UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA
Facultad de arquitectura, ingeniería y diseño.(FIAD).

Actividad 13: MENÚS Y ARCHIVOS BINARIOS .

Alumna: Luisa Lizeth Zerega Soto.

Materia: Laboratorio de Programación Estructurada.

Clave: 36276.

Fecha de entrega: 24 de Noviembre del 2023.

Matricula: 356491.

Maestro: Pedro Nunez Yepiz .

Tema-Unidad: archivo binario o de archivo texto

Introducción.
Hay dos tipos de archivos, archivos de texto y archivos binarios. Un archivo de texto es una secuencia de caracteres dispuestos en líneas terminadas por un carácter de nueva línea. En estos archivos puedes almacenar música, fuentes de programas, bases de datos simples, etc. Los archivos de texto se caracterizan por ser planos, es decir, que todas las letras tengan la misma forma y no haya palabras subrayadas, negritas o mayúsculas.
diferente tamaño o ancho.

Competencia.
El alumno utilizará archivos para almacenar información permanente en su disco.

Fundamentos.
la memoria principal., L. D. Q. H. T. H. el M. H. R. en, & Embargo, S. (n.d.). 7. Manejo de Archivos en C. Ual.Es. Retrieved November 12, 2023, from https://w3.ual.es/~abecerra/ID/archivos#:~:text=Un%20archivo%20de%20texto%20es,un%20car%C3%A1cter%20de%20nueva%20l%C3%ADnea.

Procedimiento.
REALICE EL SIGUIENTE PROGRAMA QUE CONTENGA UN MENÚ.

MENÚ

1.- AGREGAR (AUTOM 100 REGISTROS)

2.- EDITAR REGISTRO

3.- ELIMINAR REGISTRO (lógico)

4.- BUSCAR

5.- ORDENAR

6- IMPRIMIR

7.- GENERAR ARCHIVO TEXTO

8.- VER ARCHIVO TEXTO

9.- CREAR ARCH BINARIO

10.- CARGAR ARCH BINARIO

11.- MOSTRAR ELIMINADOS

0.- SALIR

UTILIZAR UN ARREGLO DE 5000 REGISTROS

SE DEBERÁ UTILIZAR ESTRUCTURAS CON LOS DATOS BÁSICOS DE UN EMPLEADO

preguntar nombre de archivo binario o de archivo texto

Busqueda y Ordenacion por CAMPO LLAVE

nota: usar librería propia con funciones

nota2: 100 % validado, Cuidar desbordamiento de vector

nota3: Campo llave matricula no repetido, archivos solo cargar 1 sola vez.

nota4: Usar el tipo Tkey para hacer mas practico el programa

AYUDA:

https://www.fing.edu.uy/tecnoinf/mvd/cursos/prinprog/material/teo/prinprog-teorico08.pdf

https://kesquivel.files.wordpress.com/2013/05/estructuras2013final2.pdf

http://platea.pntic.mec.es/vgonzale/cyr_0204/cyr_01/control/lengua_C/estructuras.htm

INSTRUCCIONES DEL MENU

1.- Agregar : El programa deberá ser capaz de agregar 100 registros al vector de registros (Generar automáticamente los datos).

2.- Editar Registro : El programa deberá buscar una matrícula en el vector por medio del método de búsqueda más óptimo. Mostrar los datos en forma de registro Preguntar que campo quire Editar, actualiar los datos en el vector (solo a registros activos)

3.- Eliminar Registro : El programa deberá buscar una matrícula en el vector por medio del método de búsqueda más óptimo. Utilizar banderas para

escoger el método más adecuado., imprimir el registro y preguntar si se quiere eliminar el registro.

4.- Buscar : El programa deberá buscar una matrícula en el vector por medio del método de búsqueda más óptimo. Utilizar banderas para escoger el método más adecuado. Mostrar los datos en forma de registro

5.- Ordenar : El programa deberá ordenar el vector por medio del método de ordenación más óptimo. Utilizar banderas para escoger el método más adecuado se ordenará por el campo llave (matrícula)

6.- Imprimir: El programa deberá mostrar todos los registros del vector y como están en ese momento ordenado o desordenado. (mostrar en forma de tabla)

7.- Generar Archivo Texto : El programa deberá preguntar al usuario el nombre del archivo, solo nombre sin extensión, el programa generará un archivo con el nombre proporcionado por el usuario con extensión .txt los datos que pondrá en el archivo de texto serán idénticos a los contenidos en el Vector de registros. (ordenado o desordenado). El programa podrá generar múltiples archivos para comprobar las salidas.

