```c
/* MejÃ a MartÃ nez, RamÃ³n    372099                                                                                    */
/* 19 de noviembre de 2023                                                                                             */
/* Programa de gestion de empleados usando structs, distintos metodos de busqueda, ordenacion y uso de archivos de texto y binarios   */
/* MMR_ACT13_932                                                                                                       */

#include "MMRLIB.h"

#define REG 5000

/*
typedef struct _empleado{
    int status;
    Tkey key;
    int NoEmpleado;
    char sexo;
    char nombre[LN];
    char apPat[L];
    char apMat[L];
    char puesto[L];
}Tdato;
*/

void menu(void);
int msgs(void);

//case 1
int genReg(Tdato vect[],int i);
void genRegRand(Tdato vect[],int i);
int buscarTelefono(Tdato vect[],int i,double num);
void genPuesto(Tdato vect[],int i);
//case 2
void editarReg(Tdato vect[],int i,int orden);
void editMenu(Tdato vect[],int pos);
void editPuesto(Tdato vect[],int pos);
float validNumF(char txt[],int limi,int lims);
//case 3
int eliminarEmpleado(Tdato vect[],int i,int orden);
//case 4
void buscarEmpleado(Tdato vect[],int i,int orden);
void printEmpleadoReg(Tdato empleado);
//case 5
int ordenEmpleado(Tdato vect[],int i,int ordenM);
//case 6
void printEmpleados(Tdato vect[],int i);
void printEmpleadoTab(Tdato empleado);
//case 7
void genArchivoTxt(Tdato vect[],int i);
//case 8
void printArchTxt(void);
//case 9
void genArchivoBin(Tdato vect[],int i);
//case 10
int cargarArchivoBin(Tdato vect[],int i);
//case 11
void printDel(void);

int main()
{
    srand(time(NULL));
    menu();
    return 0;
}

int msgs(void)
{
    printf(" Menu\n\n");
    printf(" 1) Agregar\n");
    printf(" 2) Editar Reistro\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) Ver Archivo Texto\n");
    printf(" 9) Crear Archivo Binario\n");
    printf("10) Cargar Archivo Binario\n");
    printf("11) Mostrar Eliminados\n");
    printf("12) Salir\n");
    return (validNum("Escoge una opcion: ",1,12));
}

void menu(void)
{
    FILE *fa;
    Tdato empleados[REG];
    int op;
    int orden=0,ordenMetodo=0,cargBin=1,del=0;
    int i=0;
    fa=fopen("registrosbin.dll","rb");
    if(fa)
    {
        fclose(fa);
        remove("registrosbin.dll");
    }

    do
    {
        system("cls");
        op=msgs();
        system("cls");
        switch(op)
        {
            case 1:
                i=genReg(empleados,i);
                if(i>2500)
                {
                    ordenMetodo=1;
                }
                orden=0;
                break;
            case 2:
                editarReg(empleados,i,orden);
                break;
            case 3:
```

```
                del+=eliminarEmpleado(empleados,i,orden);
                break;
            case 4:
                buscarEmpleado(empleados,i,orden);
                break;
            case 5:
                if(!orden)
                {
                    orden=ordenEmpleado(empleados,i,ordenMetodo);
                }
                else
                {
                    printf("\nLos empleados ya estan ordenados\n");
                }
                break;
            case 6:
                printEmpleados(empleados,i);
                break;
            case 7:
                genArchivoTxt(empleados,i);
                break;
            case 8:
                printArchTxt();
                break;
            case 9:
                genArchivoBin(empleados,i);
                break;
            case 10:
                if(cargBin)
                {
                    i=cargarArchivoBin(empleados,i);
                    if(i>2500)
                    {
                        ordenMetodo=1;
                    }
                    orden=0;
                    cargBin=0;
                }
                else
                {
                    printf("\nYa se cargo el archivo binario\n");
                }
                break;
            case 11:
                if(del>0)
                {
                    printDel();
                }
                else
                {
                    printf("\nAun no hay empleados eliminados\n");
                }
                break;
        }

        printf("\n");
        system("pause");
    }while(op!=12);

    fa=fopen("registrosbin.tmp","rb");
    if(fa)
    {
        fclose(fa);
        remove("registrosbin.tmp");
    }
}

//case 1
int genReg(Tdato vect[],int i)
{
    int j;
    if(i+100<=REG)
    {
        for(j=0;j<100;j++)
        {
            genRegRand(vect,i++);
        }
        printf("\nSe generaron 100 registros\n");
    }
    else
    {
        if(i==REG)
        {
            printf("\nNo se pueden generar mas registros\n");
        }
        else
        {
            if(i+100>REG)
            {
                for(i;i<REG;i++)
                {
                    genRegRand(vect,i);
                }
                printf("\nNo se pueden generar 100 registros\n");
                printf("\nSe generaron registros hasta llegar al limite\n");
            }
        }
    }
    return i;
}

void genRegRand(Tdato vect[],int i)
{
    int s;
    int j;
    double num;
    vect[i].status=1;
    vect[i].NoEmpleado=genKey(vect,i,300000,399999);
    vect[i].key=vect[i].NoEmpleado;

    for(j=0;j<10;j++)
    {
        s=rand()%2;
    }
```

```
    if(s)
    {
        vect[i].sexo='H';
    }
    else
    {
        vect[i].sexo='M';
    }

    if(s)
    {
        genH_Name(vect,i);
    }
    else
    {
        genM_Name(vect,i);
    }

    genAp(vect,i);

    do
    {
        num=6460000000+(rand()%100000)+(rand()%99)*100000;
    }while(buscarTelefono(vect,i,num)!=-1);
    vect[i].telefono=num;

    genPuesto(vect,i);

}

void genPuesto(Tdato vect[],int i)
{
    char pst[12][L]={
        "OPERADOR","SUPERVISOR","MANTENIMIENTO","INGENIERO DE PROCESOS","INGENIERO DE CALIDAD",
        "OPERADOR DE MAQUINARIA", "ALMACENISTA", "JEFE DE ALMACEN", "INSPECTOR DE CALIDAD",
        "GERENTE DE PRODUCCION", "TECNICO EN SEGURIDAD", "RECURSOS HUMANOS"
    };
    strcopy(vect[i].puesto,pst[(rand()%12)]);
}

int buscarTelefono(Tdato vect[],int i,double num)
{
    int j;
    for(j=0;j<i;j++)
    {
        if(vect[j].telefono==num)
        {
            return j;
        }
    }
    return -1;
}

//case 2
void editarReg(Tdato vect[],int i,int orden)
{
    int pos;
    int edit;
    if(i>0)
    {
        printf("Editar empleado\n\n");
        pos=validNum("Ingrese el numero de emplado que desea editar: ",300000,399999);
        if(orden)
        {
            pos=buscarTBin(vect,i,pos);
            if(pos!=-1)
            {
                if(vect[pos].status)
                {
                    printEmpleadoReg(vect[pos]);
                    editMenu(vect,pos);
                }
                else
                {
                    printf("\nEse empleado fue eliminado de los registros\n");
                }
            }
            else
            {
                printf("\nEse empleado no existe\n");
            }
        }
        else
        {
            pos=buscarTSec(vect,i,pos);
            if(pos!=-1)
            {
                if(vect[pos].status)
                {
                    printEmpleadoReg(vect[pos]);
                    editMenu(vect,pos);
                }
                else
                {
                    printf("\nEse empleado fue eliminado de los registros\n");
                }
            }
            else
            {//322010
                printf("\nEse empleado no existe\n");
            }
        }
    }
    else
    {
        printf("\nAun no hay empleados registrados\n");
    }
}

void editMenu(Tdato vect[],int pos)
{
    int op;
    int s;
```

```
    int j;
    char num[11];
    printf("\nEditar\n\n");
    printf("1) Puesto\n");
    printf("2) Nombre\n");
    printf("3) Apellido Paterno\n");
    printf("4) Apellido Materno\n");
    printf("5) Telefono\n");
    printf("6) Sexo\n");
    printf("7) Salir\n");
    op=validNum("Esoge una opcion: ",1,7);
    system("cls");
    switch(op)
    {
        case 1:
            printf("Editar puesto\n\n");
            editPuesto(vect,pos);
            printf("\nSe edito el campo deseado\n");
            break;
        case 2:
            printf("Editar nombre\n\nIngresa el nuevo nombre: ");
            validStr(vect[pos].nombre);
            printf("\nSe edito el campo deseado\n");
            break;
        case 3:
            printf("Editar apellido paterno\n\nIngresa el nuevo apellido: ");
            validStr(vect[pos].apPat);
            printf("\nSe edito el campo deseado\n");
            break;
        case 4:
            printf("Editar apellido materno\n\nIngresa el nuevo apellido: ");
            validStr(vect[pos].apMat);
            printf("\nSe edito el campo deseado\n");
            break;
        case 5:
            printf("Editar telefono\n\n");
            vect[pos].telefono+=validNumF("Ingresa el telefono: 646",0,9999999);
            printf("\nSe edito el campo deseado\n");
            break;
        case 6:
            printf("Editar sexo\n\n");
            s=validNum("(1:Hombre 2:Mujer) Elige una opcion: ",1,2);
            if(s==1)
            {
                vect[pos].sexo='H';
