

Fundamentos de la programación – Grupos A y B

Curso 2019–2020

Examen de enero

Tiempo disponible: 3 horas

Desarrolla un programa en C++ para gestionar la colocación de los libros de una biblioteca en sus estanterías.

La biblioteca está compuesta por una serie de estanterías (máximo 15). Cada estantería contiene libros de una determinada área (Matemáticas, Física o Informática); la capacidad de cada estantería se cuenta en número de páginas, de forma que sólo caben libros por un total de páginas igual o menor que esa capacidad.

Utiliza enum, structs, arrays o combinaciones de ambos para implementar al menos los siguientes tipos:

- enum* • tArea para representar las distintas áreas,
- struct* • tLibro para representar libros, conteniendo su área, su número de páginas y su título.
- struct + array* • tListaLibros para representar listas de libros, con su array y su contador, teniendo en cuenta que todas las listas de libros tienen como máximo 100 libros,
- struct* • tEstanteria para representar cada estantería con su área, su lista de libros, su capacidad y su número de páginas disponibles.
- struct + array* • tBiblioteca para representar la biblioteca con su array de estanterías y su contador.

La distribución de las estanterías de la biblioteca se obtiene de un archivo de texto llamado biblioteca.txt que contiene el área y la capacidad en páginas de cada estante. El archivo comienza con el número de estanterías de la biblioteca y las áreas señaladas son: Mat (Matemáticas), Fis (Física) e Inf (Informática).

```
10
Mat 1000
Inf 900
Mat 800
Fis 700
Inf 1200
Fis 1000
Mat 600
Inf 1000
Fis 500
Inf 700
```

La lista de libros que hay que colocar en las estanterías se obtiene de otro archivo de texto llamado libros.txt que contiene el área, el número de páginas y el título de cada libro. Este archivo nunca va a tener más de 100 libros y termina con el centinela XXX.

Inf 200 Arquitectura de Computadores IV
Mat 150 Matematicas Basicas III
Mat 100 Matematicas Discretas I
Fis 100 Introduccion a Fisicas
Inf 350 Arquitectura de Computadores III
Inf 250 Programacion Avanzada III
Mat 250 Matematicas Avanzadas I
Inf 250 Programacion de Computadores IV
Mat 150 Matematicas Avanzadas III
Mat 150 Matematicas Discretas II
...
XXX

Tras cargar la distribución de las estanterías y la lista de libros a colocar, se van asignando estos a las estanterías por orden de mayor número de páginas. Para cada libro, se busca la primera estantería de su área que tenga sitio para el número de páginas del libro. Si no se consigue colocar el libro, este se guarda en una lista de libros no colocados. Cada estantería tiene, además de una capacidad en páginas, un número de páginas disponibles.

Programa Principal (0.5 puntos)

El programa comenzará cargando la distribución de las estanterías de la biblioteca del archivo biblioteca.txt. Si esta carga tiene éxito, se cargará la lista de libros a colocar desde el archivo libros.txt. Si esta última carga también tiene éxito, se asignarán los libros a las estanterías de la biblioteca por orden de mayor número de páginas. Tras procesar todos los libros, se mostrará la lista de libros de cada estantería y la lista de libros que no se han podido colocar, según el formato del ejemplo de ejecución (al final del documento).

Carga de Datos (2 puntos)

Implementa, al menos:

str
return

✓ tArea strToArea (string str): dado un string, devuelve el valor correspondiente del tipo tArea.
return

funcion ✓ bool cargarEstanterias(tBiblioteca &biblioteca): devuelve la biblioteca con las estanterías del archivo biblioteca.txt, y un booleano que indica si el archivo se ha podido abrir o no.

f. ✓ bool cargarLibros(tListaLibros &listaLibros): devuelve la lista de libros con los libros del archivo libros.txt, y un booleano que indica si el archivo se ha podido abrir o no.

Asignación de Libros a Estanterías (3.5 puntos)

Implementa, al menos:

- ✓ `void borrarLibro(tListaLibros &listaLibros, int indice):` elimina de la lista de libros `listaLibros` el libro que está en la posición `indice`.
- ✓ `tLibro mayor(tListaLibros &listaLibros):` dada una lista de libros `listaLibros`, encuentra el libro con el mayor número de páginas, lo elimina de la lista y lo devuelve.
- ✓ `int buscarEstanteria(const tBiblioteca &biblioteca, tLibro libro):` dada una biblioteca (lista de estanterías) y un libro, busca una estantería en la biblioteca cuya área es la del libro y cuyo número de páginas disponibles son suficientes para el libro. Devuelve la posición de la estantería o -1 si no encuentra ninguna.
- ✓ `void asignar(tBiblioteca &biblioteca, tListaLibros listaLibros, tListaLibros &sinColocar):` recorre la lista de libros `listaLibros` cogiendo el libro que tiene mayor número de páginas (usando la función `mayor`), buscando una estantería para dicho libro (usando la función `buscarEstanteria`) y colocándolo en la biblioteca o en la lista de libros `sinColocar` si no hay estantería.

Visualización de las Listas (2 puntos)

Implementa, al menos:

- ✓ `string areaToStr(tArea area):` dado un valor de `tArea`, devuelve el string correspondiente.
- ✓ `void mostrarLibros(const tListaLibros &listaLibros):` dada una lista de libros muestra la información de cada libro en la pantalla, cada uno en una línea, según el formato del ejemplo de ejecución (al final del documento).
- ✓ `void mostrarBiblioteca(const tBiblioteca &biblioteca):` dada una biblioteca muestra la información de cada una de sus estanterías en la pantalla, según el formato del ejemplo de ejecución (al final del documento). Para ello usa el procedimiento `mostrarLibros`.

Corrección

Recuerda que no se permite el uso de variables globales ni de instrucciones de salto, salvo un `return` como última instrucción de las funciones. Se valorará (2 puntos) la corrección del diseño, la ausencia de código repetido, utilización de esquemas, la eficiencia del código, etc.

Instrucciones de Entrega

- Añade al inicio de tu archivo `.cpp` un comentario con tus datos:
/*
Nombre completo y DNI
Grupo
Laboratorio y puesto
*/

- Sube tu examen (solo el archivo .cpp) al enlace disponible en el campus.
- Pasa por el ordenador del profesor, muestra el DNI y firma.

Ejemplo de Ejecución

Matematicas - 1 (1000/0)

Matematicas Basicas II (400 paginas)
 Matematicas Avanzadas II (350 paginas)
 Matematicas Avanzadas I (250 paginas)

Matematicas - 2 (800/50)

Matematicas Basicas IV (300 paginas)
 Matematicas Discretas III (300 paginas)
 Matematicas Basicas III (150 paginas)

Matematicas - 3 (600/50)

Matematicas Basicas I (250 paginas)
 Matematicas Avanzadas III (150 paginas)
 Matematicas Discretas II (150 paginas)

Fisica - 1 (700/0)

Fisica Avanzada I (250 paginas)
 Fisica Basica I (250 paginas)
 Fisica Basica II (200 paginas)

Fisica - 2 (1000/500)

Fisica Avanzada II (250 paginas)
 Fisica Basica III (150 paginas)
 Introduccion a Fisicas (100 paginas)

Fisica - 3 (500/500)

...

Informatica - 4 (700/0)

Programacion Avanzada III (250 paginas)
 Programacion de Computadores IV (250 paginas)
 Arquitectura de Computadores IV (200 paginas)

Libros que no pudieron ser colocados...

Programacion Avanzada II (250 paginas)
 Programacion de Computadores II (250 paginas)
 Matematicas Discretas I (100 paginas)