8.- Mostrar Archivo Texto: El programa deberá preguntar al usuario el nombre del archivo, solo nombre sin extensión, el programa generará un

archivo con el nombre proporcionado por el usuario con extensión .txt mostrar el archivo de texto tal y como se encuantra.

9.- Crear archivo binario : El programa deberá crear un archivo binario con los datos del vector actualizados, sustituir el archivo base, realizar respaldo del archivo anterior y guardarlo con el mismo nombre pero extencion .tmp (validar msges si el archivo no se puede crear por falta de registros en el vector)

10 .- Cargar Archivo Binario : El programa deberá cargar al vector los registros del archivo binario (solo podrá cargarse una sola vez el archivo, el archivo binario se debara llamar datos.dll y si no existe debera indicar)

11.- Mostrar Borrados: El programa deberá mostrar del archivo binario solo los registros que se eliminaron (marcados con status 0) y que fueron marcados en su momento como registros eliminados.

Resultados.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_REGISTROS 5000

typedef struct {
    int status;
    char nombre[50];
    char matricula[50];
    char apPat[50];
    char apMat[50];
    int edad;
```

```
    char sexo;
} Empleado;

typedef struct {
    Empleado registros[MAX_REGISTROS];
    int cantidad;
} VectorRegistros;

void swap(Empleado *x, Empleado *y);

void agregarRegistros(VectorRegistros *vector);
void editarRegistro(VectorRegistros *vector);
void eliminarRegistro(VectorRegistros *vector);
void buscarRegistro(VectorRegistros *vector);
void ordenarRegistros(VectorRegistros *vector);
void quicksort(Empleado arr[], int inicio, int fin);
int particion(Empleado arr[], int inicio, int fin);
void imprimirRegistros(VectorRegistros *vector);
void generarArchivoTexto(VectorRegistros *vector);
void verArchivoTexto();
void crearArchivoBinario(VectorRegistros *vector);
void cargarArchivoBinario(VectorRegistros *vector);
void mostrarEliminados();

int main() {
    VectorRegistros vector;
    vector.cantidad = 0;

    int opcion;
    do {
        printf("MENÚ\n");
        printf("1. Agregar\n");
        printf("2. Editar Registro\n");
        printf("3. Eliminar Registro\n");
        printf("4. Buscar\n");
        printf("5. Ordenar\n");
        printf("6. Imprimir\n");
        printf("7. Generar Archivo Texto\n");
        printf("8. Mostrar Archivo Texto\n");
        printf("9. Crear Archivo Binario\n");
        printf("10. Cargar Archivo Binario\n");
        printf("11. Mostrar Eliminados\n");
        printf("0. Salir\n");
```

```
        printf("Ingrese una opción: ");
        scanf("%d", &opcion);

        switch (opcion) {
        case 1:
            agregarRegistros(&vector);
            break;
        case 2:
            editarRegistro(&vector);
            break;
        case 3:
            eliminarRegistro(&vector);
            break;
        case 4:
            buscarRegistro(&vector);
            break;
        case 5:
            ordenarRegistros(&vector);
            break;
        case 6:
            imprimirRegistros(&vector);
            break;
        case 7:
            generarArchivoTexto(&vector);
            break;
        case 8:
            verArchivoTexto();
            break;
        case 9:
            crearArchivoBinario(&vector);
            break;
        case 10:
            cargarArchivoBinario(&vector);
            break;
        case 11:
            mostrarEliminados();
            break;
        case 0:
            printf("Saliendo del programa...\n");
            break;
        default:
            printf("Opción inválida. Intente nuevamente.\n");
            break;
```

```
        }

        printf("\n");
    } while (opcion != 0);

    return 0;
}

void agregarRegistros(VectorRegistros *vector) {
    for (int i = 0; i < 100; i++) {
        Empleado empleado;
        empleado.status = i + 1;
        sprintf(empleado.nombre, "Empleado %d: Fulano");
        sprintf(empleado.matricula, "Matricula %d:", i + 1);
        sprintf(empleado.apPat, "Apellido Paterno %d:Bolillo", i + 1);
        sprintf(empleado.apMat, "Apellido Materno %d:De Frijol", i +
1);
        empleado.edad = 25;
        empleado.sexo = 'M';
        vector->registros[vector->cantidad++] = empleado;
    }
}

