Estructuras de Datos y Algoritmos II

Práctico de máquina 2 - Año 2025

Fecha de entrega: Lunes 2 de junio de 2025 hasta las 8 hs.

Una empresa de ventas con múltiples franquicias desea registrar el desempeño de sus vendedores manteniendo la siguiente información: el *número de documento* que identifica a cada vendedor y es único, *Nombre y Apellido, teléfono, monto vendido, cantidad vendida* y *canal de venta principal*.

Se necesita diseñar una aplicación que permita resolver los requerimientos y para ello se cuenta con la siguiente estructura de datos para almacenar la información mencionada:

- Rebalse Abierto Lineal (**RAL**).

La aplicación deberá presentar un menú que muestre las siguientes opciones para administrarla:

- Ingreso de nuevos vendedores: solicitar los datos al usuario y llamar a la rutina de alta.
- Eliminación de vendedores existentes: al seleccionar ésta opción se debe ingresar el dni de un vendedor, pedir la confirmación de baja por pantalla y realizarla si corresponde.
- Consulta de vendedores: debe pedir el dni al usuario, realizar la evocación y con el resultado obtenido informar al usuario convenientemente.
- Consulta de balde contenedor: debe pedir el dni de un vendedor al usuario, localizar el vendedor retornando el balde que lo contiene y mostrar todas las ranuras del balde por pantalla presentando la información completa de los elementos presentes, y para las ranuras que no están ocupadas, diferenciar claramente las ranuras *libres* de las ranuras *nunca usadas*.
- Mostrar estructura: debe mostrar por pantalla el contenido de la **estructura completa**. Se deben mostrar las **M*r** posiciones de cada estructura, presentando la información completa de los elementos presentes en ella, y para las ranuras que no están ocupadas, diferenciar claramente las ranuras *libres* de las ranuras *nunca usadas*.
- Memorización Previa: es una rutina que permite guardar en una estructura la información incluida en el archivo de texto *"Vendedores.txt"* provisto por la cátedra. Esta rutina debe leer desde el archivo la información correspondiente a un vendedor e insertarla en la estructura correspondiente.
- Análisis de costos: en esta opción debe realizar y mostrar un análisis adecuado de lo que cuesta **realizar ingresos, eliminaciones y consultas de un vendedor dado**. 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). Los costos para todas las operaciones se mediran tanto en accesos a baldes como en ranuras consultadas (mostrar valores por separado).

  Para el análisis se utilizará una secuencia de operaciones detallada en el archivo de texto *"Operaciones.txt"* provisto por la cátedra. Esta secuencia de operaciones se deberá realizar sobre la estructura, asegurando que la misma **no contenga ningún dato inicialmente**. Una vez finalizada la secuencia de operaciones se mostrará por pantalla los costos obtenidos.

Consideraciones a tener en cuenta:

- El número esperado de vendedores a almacenar es de 210.
- El número de documento es un entero no mayor a 99.999.999.
- El campo nombre y apellido puede contener un máximo de 50 caracteres.
- El teléfono puede contener un máximo de 15 caracteres.
- El monto es un valor real positivo.
- La cantidad es un número entero positivo.
- El canal de venta puede contener un máximo de 20 caracteres.
- El factor de carga será ρ= 0.90.
- La estructura completa deberá guardarse en disco, para el cual se tiene un tamaño de página de 4kb (calcular ranueras por balde, baldes y desperdicio si tuviera).
- Se debe utilizar la siguiente función de hashing:

```
int hashing (int dni, int M) {
        char x[8];
        int longitud, i;
        int contador=0;
        sprintf(x,''%d'', dni);
        longitud=strlen(x);
        for (i=0; i< longitud; i++)
                contador+=((int)x[i]) * (i+1);

        return (contador %M );
}
```

- El programa deberá desarrollarse en Lenguaje C, 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**. También puede ser entregado un archivo comprimido respetando la nomenclatura anterior. En cualquier caso no debe entregarse el archivo ejecutable (.exe) sólo los archivos con el código fuente.

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()
{
        FILE *fp;
        if (( fp = fopen ( ''Vendedores.txt'' , ''r'') )==NULL)
                return 0;
        else {
                while(!(feof(fp))){
                        fscanf(fp,''%d'',&aux.dni);
                        fscanf(fp,'' %[^\n]'',aux.Nombre);
                        fscanf(fp,'' %[^\n]'',aux.telefono);
                        fscanf(fp,''%f'',&aux.Valor);
                        fscanf(fp,''%d'',&aux.Cant);
                        fscanf(fp,'' %[^\n]'',aux.tipoventa);
                        /* Donde aux tiene los campos adecuados que se
                        corresponden con la información guardada en el
                        archivo Vendedores.txt en la posición corriente */
                        .
                        .
                        .
                        /* Invocar los procedimientos que correspondan */
                        .
                        .
                        .
                        }
                return 1;
        }
        fclose(fp);
}
```