            }
            else
            {
                vect[pos].sexo='M';
            }
            printf("\nSe edito el campo deseado\n");
            break;
    }

}
//308895 312946
void editPuesto(Tdato vect[],int pos)
{
    char pst[12][L]={
        "OPERADOR","SUPERVISOR","MANTENIMIENTO","INGENIERO DE PROCESOS","INGENIERO DE CALIDAD",
        "OPERADOR DE MAQUINARIA", "ALMACENISTA", "JEFE DE ALMACEN", "INSPECTOR DE CALIDAD",
        "GERENTE DE PRODUCCION", "TECNICO EN SEGURIDAD", "RECURSOS HUMANOS"
    };

    printf("1) Operador\n");
    printf("2) Supervisor\n");
    printf("3) Mantenimiento\n");
    printf("4) Ingeniero de Procesos\n");
    printf("5) Ingeniero de Calidad\n");
    printf("6) Operador de Maquinaria\n");
    printf("7) Almacenista\n");
    printf("8) Jefe de Almacen\n");
    printf("9) Inspector de Calidad\n");
    printf("10) Gerente de Produccion\n");
    printf("11) Tecnico en Seguridad\n");
    printf("12) Recursos Humanos\n");
    strcopy(vect[pos].puesto,pst[validNum("Escoge una opcion: ",1,12)-1]);
}

float validNumF(char txt[],int limi,int lims)
{
    char r[STR];
    int rfloat;
    do
    {
        printf("%s",txt);
        fflush(stdin);
        gets(r);
        rfloat=atof(r);
        if(rfloat<limi || rfloat>lims)
        {
            printf("Valor invalido, ingresalo de nuevo\n");
        }
    }while(rfloat<limi || rfloat>lims);
    return rfloat;
}

//case 3
int eliminarEmpleado(Tdato vect[],int i,int orden)
{
    int pos;
    int del;
    if(i>0)
    {
        printf("Eliminar empleado\n\n");
        pos=validNum("Ingrese el numero de emplado que desea eliminar: ",300000,399999);
        if(orden)
        {
            pos=buscarTBin(vect,i,pos);
            if(pos!=-1)
            {
                if(vect[pos].status)
                {
                    printEmpleadoReg(vect[pos]);
```

```
                    printf("\nDesea eliminar al empleado? (1:Si 2:No)\n");
                    del=validNum("Escoge una opcion: ",1,2);
                    if(del==1)
                    {
                        vect[pos].status=0;
                        return 1;
                    }
                }
                else
                {
                    printf("\nEse empleado ya fue eliminado de los registros\n");
                }
            }
            else
            {
                printf("\nEse empleado no existe\n");
            }
        }
        else
        {
            pos=buscarTSec(vect,i,pos);
            if(pos!=-1)
            {
                if(vect[pos].status)
                {
                    printEmpleadoReg(vect[pos]);
                    printf("\nDesea eliminar al empleado? (1:Si 2:No)\n");
                    del=validNum("Escoge una opcion: ",1,2);
                    if(del==1)
                    {
                        vect[pos].status=0;
                        return 1;
                    }
                }
                else
                {
                    printf("\nEse empleado ya fue eliminado de los registros\n");
                }
            }
            else
            {
                printf("\nEse empleado no existe\n");
            }
        }
    }
    else
    {
        printf("\nAun no hay empleados registrados\n");
    }
    return 0;
}

//case 4
void buscarEmpleado(Tdato vect[],int i,int orden)
{
    int pos;
    if(i>0)
    {
        printf("Buscar empleado\n\n");
        pos=validNum("Ingrese el numero de empleado que desea buscar: ",300000,399999);
        if(orden)
        {
            pos=buscarTBin(vect,i,pos);
            if(pos!=-1)
            {
                if(vect[pos].status)
                {
                    printEmpleadoReg(vect[pos]);
                }
                else
                {
                    printf("\nEse empleado fue eliminado de los registros\n");
                }
            }
            else
            {
                printf("\nEse empleado no existe\n");
            }
        }
        else
        {
            pos=buscarTSec(vect,i,pos);
            if(pos!=-1)
            {
                if(vect[pos].status)
                {
                    printEmpleadoReg(vect[pos]);
                }
                else
                {
                    printf("\nEse empleado fue eliminado de los registros\n");
                }
            }
            else
            {
                printf("\nEse empleado no existe\n");
            }
        }
    }
    else
    {
        printf("\nAun no hay empleados registrados\n");
    }
}

void printEmpleadoReg(Tdato empleado)
{
    printf("\nNo. Empleado: %d\n",empleado.NoEmpleado);
    printf("Puesto: %s\n",empleado.puesto);
    printf("Apellido Paterno: %s\n",empleado.apPat);
    printf("Apellido Materno: %s\n",empleado.apMat);
    printf("Nombre: %s\n",empleado.nombre);
    printf("Telefono: %.0f\n",empleado.telefono);
    printf("Sexo: ");
```

```
    if(empleado.sexo=='H')
    {
        printf("MASCULINO");
    }
    else
    {
        printf("FEMENINO");
    }
    printf("\n");
}

//case 5
int ordenEmpleado(Tdato vect[],int i,int ordenM)
{
    if(i>0)
    {
        if(ordenM)
        {
            mergeSortT(vect,0,i-1);
        }
        else
        {
            heapSortT(vect,i);
        }
        printf("\nSe ordenaron los empleados\n");
        return 1;
    }
    else
    {
        printf("\nAun no hay empleado registrados\n");
        return 0;
    }
}

//case 6
void printEmpleados(Tdato vect[],int i)
{
    int j,k=0;
    if(i>0)
    {
        printf("| No   | No.Empleado |         Puesto         | Apellido Paterno | Apellido Materno | Nombre              |  Telefono  |  Sexo     |\n");
        for(j=0;j<i;j++)
        {
            if(vect[j].status)
            {
                printf("| %4d ",k);
                printEmpleadoTab(vect[j]);
                k++;
            }
        }
    }
    else
    {
        printf("\nAun no hay empleado registrados\n");
    }
}

void printEmpleadoTab(Tdato empleado)
{
    printf("| %11d | %-22s | %-16s | %-17s| %-20s | %.0f |",empleado.NoEmpleado,empleado.puesto,empleado.apPat,empleado.apMat,empleado.nombre,empleado.telefono);
    if(empleado.sexo=='H')
    {
        printf(" MASCULINO |\n");
    }
    else
    {
        printf(" FEMENINO  |\n");
    }
}

//case 7
void genArchivoTxt(Tdato vect[],int i)
{
    FILE *fa;
    int j,k;
    int len;
    char name[L];
    printf("Generar archivo\n\n");
    if(i>0)
    {
        printf("Ingresa el nombre del archivo de texto: ");
        gets(name);
        len=strlen(name);
        name[len++]='.';
        name[len++]='t';
        name[len++]='x';
        name[len++]='t';
        name[len]='\0';

        fa=fopen(name,"r");
        if(!fa)
        {
            fa=fopen(name,"a");
            fprintf(fa,"| No   | No.Empleado |         Puesto         | Apellido Paterno | Apellido Materno | Nombre              |  Telefono  |  Sexo     |\n");
            k=0;
            for(j=0;j<i;j++)
            {
                if(vect[j].status)
                {
                    fprintf(fa,"| %4d ",k);
                    fprintf(fa,"| %11d | %-22s | %-16s | %-17s| %-20s | %.0f |",vect[j].NoEmpleado,vect[j].puesto,vect[j].apPat,vect[j].apMat,vect[j].nombre,vect[j].telefono);
                    if(vect[j].sexo=='H')
                    {
                        fprintf(fa," MASCULINO |\n");
                    }
                    else
                    {
                        fprintf(fa," FEMENINO  |\n");
                    }
                    k++;
                }
            }
            printf("\nSe genero el archivo: %s\n",name);
        }
```

```
        else
        {
            fclose(fa);
            printf("\nYa existe un archivo con ese nombre\n");
        }
    }
    else
    {
        printf("\nAun no hay empleados registrados\n");
    }
}

//case 8
void printArchTxt(void)
{
    FILE *fa;
    char name[L];
    int len;
    char c;
    printf("Mostrar archivo\n\n");
    printf("Ingresa el nombre del archivo de texto: ");
    gets(name);
    len=strlen(name);
    name[len++]='.';
    name[len++]='t';
    name[len++]='x';
    name[len++]='t';
    name[len]='\0';

    fa=fopen(name,"r");
    if(fa)
    {
        printf("\n %s\n\n",name);
        do
        {
            c=fgetc(fa);
            printf("%c",c);
        }while(!feof(fa));
        fclose(fa);
    }
    else
    {
        printf("\nEl nombre no coincide con un archivo existente\n");
    }
}

//case 9
void genArchivoBin(Tdato vect[],int i)
{
    FILE *fa;
    int j;
    if(i>0)
    {
        fa=fopen("registrosbin.dll","rb");
        if(fa)
        {
            fclose(fa);
            fa=fopen("reigstrosbin.tmp","rb");
            if(fa)
            {
                remove("registros.tmp");
                rename("registrosbin.dll","registros.tmp");
                fa=fopen("registrosbin.dll","wb");
                for(j=0;j<i;j++)
                {
                    fwrite(&vect[j],sizeof(Tdato),1,fa);
                }
                fclose(fa);
            }
            else
            {
                rename("registrosbin.dll","registros.tmp");
                fa=fopen("registrosbin.dll","wb");
                for(j=0;j<i;j++)
                {
                    fwrite(&vect[j],sizeof(Tdato),1,fa);