void editarRegistro(VectorRegistros *vector) {
    int status;
    printf("Ingrese la matrícula del empleado a editar: ");
    scanf("%d", &status);
    for (int i = 0; i < vector->cantidad; i++) {
        if (vector->registros[i].status == status) {
            printf("Datos del empleado:\n");
            printf("Matrícula: %d\n", vector->registros[i].status);
            printf("Nombre: %s\n", vector->registros[i].nombre);
            printf("Matrícula: %s\n", vector->registros[i].matricula);
            printf("Apellido Paterno: %s\n",
vector->registros[i].apPat);
            printf("Apellido Materno: %s\n",
vector->registros[i].apMat);
            printf("Edad: %d\n", vector->registros[i].edad);
            printf("Sexo: %c\n", vector->registros[i].sexo);
            return;
        }
    }
```

```
    printf("No se encontró un empleado con la matrícula ingresada.\n");
}

void eliminarRegistro(VectorRegistros *vector) {
    int status;
    printf("Ingrese la matrícula del empleado a eliminar: ");
    scanf("%d", &status);
    for (int i = 0; i < vector->cantidad; i++) {
        if (vector->registros[i].status == status) {
            printf("Datos del empleado:\n");
            printf("Matrícula: %d\n", vector->registros[i].status);
            printf("Nombre: %s\n", vector->registros[i].nombre);
            printf("Matrícula: %s\n", vector->registros[i].matricula);
            printf("Apellido Paterno: %s\n",
vector->registros[i].apPat);
            printf("Apellido Materno: %s\n",
vector->registros[i].apMat);
            printf("Edad: %d\n", vector->registros[i].edad);
            printf("Sexo: %c\n", vector->registros[i].sexo);
            return;
        }
    }

    printf("No se encontró un empleado con la matrícula ingresada.\n");
}

void buscarRegistro(VectorRegistros *vector) {
    int status;
    printf("Ingrese la matrícula del empleado a buscar: ");
    scanf("%d", &status);

    for (int i = 0; i < vector->cantidad; i++) {
        if (vector->registros[i].status == status) {
            printf("Datos del empleado:\n");
            printf("Matrícula: %d\n", vector->registros[i].status);
            printf("Nombre: %s\n", vector->registros[i].nombre);
            printf("Matrícula: %s\n", vector->registros[i].matricula);
            printf("Apellido Paterno: %s\n",
vector->registros[i].apPat);
            printf("Apellido Materno: %s\n",
vector->registros[i].apMat);
            printf("Edad: %d\n", vector->registros[i].edad);
            printf("Sexo: %c\n", vector->registros[i].sexo);
```

```
            return;
        }
    }

    printf("No se encontró un empleado con la matrícula ingresada.\n");
}

void ordenarRegistros(VectorRegistros *vector) {
    quicksort(vector->registros, 0, vector->cantidad - 1);
}

void quicksort(Empleado arr[], int inicio, int fin) {
    int i, j, pivote;

    if (inicio < fin) {
        pivote = particion(arr, inicio, fin);
        quicksort(arr, inicio, pivote - 1);
        quicksort(arr, pivote + 1, fin);
    }
}

int particion(Empleado arr[], int inicio, int fin) {
    char pivote[50];
    strcpy(pivote, arr[fin].matricula);

    int i = (inicio - 1);

    for (int j = inicio; j <= fin - 1; j++) {
        if (strcmp(arr[j].matricula, pivote) <= 0) {
            i++;
            swap(&arr[i], &arr[j]);
        }
    }

    swap(&arr[i + 1], &arr[fin]);

    return (i + 1);
}

void swap(Empleado *x, Empleado *y) {
    Empleado temp = *x;
    *x = *y;
    *y = temp;
```

```
}

void imprimirRegistros(VectorRegistros *vector) {
    printf("Registros de empleados:\n");
    for (int i = 0; i < vector->cantidad; i++) {
        printf("Matrícula: %d\n", vector->registros[i].status);
        printf("Nombre: %s\n", vector->registros[i].nombre);
        printf("Matrícula: %s\n", vector->registros[i].matricula);
        printf("Apellido Paterno: %s\n", vector->registros[i].apPat);
        printf("Apellido Materno: %s\n", vector->registros[i].apMat);
        printf("Edad: %d\n", vector->registros[i].edad);
        printf("Sexo: %c\n", vector->registros[i].sexo);
        printf("\n");
    }
}

void generarArchivoTexto(VectorRegistros *vector) {
    char nombreArchivo[50];
    printf("Ingrese el nombre del archivo de texto (sin extensión): ");
    scanf("%s", nombreArchivo);

    char nombreCompleto[60];
    sprintf(nombreCompleto, "%s.txt", nombreArchivo);

