantidad necesaria de nodos al finalizar cada operación de inserción**.
Desarrolle al menos 3 lotes de prueba donde simula insertar una cantidad M de participantes en el podio utilizando un archivo de texto, donde se especifica en la primera línea la cantidad de participantes y en las líneas siguientes los puntajes para cada uno de ellos sin ningún orden en particular.

podio.h

```
#ifndef PODIO_H_INCLUDED
#define PODIO_H_INCLUDED
#include<lista.h>

typedef struct{
    char desc[51];
    unsigned puntos;
}t_participante;

typedef struct{
    t_lista lista;
    unsigned tam;
}t_podio;

void crear_podio(t_podio* p, unsigned tam);
int agregar_participante(t_podio* p, const t_participante* dato);
void imprimir_podio(t_podio* p);
void vaciar_podio(t_podio* p);

#endif //PODIO_H_INCLUDED
```

Prueba 1

```
3
AAA 500
BBB 700
DDD 540
EEE 320
FFF 630
GGG 520
HHH 710
III 620
```

Salida 1

```
1. HHH …… 710
2. BBB …… 700
3. FFF …… 630
```

Prueba 2

```
3
AAA 500
BBB 700
DDD 540
EEE 320
FFF 630
GGG 700
HHH 710
III 700
```

Salida 2

```
1. HHH …… 710
2. BBB …… 700
2. GGG …… 700
2. III …… 700
```

EVALUACIÓN TOMADA EN LABORATORIO

EVALUACIÓN PRESENCIAL

NOTA:
Resolver:

- **Ejercicio1 para notas de 4 a 6,**
- **Ejercicio2 para notas de 7 a 10.**

La resolución es inválida en cualquiera de los ejercicios si supone y/o utiliza variables globales.

Crear una carpeta con su DNI_Apellido_Nombre, guardar en ella todo lo realizado.

NOTA GENERAL

- **La hora límite de entrega será indicada en el aula**.
- **NO SE RETIRE DE LA UNIVERSIDAD, la corrección se hace en el momento con usted presente.**
- Debe entregar el proyecto con 0 errores y 0 warnings.
- Desarrolle lo solicitado en ANSI C estándar.
- El ejercicio debe ejecutarse correctamente.
- Los archivos deben ser leídos una única vez y no se cargan en memoria salvo que se indique expresamente el tamaño máximo posible que puedan tener y sea un valor manejable.
- Vectores y cadenas de texto deberán ser manipulados utilizando aritmética de punteros.
- Las soluciones tienen que ser eficientes:
    - En el uso de memoria, por tanto, no declare vectores o matrices auxiliares si no es necesario.
    - En cantidad de ciclos de procesador y en el caso de matrices las soluciones deben ser óptimas.
- No acceda nunca a memoria que no le pertenece y nunca deje memoria sin liberar.
- Declare variables al inicio del bloque por compatibilidad ANSI C y no utilice VLA (Variable length arrays).
- **Incluya en el encabezado de cada archivo, // DNI_apellido_nombre**
- Recuerde antes de comprimir, eliminar las carpetas bin y obj de cada proyecto.
- **Entregue cada proyecto compactado en un zip, “DNI_apellido_nombre.zip”.**
- Entregue el examen usando el portfolio de MIEL.
- Enviar a todos los tutores.
- ¡La evaluación es individual!

¡El mayor de los éxitos!