                }
                fclose(fa);
            }
        }
        else
        {
            fa=fopen("registrosbin.dll","ab");
            for(j=0;j<i;j++)
            {
                fwrite(&vect[j],sizeof(Tdato),1,fa);
            }
            fclose(fa);
        }
        printf("\nSe genero el archivo binario\n");
    }
    else
    {
        printf("\nAun no hay empleados registrados\n");
    }
}

//case 10
int cargarArchivoBin(Tdato vect[],int i)
{
    FILE *fa;
    int j;
    fa=fopen("datos.dll","rb");
    if(fa)
    {
        j=0;
        while(!feof(fa))
        {
            fread(&vect[j],sizeof(vect[j]),1,fa);
            vect[j].key=vect[j].NoEmpleado;
            j++;
        }
        printf("\nSe cargo el archivo binario datos.dll a los registros\n");
        fclose(fa);
```

```
        return j;
    }
    else
    {
        printf("\nEl archivo binario a cargar no existe\n");
    }
    return 0;
}

//case 11
void printDel(void)
{
    FILE *fa;
    Tdato reg;
    int k;
    fa=fopen("registrosbin.dll","rb");
    if(fa)
    {
        printf("\nEmpleados eliminados\n\n");
        printf("| No   | No.Empleado |         Puesto         | Apellido Paterno | Apellido Materno | Nombre              |  Telefono  |  Sexo     |\n");
        k=0;
        while(!feof(fa))
        {
            fread(&reg,sizeof(Tdato),1,fa);
            if(!reg.status)
            {
                printf("| %4d ",k);
                printEmpleadoTab(reg);
                k++;
            }
        }
        fclose(fa);
    }
    else
    {
        printf("\nAun no hay un archivo binario\n");
    }
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>

#define NM 4
#define STR 100

#define L 30
#define LN 50

//** Structs    ***********************************************************************************************

typedef int Tkey;

typedef struct _empleado{
    Tkey key;
    int NoEmpleado;
    char nombre[LN];
    char apPat[L];
    char apMat[L];
    char sexo;
    int status;
    char puesto[L];
    double telefono;
}Tdato;

//** Validacion ***********************************************************************************************

//Valida un numero entero
//Parametros:
//-txt  Cadena a mostrar antes de leer la varia
//-limi Limite inferior aceptado
//-lims Limite superior aceptado
int validNum(char txt[],int limi,int lims);

//Valida una cadena
//Parametros:
//-str  Cadena a validar
//Devuelve:
//-invalido     Numero de errores
//Solo permite letras
//No permite empezar ni terminar con espacios
//No permite dobles pesacios
//Muestra los errores
int validStr(char str[]);

//** Cadenas ***********************************************************************************************

//Concatena una cadena con otra
//Parametros:
//-str2 Cadena donde se juntaran las cadenas
//-str1 Cadena que se juntara a la anterior
void strConcat(char str2[],char str1[]);

//Convierte una cadena a mayusculas
//Parametros:
//-str  Cadena a convertir a mayusculas
void strM(char str[]);

//Convierte una cadena a minusculas
//Paramestros:
//-str  Cadena a convertir a minusculas
void strm(char str[]);

//Regresa la longitud de la cadena
//Parametros:
//-str  Cadena de la que se quiere saber su longitud
int strlen(char str[]);

//Imprime una cadena
//Parametros:
//-str  Cadena que se quiere imprimir
void strprint(char str[]);

//Copia la cadena en la cadena 2
```

```
//Parametros:
//-str  Cadena original
//-str2 Cadena que en la que se copiara str
void strcopy(char str2[],char str[]);

//Valida un numero entero
//Parametros:
//-txt Texto a mostrar al leer el numero
//Devuelve:
//Numero validado
//No requiere un limite para funcionar
int validNumUnLimit(char txt[]);

//** Arreglos ********************************************************************************************************

//** Asignacion **
//Regresan 1, indica que se lleno

//LLena un vector
//Parametros:
//-vect     Vector a llenar
//-n        TamaÃ±o
//-limi     Limite inferior aceptable
//-lims     Limite superior aceptable
int scanVector(int vect[],int n,int limi,int lims);

//Llena un vector con numeros aleatorios sin repetir
//Parametros:
//-vect     Vector a llenar
//-n        TamaÃ±o
//-ri       Rango inicial
//-rf       Rango final
int randVector(int vect[],int n,int ri,int rf);

//LLena una matriz
//Parametros:
//-matriz   Matriz a llenar
//-r        Numero de renglones
//-c        Numero de columnas
//-limi     Limite inferior aceptable
//-lims     Limte superior aceptable
int scanMatriz(int matriz[][NM],int r,int c,int limi,int lims);

//Llena una matriz con numeros aleatorios sin repetir
//Parametros:
//-matriz   Matriz a llenar
//-r        Numero de renglones
//-c        Numero de columnas
//-ri       Rango inicial
//-rf       Rango final
int randMatriz(int matriz[][NM],int r,int c,int ri,int rf);

//Imprimir vector
//Parametros:
//-vect     Vector a imprimir
//-n        TamaÃ±o
void printVector(int vect[],int n);

//Imprimir matriz
//Parametros:
//-matriz   Matriz a imprimir
//-r        Numero de renglones
//-c        Numero de columnas
void printMatriz(int matriz[][NM],int r,int c);

//Busca un valor en el vector
//Parametros:
//-vect     Vector en el que se buscara
//-n        TamaÃ±o
//-num      Numero a buscar
//Devuelve:
//Posicion si encontro el numero
//-1 Si no encontro el numero
int searchVector(int vect[],int n,int num);

//Busca un valor en una matriz
//Parametros:
```

```
//-matriz   Matriz en la que se buscara
//-pos      Guarda la posicion en i y j
//-r        Numero de renglones
//-c        Numero de columnas
//-num      Numero a buscar
//Devuelve:
//Renglon en que se encontro el numero
//-1 Si no encontro el numero
int searchMatriz(int matriz[][NM],int pos[],int r,int c, int num);

//** Metodos de ordenacion ****************************************************************************

void improvedBubbleSort(int vect[],int n);

//** Manejo de structs ********************************************************************************

//** Ordenar structs

//Ordena un vector de structs
//Parametros:
//-vect     El vector a ordenar
//-izq      Debe recibir un 0
//-derch    Debe recibir n-1, n=tamaÃ±o
//No devuelve nada
void mergeSortT(Tdato vect[],int izq,int derch);
void fusionMergeSortT(Tdato vect[],int izq,int medio,int derch);

//Ordena un vector de structs
//Parametros:
//-n TamaÃ±o del vector
//No devuelve nada
void heapSortT(Tdato vect[], int n);
void swapT(Tdato *x,Tdato *y);
void adjustHeapT(Tdato vect[],int n,int raiz);

//** Buscar structs

//Busca por campo key un registro del vector
//El vector debe estar ordenado
//Parametros:
//-vect     Vector de registros
//-i        Numero de registros ocupados
//-num      Numero de key que se desea buscar
//Retorno:
//-i        Posicion si se encontro
//-1        Si no se encontro
int buscarTBin(Tdato vect[],int i,int num);

//Busca por campo key un registro del vector
//Parametros:
//-vect     Vector de registros
//-i        Numero de registros ocupados
//-num      Numero de key que se desea buscar
//Retorno:
//-i        Posicion si se encontro
//-1        Si no se encontro
int buscarTSec(Tdato vect[],int i,int num);

//** GeneraciÃ³n automatica

//Genera un campo key sin repetir aleatoriamente
//Parametros:
//-vect     Vector de registros
//-i        Numero de registros ocupados
//-ri       Rango inicial
//-rf       Rango final
//Retorno:
//Matricula generada
int genKey(Tdato vect[],int i,int ri,int rf);

//Genera un nombre de hombre aleatoriamente
//Parametros:
//-vect     Vector de registros
//-i        Numero de registros ocupados
//No retorna nigun valor
void genH_Name(Tdato vect[],int i);
```

```
//Genera un nombre de mujer aleatoriamente
//Parametros:
//-vect     Vector de registros
//-i        Numero de registros ocupados
//No retorna nigun valor
void genM_Name(Tdato vect[],int i);

//Genera un apellido aleatoriamente
//Parametros:
//-vect     Vector de registros
//-i        Numero de registros ocupados
//No retorna nigun valor
void genAp(Tdato vect[],int i);

int validNum(char txt[],int limi,int lims)
{
    char r[STR];
    int rint;
    do
    {
        printf("%s",txt);
        fflush(stdin);
        gets(r);
        rint=atoi(r);
        if(rint<limi || rint>lims)
        {
            printf("Valor invalido, ingresalo de nuevo\n");
        }
    }while(rint<limi || rint>lims);
    return rint;
}

int validStr(char str[])
{
    int i=0,invalido=0;
    fflush(stdin);
    gets(str);
    do
    {
        if(str[i]>90)
        {
            if(str[i]<97)
            {
                printf("\nEl simbolo %c no es valido \n",str[i]);
                invalido++;
            }
        }
        else
        {
            if(str[i]>122)
            {
                printf("\nEl simbolo %c no es valido \n",str[i]);
                invalido++;
            }
            else
            {
                if(str[i]<65)
                {
                    if(str[i]==' ')
                    {
                        if(i==0)
                        {
                            printf("\nLa cadena no debe empezar con espacio\n");
                            invalido++;
                        }
                        else
                        {
                            if(str[i-1]==' ')
                            {