    FILE *archivo = fopen(nombreCompleto, "w");
    if (archivo == NULL) {
        printf("No se pudo crear el archivo de texto.\n");
        return;
    }

    for (int i = 0; i < vector->cantidad; i++) {
        fprintf(archivo, "Matrícula: %d\n",
vector->registros[i].status);
        fprintf(archivo, "Nombre: %s\n", vector->registros[i].nombre);
        fprintf(archivo, "Matrícula: %s\n",
vector->registros[i].matricula);
        fprintf(archivo, "Apellido Paterno: %s\n",
vector->registros[i].apPat);
        fprintf(archivo, "Apellido Materno: %s\n",
vector->registros[i].apMat);
        fprintf(archivo, "Edad: %d\n", vector->registros[i].edad);
        fprintf(archivo, "Sexo: %c\n", vector->registros[i].sexo);
        fprintf(archivo, "\n");
```

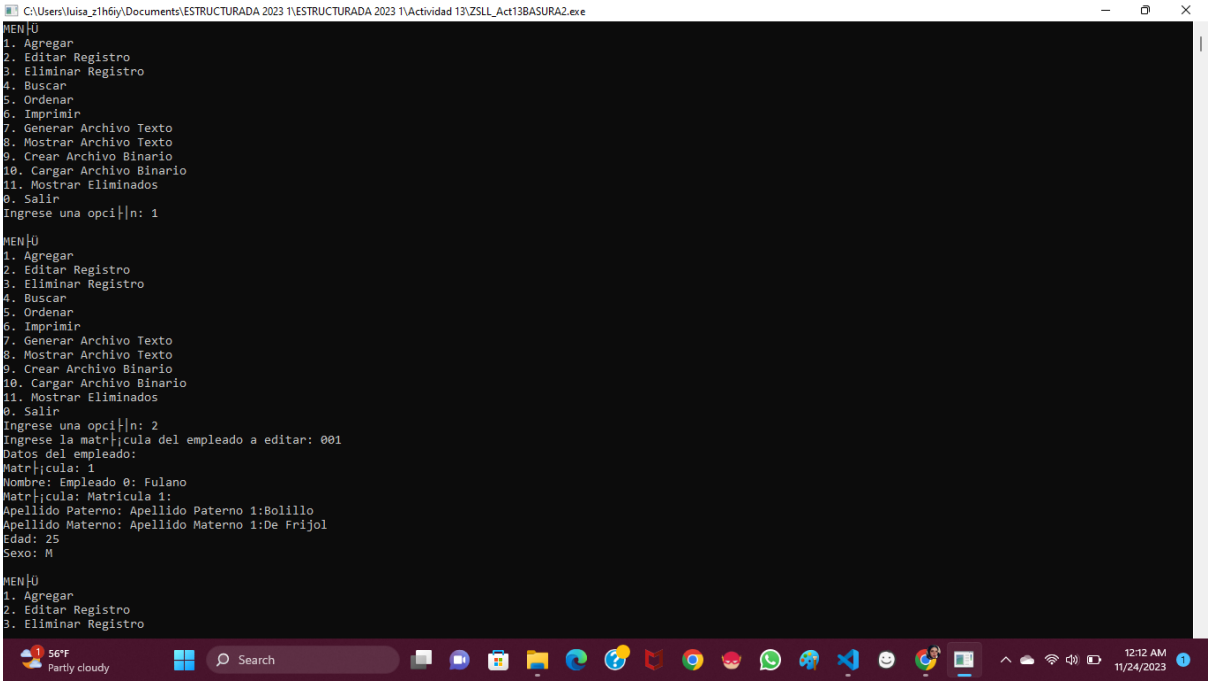
```
    }

    fclose(archivo);
    printf("Se generó el archivo de texto correctamente.\n");
}

void verArchivoTexto() {
    char nombreArchivo[50];
    printf("Ingrese el nombre del archivo de texto (sin extensión): ");
    scanf("%s", nombreArchivo);

    char nombreCompleto[60];
    sprintf(nombreCompleto, "%s.txt", nombreArchivo);

