Estructuras de Datos y Algoritmos

Práctico de máquina 2 - Año 2021

Fecha de entrega: Lunes 18 de Octubre de 2021 hasta las 8 hs.

Un importador de electrodomésticos necesita informatizar el listado de artículos existentes en su depósito. Cada artículo tiene un *código* que lo identifica y la información asociada al mismo es: *tipo de artículo, marca, descripción, precio unitario en dólares* y *cantidad existente*.

Se necesita diseñar una aplicación que permita resolver los requerimientos del importador y para ello se cuenta con las siguientes estructuras de datos para almacenar la información mencionada:

a) Rebalse Abierto Lineal (RAL).

b) Rebalse Abierto Cuadrático (RAC).

c) Árbol Binario de Búsqueda (ABB).

La aplicación deberá presentar un menú de opciones principal que permita seleccionar la estructura con la que se desea trabajar y para cada una de ellas un nuevo menú que muestre las opciones para administrarla. Este menú debe presentar por pantalla las siguientes opciones: **ingreso de nuevos artículos, eliminación de artículos existentes** (con confirmación por pantalla) y **consulta de artículos**. Además, debe contener las opciones **Mostrar Estructura** y **Memorización Previa**.

La opción **Memorización Previa** es una rutina que permite guardar en una estructura la información incluida en el archivo de texto *"Articulos.txt"* provisto por la cátedra. Esta rutina debe leer desde el archivo la información correspondiente a un artículo e insertarla en la estructura correspondiente.
La opción **Mostrar Estructura** para los **Rebalses** debe mostrar por pantalla el contenido de la estructura seleccionada, listando la **estructura completa**. Se deben mostrar las **M** posiciones de cada estructura, presentando la información completa de los elementos presentes en ella, y para las celdas que no están ocupadas, diferenciar claramente las celdas *libres* de las celdas *nunca usadas*; para el **ABB** se deberá realizar un barrido preorden mostrando la información completa de la nupla y además el código de los hijos izquierdo y derecho si los tuviera o un cartel indicando que no los tiene.

Consideraciones a tener en cuenta:

- El avance del RAL y la política de reemplazo en ABB será la consigna asignada a cada grupo en el cuadro anexo "Consignas por grupo"
- Se esperan 250 artículos.
- Para el **RAL** se tendrá ρ= 0.75.
- Para el **RAC** se tendrá ρ= 0.83.
- Se utilizará una ranura por balde en cada Rebalse.
- Se debe utilizar la siguiente función de hashing:

```
int hashing (char* x, int M) {
        int longitud, i;
        int contador=0;
        longitud=strlen(x);
        for (i=0; i< longitud; i++)
                contador+=((int)x[i]) * (i+1);
        return (contador %M );
}
```

- Sobre los datos:
  - El código de artículo es una secuencia de 8 caracteres.
  - El tipo de artículo puede contener un máximo de 20 caracteres.
  - La marca es una secuencia de a lo más 30 caracteres.
  - La descripción puede contener un máximo de 100 caracteres.
  - El valor del artículo es un valor real positivo.
  - La cantidad es un número entero positivo.
- El ingreso de datos **no debe ser sensible a mayúsculas y minúsculas**, esto significa que al buscar un código de artículo deberá ser reconocido independientemente de cómo se ingresen las letras del mismo. Ejemplo: los códigos AAAB534A, aaAb534a, AaaB534a, Aab534A son consideradas iguales.
- El programa deberá desarrollarse en Lenguaje C (estándar), utilizando como herramienta para tal fin **Code::Blocks** (disponible en www.codeblocks.org).

Nota Importante: 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**.

Consignas por Grupo

Grupos	Avance RAL	Política de reemplazo ABB
1, 9, 17, 25 y 33	3	mayor de los menores
2, 10, 18, 26 y 34	3	menor de los mayores
3, 11, 19, 27 y 35	5	mayor de los menores
4, 12, 20, 28 y 36	5	menor de los mayores
5, 13, 21, 29 y 37	-3	mayor de los menores
6, 14, 22, 30 y 38	-3	menor de los mayores
7, 15, 23, 31 y 39	-5	mayor de los menores
8, 16, 24, 32 y 40	-5	menor de los mayores

Ejemplo de rutina para Memorización Previa

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 Memorización_Previa()
{   .... //declaraciones
        FILE *fp;
        if (( fp = fopen ( ''Articulos.txt'' , ''r'') )==NULL)
                return 0;
        else {
                while(!(feof(fp))){
                        fgets (CodArt ,10 , fp);
                        CodArt[8]='\0';
                        fgets (TipoArt ,21 , fp );
                        TipoArt[strlen(TipoArt)-1]='\0';
                        fgets (Marca ,31 , fp );
                        Marca[strlen(Marca)-1]='\0';
                        fgets (Desc ,101 , fp );
                        Desc[strlen(Desc)-1]='\0';
                        fscanf(fp,''%f'',&Valor);
                        fscanf(fp, ''%c'', &barraene );
                        fscanf(fp,''%d'',&Cant);
                        fscanf(fp, ''%c'', &barraene );

            /* Donde CodArt , TipoArt, Marca, Descripción, Valor,
                Cant corresponden a la información guardada en el
                archivo Artículos.txt en la posición corriente y barraene
                es una variable auxiliar para consumir el caracter nueva
                línea después de los valores numéricos */
                        .....
                /* Invocar los procedimientos que correspondan */
                        .....
                        }
                return 1;
        }
        fclose(fp);
}
```

(*) La función **fgets** lee sólo strings desde un archivo, por lo tanto si se usa para leer números éstos son considerados strings y no valores numéricos, si se los quiere operar como números hay que transformarlos en tales o leerlos con otra función (por ejemplo fscanf). El **fgets** lee como último caracter el "\n", por lo tanto para que **NO** dé error al comparar lo que se leyó con lo que se guarda el "\n", éste deberá ser reemplazado por "\0".

char *fgets(char *s, int n, FILE *stream);

donde n es la cantidad de caracteres que se quieren leer, s la variable donde se lee y file el archivo desde donde se lee.

int fscanf(FILE *stream, const char *format[, address, ...])

donde format es el string de formato de lo que se va a leer (entero, flotante, string, etc.), address la variable donde se lee y file el archivo desde donde se lee.