                                printf("\nLa cadena no debe llevar dos espacios seguidos\n");
                                invalido++;
                            }
                        }
                    }
                    else
                    {
                        if(str[i]=='\0')
```

```
                        {
                            if(str[i-1]==' ')
                            {
                                printf("\nLa cadena no debe terminar en espacio\n");
                                invalido++;
                            }
                        }
                        else
                        {
                            printf("\nEl simbolo %c no es valido \n",str[i]);
                            invalido++;
                        }
                    }
                }
            }
        }
        i++;
    }while(str[i-1]!='\0');
    return invalido;
}

int validNumUnLimit(char txt[])
{
    char r[STR];
    int rint;
    printf("%s",txt);
    fflush(stdin);
    gets(r);
    rint=atoi(r);
    return rint;
}

void strM(char str[])
{
    int i;
    for(i=0;str[i]!='\0';i++)
    {
        if(str[i]>=97)
        {
            if(str[i]<=122)
            {
                str[i]-=32;
            }
        }
    }
}

void strm(char str[])
{
    int i;
    for(i=0;str[i]!='\0';i++)
    {
        if(str[i]>=65)
        {
            if(str[i]<=90)
            {
                str[i]+=32;
            }
        }
    }
}

int strlen(char str[])
{
    int i=0;
    while(str[i]!='\0')
    {
        i++;
    }
    return i;
}

void strprint(char str[])
{
    int i;
    for(i=0;str[i]!='\0';i++)
    {
```

```
        printf("%c",str[i]);
    }
}

void strcopy(char str2[],char str[])
{
    int i=0;
    for(i=0;str[i]!='\0';i++)
    {
        str2[i]=str[i];
    }
    str2[i]='\0';
}

int scanVector(int vect[],int n,int limi,int lims)
{
    int i;
    for(i=0;i<n;i++)
    {
        printf("Ingresa el valor de la posicion: %d\n",i+1);
        vect[i]=validNum("",limi,lims);
    }
    return 1;
}

int randVector(int vect[],int n,int ri,int rf)
{
    int i,num,range;
    srand(time(NULL));
    range=(rf-ri)+1;
    for(i=0;i<n;i++)
    {
        do
        {
            num=(rand()%range)+ri;
        }while(searchVector(vect,n,num)!=-1);
        vect[i]=num;
    }
    return 1;
}

int scanMatriz(int matriz[][NM],int r,int c,int limi,int lims)
{
    int i,j;
    for(i=0;i<r;i++)
    {
        for(j=0;j<c;j++)
        {
            printf("Ingresa el valor de la posicion: [%d][%d]\n",i+1,j+1);
            matriz[i][j]=validNum("",limi,lims);
        }
    }
    return 1;
}

int randMatriz(int matriz[][NM],int r,int c,int ri,int rf)
{
    int i,j,k;
    int n;
    n=r*c;
    int vect[n];
    randVector(vect,n,ri,rf);
    k=0;
    for(i=0;i<r;i++)
    {
        for(j=0;j<r;j++)
        {
            matriz[i][j]=vect[k];
            k++;
        }
    }
    return 1;
}

void printVector(int vect[],int n)
{
    int i;
```

```
    for(i=0;i<n;i++)
    {
        printf("%d ",vect[i]);
    }
}

void printMatriz(int matriz[][NM],int r,int c)
{
    int i,j;
    for(i=0;i<r;i++)
    {
        for(j=0;j<c;j++)
        {
            printf("%4d ",matriz[i][j]);
        }
        printf("\n");
    }
}

int searchVector(int vect[],int n,int num)
{
    int i;
    for(i=0;i<n;i++)
    {
        if(vect[i]==num)
        {
            return i;
        }
    }
    return -1;
}

int searchMatriz(int matriz[][NM],int pos[],int r,int c,int num)
{
    int i,j;
    for(i=0;i<r;i++)
    {
        for(j=0;j<c;j++)
        {
            if(matriz[i][j]==num)
            {
                pos[0]=i;
                pos[1]=j;
                return i;
            }
        }
    }
    return -1;
}

void improvedBubbleSort(int vect[],int n)
{
    int i,j;
    int temp,swap;
    for(i=0;i<n-1;i++)
    {
        swap=0;
        for(j=0;j<n-i-1;j++)
        {
            if(vect[j]>vect[j+1])
            {
                temp=vect[j];
                vect[j]=vect[j+1];
                vect[j+1]=temp;
                swap=1;
            }
        }
        if (swap==0)
        {
            i=n;
        }
    }
}

void strConcat(char str2[],char str1[])
{
    int j,k=0;
```

```
    j=strlen(str2);
    str2[j++]=' ';
    while(str1[k]!='\0')
    {
        str2[j]=str1[k];
        k++;
        j++;
    }
    str2[j]='\0';
}

void mergeSortT(Tdato vect[],int izq,int derch)
{
    int med;

    if(izq<derch)
    {
        med=izq+(derch-izq)/2;
        mergeSortT(vect,izq,med);
        mergeSortT(vect,med+1,derch);
        fusionMergeSortT(vect,izq,med,derch);
    }
}

void fusionMergeSortT(Tdato vect[],int izq,int medio,int derch)
{
    int i,j,k;
    int n1=medio-izq+1;
    int n2=derch-medio;
    Tdato Lf[n1], R[n2];

    for (i=0;i<n1;i++)
    {
        Lf[i]=vect[izq+i];
    }
    for (j=0;j<n2;j++)
    {
        R[j]=vect[medio+1+j];
    }

    i=0;
    j=0;
    k=izq;
    while (i<n1)
    {
        if(j<n2)
        {
            if(Lf[i].key<=R[j].key)
            {
                vect[k]=Lf[i];
                i++;
            }
            else
            {
                vect[k]=R[j];
                j++;
            }
            k++;
        }
        else
        {
            break;
        }
    }
    while(i<n1)
    {
        vect[k]=Lf[i];
        i++;
        k++;
    }
    while (j < n2)
    {
        vect[k]=R[j];
        j++;
        k++;
    }
}
```

```
void swapT(Tdato *x,Tdato *y)
{
    Tdato temp = *x;
    *x = *y;
    *y = temp;
}

void adjustHeapT(Tdato vect[],int n,int raiz)
{
    int may=raiz;
    int izq =2 *raiz+1;
    int derch =2*raiz+2;

    if(izq<n)
    {
        if(vect[izq].key>vect[may].key)
        {
            may=izq;
        }
    }

    if(derch<n)
    {
        if(vect[derch].key>vect[may].key)
        {
            may=derch;
        }

    }

    if(may!=raiz)
    {
        swapT(&vect[raiz],&vect[may]);
        adjustHeapT(vect,n,may);
    }
}

void heapSortT(Tdato vect[],int n)
{
    int i;
    for (i=n/2-1;i>=0;i--)
    {
        adjustHeapT(vect,n,i);
    }

    for (i=n-1;i>0;i--)
    {
        swapT(&vect[0],&vect[i]);
        adjustHeapT(vect,i,0);
    }
}

int buscarTBin(Tdato vect[],int i,int num)
{
    int izq,drcha;
    int med;
    izq=0;
    drcha=i-1;
    while(izq<=drcha)
    {
        med=izq+(drcha-izq)/2;
        if(vect[med].key==num)
        {
            return med;
        }

        if(vect[med].key<num)
        {
            izq=med+1;
        }
        else
        {
            drcha=med-1;
        }
    }
    return -1;
```

```
}

int buscarTSec(Tdato vect[],int i,int num)
{
    int j;
    for(j=0;j<i;j++)
    {
        if(num==vect[j].key)
        {
            return j;
        }
    }
    return -1;
}

int genKey(Tdato vect[],int i,int ri,int rf)
{
    int num;
    int rango=rf-ri;
    do
    {
        num=(rand()%rango+1)+ri;
    }while(buscarTSec(vect,i,num)!=-1);
    return num;
}

void genH_Name(Tdato vect[],int i)
{
    char h_name[100][LN]={
        "JOSE",
        "JUAN",
        "LUIS",
        "MANUEL",
        "ANTONIO",
        "JESUS",
        "CARLOS",
        "FRANCISCO",
        "ALBERTO",
        "JORGE",
        "MIGUEL",
        "ANGEL",
        "JAVIER",
        "ALEJANDRO",
        "ENRIQUE",
        "VICTOR",
        "ARTURO",
        "CESAR",
        "FERNANDO",
        "PEDRO",
        "MARTIN",
        "ROBERTO",
        "EDUARDO",
        "MARIO",
        "ARMANDO",
        "SERGIO",
        "RAUL",
        "ALFREDO",
        "RAFAEL",
        "RICARDO",
        "HECTOR",
        "OSCAR",
        "GERARDO",
        "DAVID",
        "DANIEL",
        "HUGO",
        "JAIME",
        "JULIO",
        "RUBEN",
        "RAMON",
        "MARCO",
        "GABRIEL",
        "EDGAR",
        "GUADALUPE",
        "ALFONSO",
        "GUILLERMO",
        "SALVADOR",
        "OMAR",
```

```
        "IVAN",
        "HUMBERTO",
        "FELIPE",
        "ERNESTO",
        "PABLO",
        "IGNACIO",
        "GUSTAVO",
        "ANDRES",
        "ADRIAN",
        "JOEL",
        "AGUSTIN",
        "RODOLFO",
        "GILBERTO",
        "ROGELIO",
        "RENE",
        "TOMAS",
        "SAUL",
        "ISRAEL",
        "OCTAVIO",
        "VICENTE",
        "NOE",
        "GREGORIO",
        "ISMAEL",
        "NICOLAS",
        "BENJAMIN",
        "MOISES",
        "SANTIAGO",
        "EFRAIN",
        "ALONSO",
        "ABEL",
        "WILBERT",
        "ALVARO",
        "FELIX",
        "MARCOS",
        "ADOLFO",
        "RODRIGO",
        "RAMIRO",
        "SAMUEL",
        "JOAQUIN",
        "ABRAHAM",
        "ESTEBAN",
        "ULISES",
        "RAYMUNDO",
        "FIDEL",
        "LORENZO",
        "GERMAN",
        "MAURICIO",
        "LEONARDO",
        "JOSUE",
        "EMMANUEL",
        "JULIAN",
        "SANTOS",
    };
    int prob;
    prob=rand()%10;
    if(prob<=7)
    {
        strcopy(vect[i].nombre,h_name[rand()%100]);
    }
    else
    {
        strcopy(vect[i].nombre,h_name[rand()%100]);
        strConcat(vect[i].nombre,h_name[rand()%100]);
    }
}

void genM_Name(Tdato vect[],int i)
{
    char m_name[100][LN]={
        "MARIA",
        "GUADALUPE",
        "ROSA",
        "MARTHA",
        "ANA",
        "PATRICIA",
        "LETICIA",
        "ELENA",
```