    FILE *archivo = fopen(nombreCompleto, "r");
    if (archivo == NULL) {
        printf("No se pudo abrir el archivo de texto.\n");
        return;
    }

    char linea[100];
    while (fgets(linea, sizeof(linea), archivo)) {
        printf("%s", linea);
    }

    fclose(archivo);
}

void crearArchivoBinario(VectorRegistros *vector) {
    FILE *archivo = fopen("datos.bin", "wb");
    if (archivo == NULL) {
        printf("No se pudo crear el archivo binario.\n");
        return;
    }

    fwrite(vector->registros, sizeof(Empleado), vector->cantidad,
archivo);

    fclose(archivo);
    printf("Se creó el archivo binario correctamente.\n");
}

void cargarArchivoBinario(VectorRegistros *vector) {
```

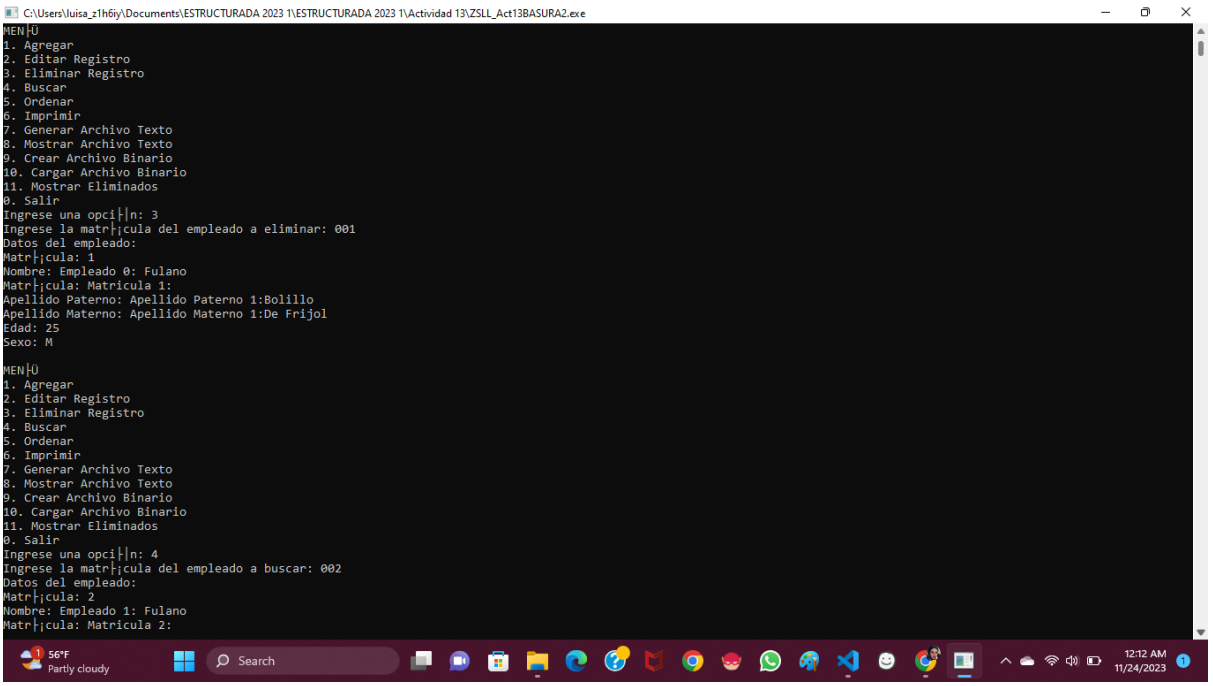
```
    FILE *archivo = fopen("datos.bin", "rb");
    if (archivo == NULL) {
        printf("El archivo binario no existe.\n");
        return;
    }

    fread(vector->registros, sizeof(Empleado), MAX_REGISTROS, archivo);

    fclose(archivo);
    printf("Se cargó el archivo binario correctamente.\n");
}

void mostrarEliminados() {
    FILE *archivo = fopen("datos.bin", "rb");
    if (archivo == NULL) {
        printf("El archivo binario no existe.\n");
        return;
    }

