Estructuras de Datos y Algoritmos II

Práctico de máquina 1 - Año 2025

Fecha de entrega: Jueves 08 de Mayo de 2025 hasta las 8 hs.

Se desea implementar una agenda mensual diaria definida con la siguiente relación y el siguiente servicio de evocación:

$$\begin{array}{rcccccccl} AgendaMensual & \subseteq & Fecha & \times & Evento & \times & Hora & \times & Lugar \\ S & : & * & & ? & & ? & & ? \quad \text{donde } Fecha \nrightarrow (Evento, Hora, Lugar) \end{array}$$

Además se sabe que por día sólo puede haber un evento por cada hora. Para almacenar la información planteada se utilizarán las siguiente estructuras:

- Lista Secuencial Ordenada con búsqueda binaria (LSOBB) sin forzar dependencia funcional.
- Lista Secuencial Ordenada con búsqueda binaria (LSOBB_F) forzando dependencia funcional.
- Árbol Binario de Búsqueda (ABB) sin forzar dependencia funcional.
- Árbol Binario de Búsqueda (ABB_F) forzando dependencia funcional.

La aplicación deberá presentar un menú de opciones principal que permita seleccionar las siguientes opciones: **Comparación de estructuras** y **Administrar Estructura** (una opción de administración por cada estructura).

La opción **Administrar Estructura** deberá presentar dos opciones: ***Mostrar Estructura*** y ***Evocar***.

La opción **Mostrar Estructura** debe mostrar por pantalla el contenido de la estructura seleccionada. Para las listas mostrar los días y los eventos asociados para cada uno, para los Árboles implementar un barrido preorden mostrando los eventos de cada día y por cada nodo además mostrar el campo **Fecha** de los nodos hijos indicando si es hijo izquierdo o derecho.

La opción **Evocar** debe permitir al usuario ingresar un día y retornar los aventos asociados al mismo, el procesamiento de los datos será mostrarlos por pantalla.

Comparación de Estructuras: Esta opción debe realizar y mostrar una comparación adecuada de lo que cuesta, en cada una de las estructuras, **realizar ingresos, eliminaciones y consultas de eventos en una fecha dada**. En el análisis debe considerar el peor escenario y el comportamiento esperado en cada caso. Una vez finalizada esta operación, deberá realizar un análisis de los resultados obtenidos y sacar una conclusión de los mismos; dicha conclusión deberá quedar plasmada al principio de su programa principal (donde se encuentra el main) como comentario (**incluir los resultados de la comparación**).

Para el cálculo de los costos de ingreso y eliminación: en las listas secuenciales la función de costo será la cantidad de corrimientos y cada corrimiento de elemento tiene costo **1 (uno)**. En el ABB se considerarán modificaciones de punteros (**0,5 por cada modificación**) y en el caso de utilizar la política de reemplazo se deberá sumar **un costo más (1)** por la copia de datos.
Para las consultas el costo se determinará en **cantidad de comparaciones por el campo Fecha** para todas las estructuras, **un punto (1) por cada comparación**.

Para comparar las estructuras se utilizará una secuencia de operaciones detallada en el archivo de texto *"Operaciones.txt"* que contiene información de eventos y será provisto por la cátedra (disponible en la *página web de la materia*). Esta secuencia de operaciones se deberá realizar sobre cada una de las estructuras, asegurando que las mismas **no contengan ningún dato inicialmente**. Una vez finalizada la secuencia de operaciones se mostrarán por pantalla los costos

obtenidos para cada estructura. Además en las estructuras **deben quedar** los datos resultantes de efectuar las operaciones del archivo para ser alcanzados desde la opción administrar de cada una de ellas.