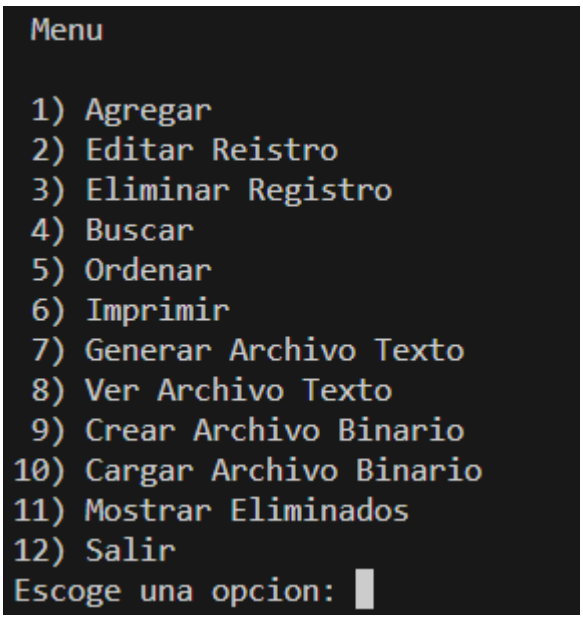
```
        "LAURA",
        "ELIZABETH",
        "ISABEL",
        "ALICIA",
        "MARGARITA",
        "LUZ",
        "ANGELICA",
        "CLAUDIA",
        "SILVIA",
        "NORMA",
        "JUANA",
        "GABRIELA",
        "ADRIANA",
        "VERONICA",
        "TERESA",
        "ALEJANDRA",
        "ALMA",
        "BEATRIZ",
        "YANET",
        "YOLANDA",
        "BLANCA",
        "SANDRA",
        "ARACELI",
        "IRMA",
        "GLORIA",
        "CARMEN",
        "LUISA",
        "ROCIO",
        "CONCEPCION",
        "CRISTINA",
        "EDITH",
        "ESTHER",
        "LILIA",
        "LORENA",
        "MAGDALENA",
        "CECILIA",
        "JOSEFINA",
        "DOLORES",
        "KARINA",
        "LUCIA",
        "ESTELA",
        "SUSANA",
        "MARIBEL",
        "LUCINA",
        "LOURDES",
        "ERIKA",
        "MONICA",
        "AGUSTINA",
        "DIANA",
        "EUGENIA",
        "ALEJANDRINA",
        "PILAR",
        "FRANCISCA",
        "VIRGINIA",
        "LILIANA",
        "GRACIELA",
        "BERTHA",
        "OLGA",
        "MIRIAM",
        "FABIOLA",
        "DELIA",
        "CAROLINA",
        "KARLA",
        "HILDA",
        "ANTONIA",
        "MARICELA",
        "SONIA",
        "NANCY",
        "BERENICE",
        "YADIRA",
        "MAYRA",
        "JUDITH",
        "MARISOL",
        "ROSARIO",
        "REYNA",
        "AURORA",
        "OLIVIA",
```

```
        "RAQUEL",
        "IRENE",
        "ELVIRA",
        "CATALINA",
        "LIDIA",
        "VICTORIA",
        "DULCE",
        "ISELA",
        "SARA",
        "MARCELA",
        "ELVIA",
        "ESPERANZA",
        "GUILLERMINA",
        "DORA",
        "EVA",

    };
    int prob;
    prob=rand()%10;
    if(prob<=7)
    {
        strcopy(vect[i].nombre,m_name[rand()%100]);
    }
    else
    {
        strcopy(vect[i].nombre,m_name[rand()%100]);
        strConcat(vect[i].nombre,m_name[rand()%100]);
    }
}

void genAp(Tdato vect[],int i)
{
    char ap[100][L]={
        "HERNANDEZ",
        "GARCIA",
        "MARTINEZ",
        "LOPEZ",
        "GONZALEZ",
        "RODRIGUEZ",
        "PEREZ",
        "SANCHEZ",
        "RAMIREZ",
        "FLORES",
        "CRUZ",
        "GOMEZ",
        "MORALES",
        "VAZQUEZ",
        "JIMENEZ",
        "REYES",
        "DIAZ",
        "TORRES",
        "GUTIERREZ",
        "RUIZ",
        "AGUILAR",
        "MENDOZA",
        "CASTILLO",
        "ORTIZ",
        "MORENO",
        "RIVERA",
        "RAMOS",
        "ROMERO",
        "JUAREZ",
        "ALVAREZ",
        "MENDEZ",
        "CHAVEZ",
        "HERRERA",
        "MEDINA",
        "DOMINGUEZ",
        "CASTRO",
        "GUZMAN",
        "VARGAS",
        "VELAZQUEZ",
        "SALAZAR",
        "ROJAS",
        "ORTEGA",
        "CORTES",
        "SANTIAGO",
```