    Empleado empleado;
    while (fread(&empleado, sizeof(Empleado), 1, archivo)) {
        if (empleado.status == 0) {
            printf("Matrícula: %d\n", empleado.status);
            printf("Nombre: %s\n", empleado.nombre);
            printf("Matrícula: %s\n", empleado.matricula);
            printf("Apellido Paterno: %s\n", empleado.apPat);
            printf("Apellido Materno: %s\n", empleado.apMat);
            printf("Edad: %d\n", empleado.edad);
            printf("Sexo: %c\n", empleado.sexo);
            printf("\n");
        }
    }

    fclose(archivo);
}
```

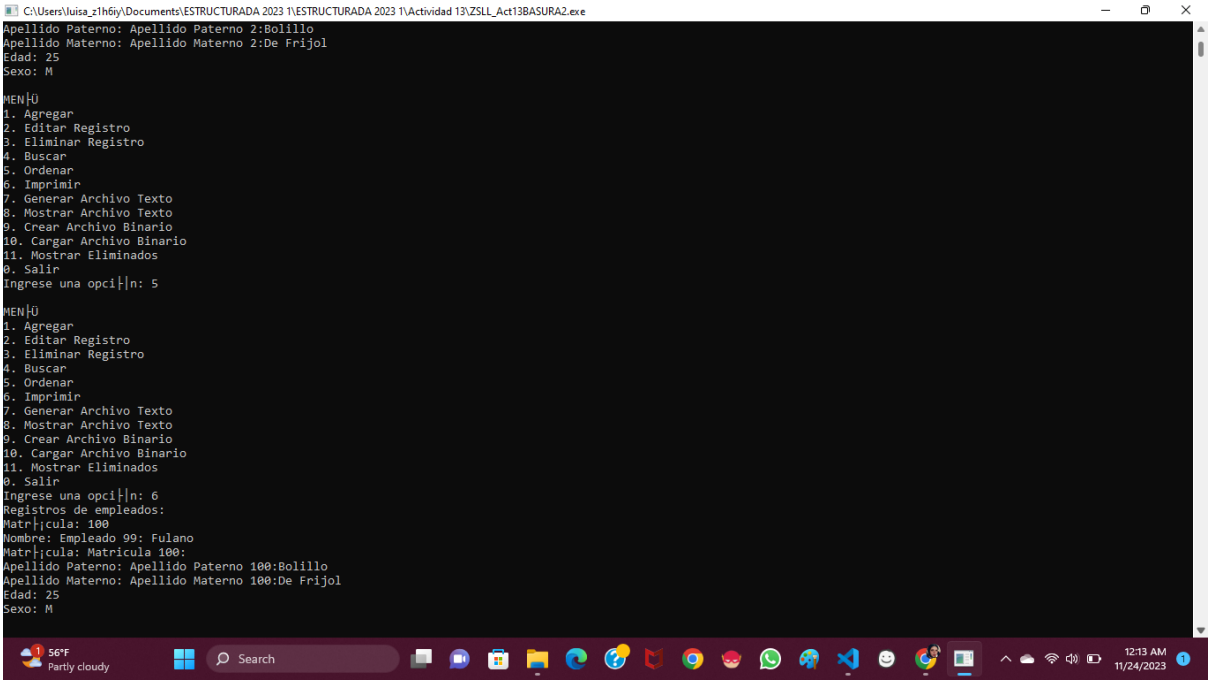
Conclusiones.

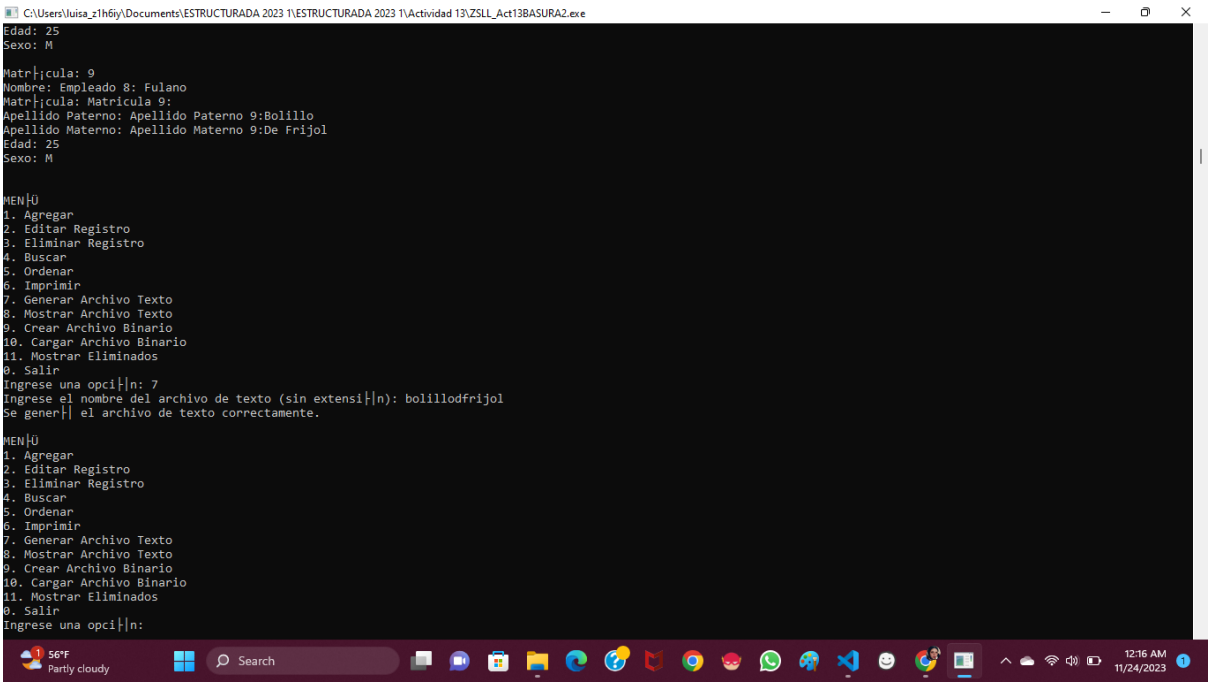
Se aprendió a utilizar funciones de validar números y cadenas para el correcto funcionamiento de datos de caracteres en menús con funciones y métodos de ordenación y búsqueda estructura y librerías.

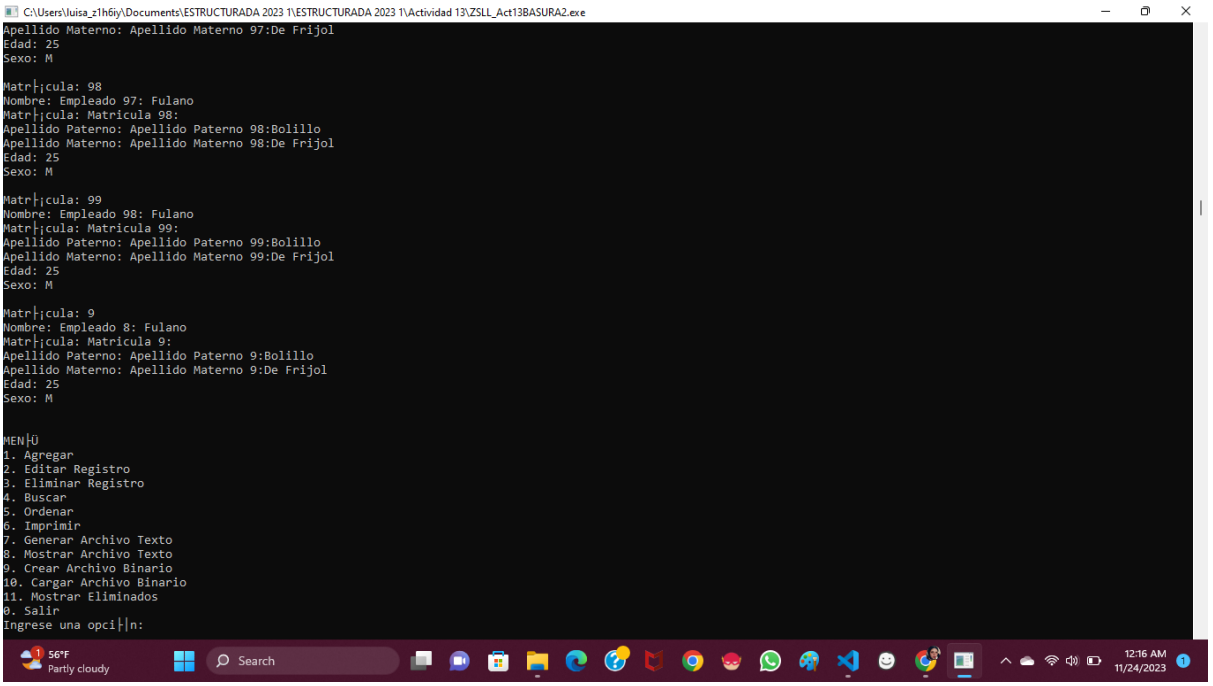
Anexos.






C:\Users\luisa_z1h6iy\Documents\ESTRUCTURADA 2023 1\ESTRUCTURADA 2023 1\Actividad 13\ZSLL_Act13BASURA2.exe
Apellido Paterno: Apellido Paterno 2:Bolillo
Apellido Materno: Apellido Materno 2:De Frijol
Edad: 25
Sexo: M
MEN├Ü
1. Agregar
2. Editar Registro
3. Eliminar Registro
4. Buscar
5. Ordenar
6. Imprimir
7. Generar Archivo Texto
8. Mostrar Archivo Texto