El archivo de texto *"Operaciones.txt"* contiene una línea con el código de operación (1-Alta, 2-Baja y 3-Evocación) y a continuación los datos de la nupla necesarios para la operación en cada línea (renglón) del mismo. Un ejemplo de esa información se muestra a continuación:

```
1                       /*código de la primera operación (Alta)*/
2025-12-01              /*fecha evento*/
Reunión Laboral         /*evento*/
10                      /*hora*/
calle 4 DE JUNIO 100    /*lugar*/
3                       /*código de la segunda operación (Evocación)*/
2025-12-25              /*fecha evento*/
                .
                .
                .
2                       /*código de la n-ésima operación (Baja)*/
2025-12-01              /*fecha evento*/
Reunión Laboral         /*evento*/
10                      /*hora*/
calle 4 DE JUNIO 100    /*lugar*/
```

Consideraciones a tener en cuenta:

- Se espera un máximo de 24 eventos por día para un máximo de 31 días al mes.
- Las listas deberán estar ordenadas de menor a mayor respecto de la Fecha (no hay orden entre los eventos de una fecha determinada).
- Para las listas con búsqueda binaria la consigna a utilizar será **bisección**, límite inferior inclusivo, límite superior inclusivo, testigo y segmento mas grande a la izquierda. Para ellas no se utilizará ningún elemento ficticio para indicar fin de lista.
- La **confirmación de la baja** en la rutina de baja debe realizarse por código, es decir comparando toda la nupla (en todas las estructuras) .
- La politica de reemplazo en la baja de los **Árboles** cuando el nodo tiene dos hijos es el **mayor de los menores** y el reemplazo deberá realizarse con copia de datos.
- En el Árbol Binario sin forzar dependencia los elementos menores o iguales se ubican por la rama izquierda.
- En las estructuras que fuerzan dependencia, las listas vinculadas para los eventos de un día particular son desordenadas.
- La fecha es una cadena de caracteres con el formato AAAA-MM-DD.
- El evento puede contener un máximo de 80 caracteres en cada caso.
- El campo hora es un entero de 0 a 23.
- El lugar puede contener un máximo de 80 caracteres.
- El ingreso de datos **no debe ser sensible a mayúsculas y minúsculas**, esto significa por ejemplo que Reunión = REUNIÓN = REUnión.
- El programa deberá desarrollarse en Lenguaje C, utilizando como entorno de desarrollo para tal fin **Code::Blocks** (disponible en www.codeblocks.org).

Ejemplo de rutina para Lectura de Operaciones

El código que se presenta a continuación es una guía para programar una rutina que permita leer datos desde un archivo de texto. **Deberá adaptarlo a la situación planteada.**

```
int Lectura_Operaciones()
{   .... //declaraciones
        FILE *fp;
        if (( fp = fopen ( ''Operaciones.txt'' , ''r'') )==NULL)
                return 0;
        else {
                while(!(feof(fp))){
                        fscanf(fp,''%d'', &codigoOperador);
                        fscanf(fp,'' %[^\n]'',&aux.fecha);

                        if(codigoOperador==1||codigoOperador==2){
                                fscanf(fp,'' %[^\n]'',aux.evento);
                                fscanf(fp,''%d'', &aux.hora);
                                fscanf(fp,'' %[^\n]'',aux.lugar);
                                //llamar al operador correspondiente (Alta o Baja)
                                //en todas las estructuras

                        }else if(codigoOperador==3){
                        //Evocar en todas las estructuras
                        }else{
                        //error, código operación no reconocido
                        }
            codigoOperador=0;
                }
        fclose(fp);
        return 1;
        }
}
```

Importante:

- Los grupos deben ser de 2 integrantes.
- Los códigos fuente entregados que no compilen o estén incompletos respecto de la funcionalidad solicitada no serán revisados.
- La entrega del práctico se realiza por medio de la página de la materia y se debe enviar el archivo fuente del programa.
- El nombre del archivo deberá estar conformado de la siguiente manera: ***P****nroP*-***Grupo****nroG* donde *nroP* es reemplazado por el número de práctico que se entrega y *nroG* por el número del grupo al que pertenece el programa. Por ejemplo, el nombre P1-Grupo22.c corresponde al práctico de máquina 1 enviado por el grupo 22. **Los programas cuyos nombres no respeten estas reglas de conformación no serán aceptados**.