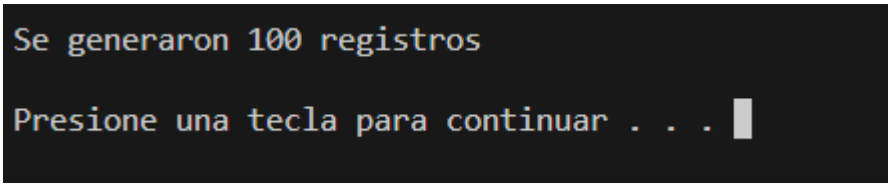
```
        "GUERRERO",
        "CONTRERAS",
        "BAUTISTA",
        "ESTRADA",
        "LUNA",
        "LARA",
        "RIOS",
        "AVILA",
        "ALVARADO",
        "DE LA CRUZ",
        "SILVA",
        "DELGADO",
        "CARRILLO",
        "SOLIS",
        "SOTO",
        "LEON",
        "FERNANDEZ",
        "CERVANTES",
        "MARQUEZ",
        "ESPINOSA",
        "MEJIA",
        "VEGA",
        "SANDOVAL",
        "CAMPOS",
        "NAVA",
        "CABRERA",
        "IBARRA",
        "ESPINOZA",
        "SANTOS",
        "ACOSTA",
        "CAMACHO",
        "VALDEZ",
        "FUENTES",
        "VARELA",
        "MIRANDA",
        "MALDONADO",
        "ROBLES",
        "ROSAS",
        "MEZA",
        "MOLINA",
        "TREJO",
        "ROSALES",
        "PACHECO",
        "NAVARRO",
        "SALGADO",
        "AGUIRRE",
        "SALAS",
        "VELASCO",
        "CARDENAS",
        "PINEDA",
        "OROZCO",
        "SERRANO",
        "RANGEL",
        "VALENCIA",
        "SOSA",
        "VASQUEZ"
    };
    int prob;
    prob=rand()%10;
    if(prob<9)
    {
        strcopy(vect[i].apPat,ap[rand()%99]);
        strcopy(vect[i].apMat,ap[rand()%99]);
    }
    else
    {
        strcopy(vect[i].apPat,ap[rand()%99]);
        vect[i].apMat[0]='X';
        vect[i].apMat[1]='\0';
    }
}
```