9. Crear Archivo Binario
10. Cargar Archivo Binario
11. Mostrar Eliminados
0. Salir
Ingrese una opci├│n: 5
MEN├Ü
1. Agregar
2. Editar Registro
3. Eliminar Registro
4. Buscar
5. Ordenar
6. Imprimir
7. Generar Archivo Texto
8. Mostrar Archivo Texto
9. Crear Archivo Binario
10. Cargar Archivo Binario
11. Mostrar Eliminados
0. Salir
Ingrese una opci├│n: 6
Registros de empleados:
Matr├¡cula: 100
Nombre: Empleado 99: Fulano
Matr├¡cula: Matricula 100:
Apellido Paterno: Apellido Paterno 100:Bolillo
Apellido Materno: Apellido Materno 100:De Frijol
Edad: 25
Sexo: M
56°F
Partly cloudy
Search
12:13 AM
11/24/2023


C:\Users\luisa_z1h6iy\Documents\ESTRUCTURADA 2023 1\ESTRUCTURADA 2023 1\Actividad 13\ZSLL_Act13BASURA2.exe
Edad: 25
Sexo: M
Matr├¡cula: 9
Nombre: Empleado 8: Fulano
Matr├¡cula: Matricula 9:
Apellido Paterno: Apellido Paterno 9:Bolillo
Apellido Materno: Apellido Materno 9:De Frijol
Edad: 25
Sexo: M
MEN├Ú
1. Agregar
2. Editar Registro
3. Eliminar Registro
4. Buscar
5. Ordenar
6. Imprimir
7. Generar Archivo Texto
8. Mostrar Archivo Texto
9. Crear Archivo Binario
10. Cargar Archivo Binario
11. Mostrar Eliminados
0. Salir
Ingrese una opci├│n: 7
Ingrese el nombre del archivo de texto (sin extensi├│n): bolillodfrijol
Se gener├│ el archivo de texto correctamente.
MEN├Ú
1. Agregar
2. Editar Registro
3. Eliminar Registro
4. Buscar
5. Ordenar
6. Imprimir
7. Generar Archivo Texto
8. Mostrar Archivo Texto
9. Crear Archivo Binario
10. Cargar Archivo Binario
11. Mostrar Eliminados
0. Salir
Ingrese una opci├│n:
56°F
Partly cloudy
Search
12:16 AM
11/24/2023


C:\Users\luisa_z1h6iy\Documents\ESTRUCTURADA 2023 1\ESTRUCTURADA 2023 1\Actividad 13\ZSLL_Act13BASURA2.exe
Apellido Materno: Apellido Materno 97:De Frijol
Edad: 25
Sexo: M
Matr├¡cula: 98
Nombre: Empleado 97: Fulano
Matr├¡cula: Matricula 98:
Apellido Paterno: Apellido Paterno 98:Bolillo
Apellido Materno: Apellido Materno 98:De Frijol
Edad: 25
Sexo: M
Matr├¡cula: 99
Nombre: Empleado 98: Fulano
Matr├¡cula: Matricula 99:
Apellido Paterno: Apellido Paterno 99:Bolillo
Apellido Materno: Apellido Materno 99:De Frijol
Edad: 25
Sexo: M
Matr├¡cula: 9
Nombre: Empleado 8: Fulano
Matr├¡cula: Matricula 9:
Apellido Paterno: Apellido Paterno 9:Bolillo
Apellido Materno: Apellido Materno 9:De Frijol
Edad: 25
Sexo: M
MEN├Ú
1. Agregar
2. Editar Registro
3. Eliminar Registro
4. Buscar
5. Ordenar
6. Imprimir
7. Generar Archivo Texto
8. Mostrar Archivo Texto
9. Crear Archivo Binario
10. Cargar Archivo Binario
11. Mostrar Eliminados
0. Salir
Ingrese una opci├│n:
56°F
Partly cloudy
Search
12:16 AM
11/24/2023