Pruebas de escritorio:

Menú:



Opción 1:



Opción 2:

```
Editar empleado

Ingrese el numero de emplado que desea editar: 330631

No. Empleado: 330631
Puesto: INGENIERO DE PROCESOS
Apellido Paterno: JIMENEZ
Apellido Materno: ROBLES
Nombre: RODOLFO JULIAN
Telefono: 6465409683
Sexo: MASCULINO
```

```
Editar

1) Puesto
2) Nombre
3) Apellido Paterno
4) Apellido Materno
5) Telefono
6) Sexo
7) Salir
Esoge una opcion:
```

```
Editar apellido paterno

Ingresa el nuevo apellido: DIAZ

Se edito el campo deseado

Presione una tecla para continuar . . .
```

Opción 3:

```
Eliminar empleado

Ingrese el numero de emplado que desea eliminar: 307711

No. Empleado: 307711
Puesto: SUPERVISOR
Apellido Paterno: DIAZ
Apellido Materno: FERNANDEZ
Nombre: ANTONIO
Telefono: 6464294436
Sexo: MASCULINO

Desea eliminar al empleado? (1:Si 2:No)
Escoge una opcion: 1

Presione una tecla para continuar . . .
```

Opción 4:

```
Buscar empleado

Ingrese el numero de empleado que desea buscar: 325467

No. Empleado: 325467
Puesto: INSPECTOR DE CALIDAD
Apellido Paterno: FLORES
Apellido Materno: GOMEZ
Nombre: CESAR
Telefono: 6467126051
Sexo: MASCULINO

Presione una tecla para continuar . . .
```

Opción 5:

```
Se ordenaron los empleados

Presione una tecla para continuar . . .
```

Opción 6:

No	No.Empleado	Puesto	Apellido Paterno	Apellido Materno	Nombre	Telefono	Sexo
0	318145	OPERADOR DE MAQUINARIA	MOLINA	MIRANDA	GUADALUPE	6464217126	MASCULINO
1	313356	GERENTE DE PRODUCCION	MEZA	CRUZ	PATRICIA CONCEPCION	6463615901	FEMENINO
2	326362	SUPERVISOR	JUAREZ	X	RICARDO	6468412684	MASCULINO
3	321004	INSPECTOR DE CALIDAD	ALVAREZ	GOMEZ	OCTAVIO	6469406720	MASCULINO
4	301440	ALMACENISTA	FERNANDEZ	CAMPOS	ANA ISELA	6462804673	FEMENINO
5	310862	MANTENIMIENTO	VELASCO	SILVA	ROCIO	6461616625	FEMENINO
6	314341	RECURSOS HUMANOS	ACOSTA	MIRANDA	CRISTINA	6460824867	FEMENINO
7	327735	SUPERVISOR	MEZA	CARRILLO	PEDRO	6464403079	MASCULINO
8	300842	INGENIERO DE CALIDAD	RODRIGUEZ	SERRANO	PATRICIA	6466731445	FEMENINO
9	310350	INSPECTOR DE CALIDAD	LARA	MOLINA	LAURA ROSARIO	6466204282	FEMENINO
10	324201	TECNICO EN SEGURIDAD	SOTO	SOLIS	ALFREDO	6469504270	MASCULINO
11	311740	INSPECTOR DE CALIDAD	SANDOVAL	SANTIAGO	ALVARO ALFREDO	6468230907	MASCULINO
12	312562	INGENIERO DE PROCESOS	PACHECO	X	ALMA VICTORIA	6466121887	FEMENINO
13	324045	ALMACENISTA	GUTIERREZ	X	LUISA	6467911130	FEMENINO
14	312860	RECURSOS HUMANOS	RAMOS	MORALES	FRANCISCO	6467915391	MASCULINO
15	302628	INSPECTOR DE CALIDAD	BAUTISTA	X	DELIA	6467003295	FEMENINO
16	326327	MANTENIMIENTO	ESPINOSA	CARRILLO	LUCIA	6461703720	FEMENINO
17	310461	OPERADOR	GOMEZ	X	LILIANA	6462323790	FEMENINO
18	312220	JEFE DE ALMACEN	SANTIAGO	OROZCO	BENJAMIN	6462812349	MASCULINO
19	329097	INGENIERO DE CALIDAD	SILVA	CAMACHO	IRENE	6460804430	FEMENINO
20	324037	ALMACENISTA	MENDEZ	IBARRA	JESUS	6469119767	MASCULINO
21	314716	OPERADOR DE MAQUINARIA	IBARRA	SILVA	MANUEL	6460414814	MASCULINO
22	326651	INGENIERO DE PROCESOS	CARDENAS	AGUIRRE	RODOLFO	6465506326	MASCULINO
23	314127	TECNICO EN SEGURIDAD	LUNA	X	GABRIEL	6468218613	MASCULINO
24	326682	MANTENIMIENTO	CRUZ	MORALES	JUDITH	6467322384	FEMENINO
25	318301	TECNICO EN SEGURIDAD	MARTINEZ	PACHECO	LILIA	6465915702	FEMENINO
26	302270	GERENTE DE PRODUCCION	ALVAREZ	X	AGUSTINA	6460516860	FEMENINO

Opción 7:

```
Generar archivo

Ingresa el nombre del archivo de texto: archivos_texto

Se genero el archivo archivos_texto.txt

Presione una tecla para continuar . . .
```

Opción 8:

```
Mostrar archivo

Ingresa el nombre del archivo de texto: archivos_texto

 archivos_texto.txt
```

No	No.Empleado	Puesto	Apellido Paterno	Apellido Materno	Nombre	Telefono	Sexo
0	321868	INGENIERO DE PROCESOS	CERVANTES	SANDOVAL	CARMEN VICTORIA	6465426843	FEMENINO
1	313188	INGENIERO DE PROCESOS	CERVANTES	TREJO	LUISA	6463600158	FEMENINO
2	328026	MANTENIMIENTO	CONTRERAS	GUZMAN	BEATRIZ	6462507023	FEMENINO
3	312548	INGENIERO DE PROCESOS	CASTRO	ROSAS	CARMEN	6464619229	FEMENINO
4	315293	TECNICO EN SEGURIDAD	RIVERA	CRUZ	ALONSO	6466013043	MASCULINO
5	312433	INGENIERO DE CALIDAD	MEDINA	VEGA	ELVIA BEATRIZ	6460919835	FEMENINO
6	330257	INGENIERO DE CALIDAD	RANGEL	NAVARRO	JOSEFINA IRMA	6469204379	FEMENINO
7	301518	MANTENIMIENTO	ALVARADO	ALVARADO	CRISTINA	6460913485	FEMENINO
8	309559	OPERADOR	MORENO	RODRIGUEZ	IGNACIO	6464504776	MASCULINO
9	319529	JEFE DE ALMACEN	ROJAS	MORENO	RAMIRO TOMAS	6468831981	MASCULINO
10	328339	GERENTE DE PRODUCCION	SOTO	MEJIA	GUSTAVO	6466220652	MASCULINO
11	310812	INGENIERO DE CALIDAD	CASTRO	VALENCIA	AGUSTIN	6463620376	MASCULINO

Opción 9:

```
Se genero el archivo binario

Presione una tecla para continuar . . .
```

Opción 10:

```
Se cargo el archivo binario datos.dll a los registros

Presione una tecla para continuar . . .
```

No	No.Empleado	Puesto	Apellido Paterno	Apellido Materno	Nombre	Telefono	Sexo
0	317876	RECURSOS HUMANOS	SERRANO	X	JUANA	6461709498	FEMENINO
1	330935	INGENIERO DE CALIDAD	HERRERA	GUTIERREZ	CATALINA	6463225311	FEMENINO
2	301660	INSPECTOR DE CALIDAD	VELASCO	NAVARRO	AURORA	6460510621	FEMENINO
3	313489	INGENIERO DE PROCESOS	ROBLES	AVILA	FIDEL	6466515075	MASCULINO
4	325468	INGENIERO DE CALIDAD	RIVERA	ROSALES	VIRGINIA	6466915595	FEMENINO
5	302945	INGENIERO DE PROCESOS	ORTIZ	MARQUEZ	MONICA LOURDES	6461303477	FEMENINO
6	321654	RECURSOS HUMANOS	SANCHEZ	CRUZ	JAIME	6468431374	MASCULINO
7	315653	TECNICO EN SEGURIDAD	VALENCIA	ROSALES	PILAR	6469708446	FEMENINO
8	305971	OPERADOR DE MAQUINARIA	CAMPOS	HERNANDEZ	GERARDO ANDRES	6468722217	MASCULINO
9	320824	INSPECTOR DE CALIDAD	VARELA	ORTIZ	TOMAS	6468300973	MASCULINO
10	325445	JEFE DE ALMACEN	ACOSTA	ALVARADO	MAGDALENA	6460318836	FEMENINO
11	313588	ALMACENISTA	VELAZQUEZ	MENDEZ	PATRICIA ESPERANZA	6466526455	FEMENINO
12	307077	ALMACENISTA	RIOS	SOTO	SONIA IRMA	6461520354	FEMENINO
13	313847	GERENTE DE PRODUCCION	VARELA	ESTRADA	GLORIA	6463927580	FEMENINO
14	311743	MANTENIMIENTO	CASTILLO	SILVA	LUZ	6467815791	FEMENINO
15	315901	SUPERVISOR	FUENTES	MARTINEZ	JOSUE JESUS	6466815423	MASCULINO
16	320307	INGENIERO DE CALIDAD	FUENTES	SALAS	GABRIEL ADOLFO	6464007672	MASCULINO
17	324591	INSPECTOR DE CALIDAD	LOPEZ	FUENTES	RICARDO	6469323224	MASCULINO

Opción 11:

```
Empleados eliminados

| No | No.Empleado | Puesto | Apellido Paterno | Apellido Materno | Nombre | Telefono | Sexo |
| 0 | 308895 | GERENTE DE PRODUCCION | GOMEZ | MARQUEZ | KARINA | 6462308665 | FEMENINO |

Presione una tecla para continuar . . .
```

Archivos generados y para cargar: