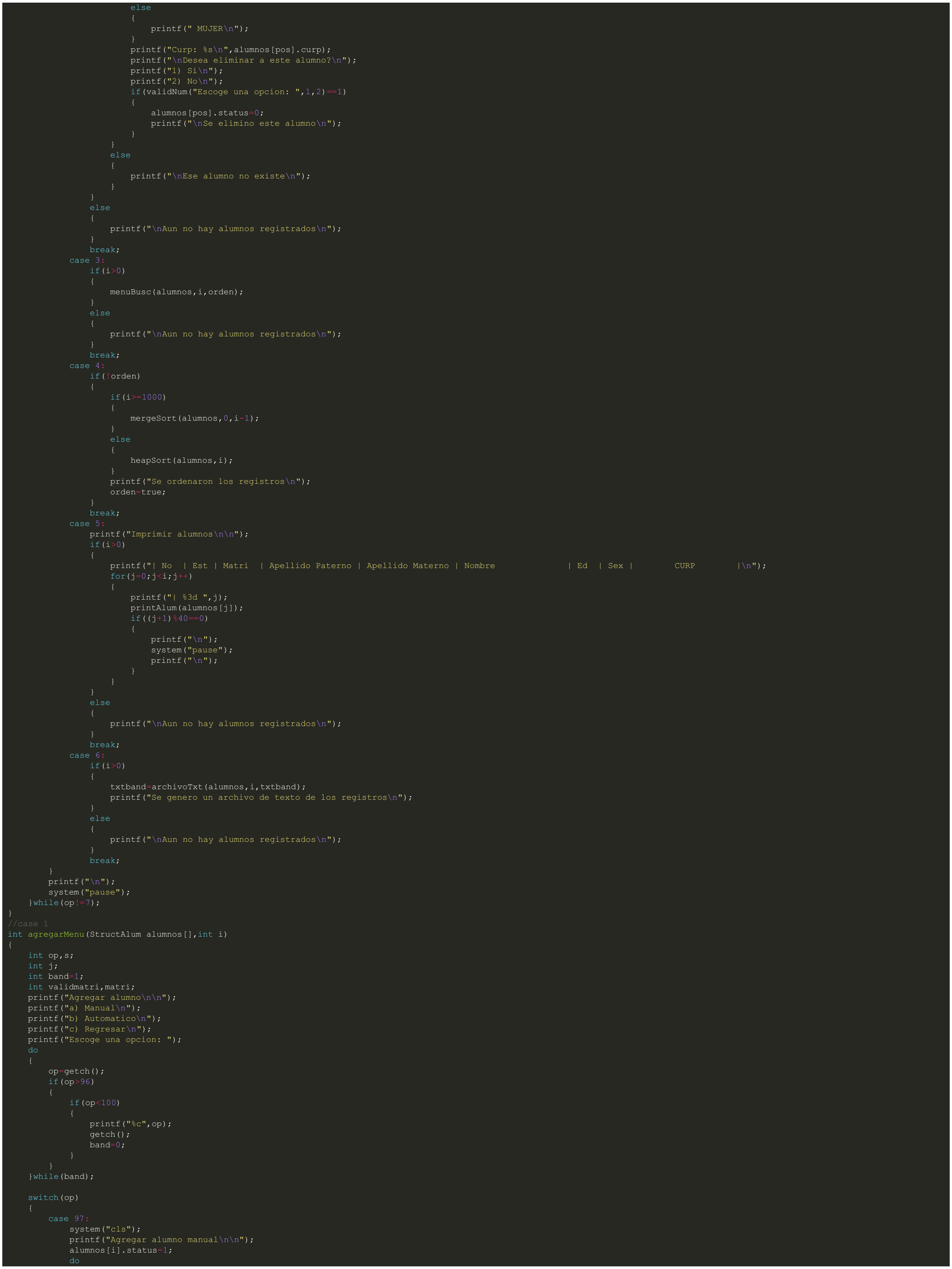
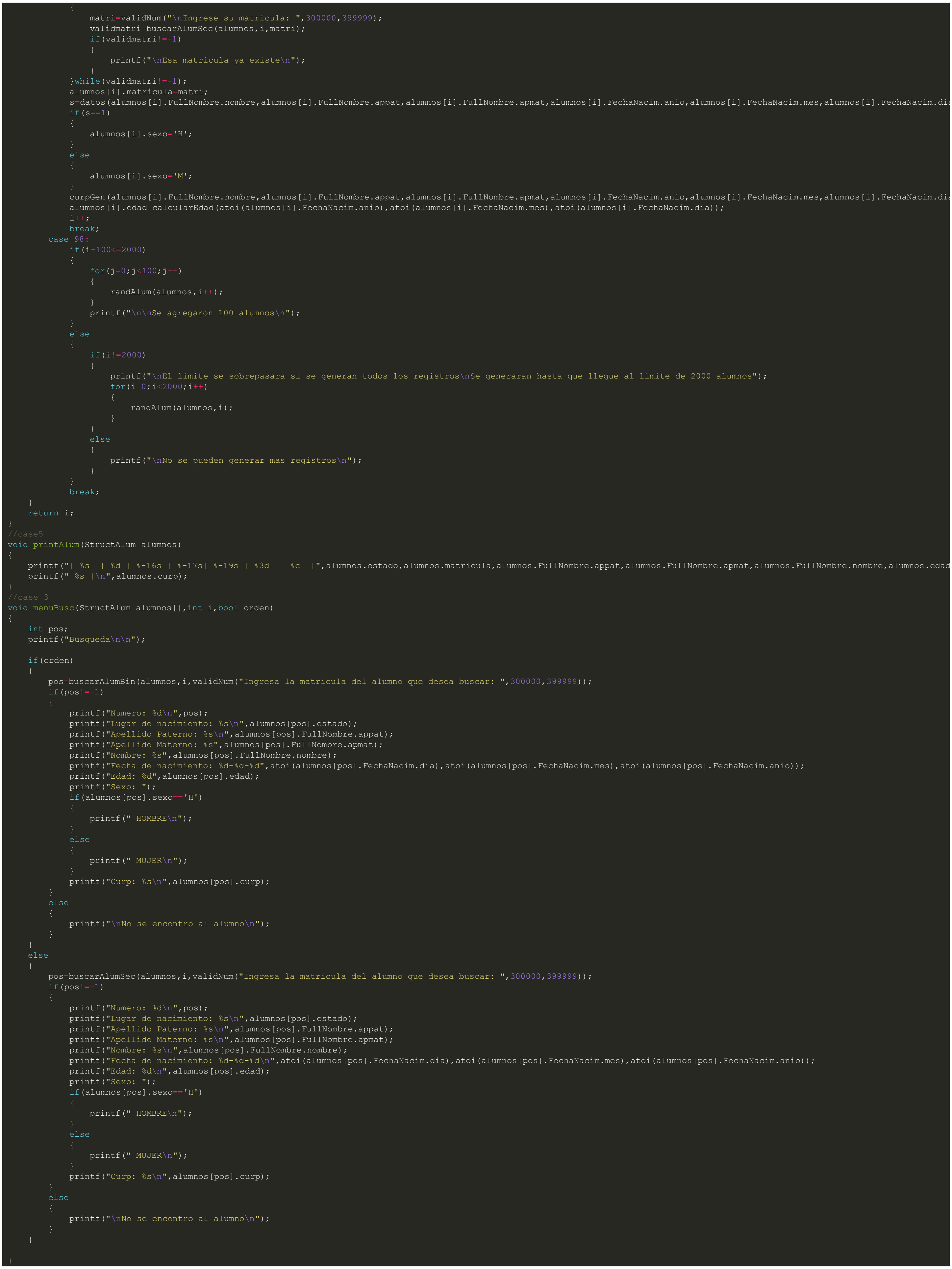
```c
/* MejÃ a MartÃ nez, RamÃ³n    372099                                                                                          */
/* 1 de noviembre de 2023                                                                                                    */
/* Programa de gestion de alumnos usando structs, implementacion de curp y distintos metodos de busqueda y ordenacion       */
/* MMR_ACT11_932                                                                                                            */

#include "CURPLIB.h"

#define STR 100
#define L 30
#define LN 50
#define STC 2000

typedef struct _name
{
    char appat[L];
    char apmat[L];
    char nombre[LN];
}StructName;

typedef struct _edad
{
    char anio[5];
    char mes[3];
    char dia[3];
}StructEdad;

typedef struct _alum
{
    int status;
    int matricula;
    StructName FullNombre;
    StructEdad FechaNacim;
    char estado[3];
    char sexo;
    int edad;
    char curp[18];
}StructAlum;

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

int agregarMenu(StructAlum alumnos[],int i);
void menuBusc(StructAlum alumnos[],int i,bool orden);
int buscarAlumSec(StructAlum alumnos[],int i,int num);
int buscarAlumBin(StructAlum alumnos[],int i,int num);
void printAlum(StructAlum alumnos);

int genMatri(StructAlum alumnos[],int i);
void genM_Name(StructAlum alumnos[],int i);
void genH_Name(StructAlum alumnos[],int i);
void genAp(StructAlum alumnos[],int i);
void randAlum(StructAlum alumnos[],int i);
void genEst(char estado[]);
int calcularEdad(int anio,int mes,int dia);
int archivoTxt(StructAlum alumnos[],int i,int band);
void mergeSort(StructAlum alumnos[],int izq,int derch);
void fusionMergeSort(StructAlum alumnos[],int izq,int med,int derch);
void heapSort(StructAlum alumnos[],int n);
void adjustHeap(StructAlum alumnos[],int n,int raiz);
void swap(StructAlum *x,StructAlum *y);

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

int msgs(void)
{
    printf("Menu \n\n");
    printf("1) Agregar\n");
    printf("2) Eliminar Registro\n");
    printf("3) Buscar\n");
    printf("4) Ordenar\n");
    printf("5) Imprimir\n");
    printf("6) Archivo Texto\n");
    printf("7) Salir\n");
    return validNum("Escoge una opcion: ",1,7);
}

void menu(void)
{
    StructAlum alumnos[STC];
    int op;
    int pos;
    int i=0,j;
    int txtband=1;
    bool orden=false;
    do
    {
        system("cls");
        op=msgs();
        system("cls");
        switch(op)
        {
            case 1:
                i=agregarMenu(alumnos,i);
                orden=false;
                break;
            case 2:
                printf("Eliminar alumno\n\n");
                if(i>0)
                {

                    pos=buscarAlumSec(alumnos,i,validNum("Ingresa la matricula del alumno que desea eliminar: ",300000,399999));
                    if(pos!=-1)
                    {
                        printf("Numero: %d\n",pos);
                        printf("Lugar de nacimiento: %s\n",alumnos[pos].estado);
                        printf("Apellido Paterno: %s\n",alumnos[pos].FullNombre.appat);
                        printf("Apellido Materno: %s",alumnos[pos].FullNombre.apmat);
                        printf("Nombre: %s",alumnos[pos].FullNombre.nombre);
                        printf("Fecha de nacimiento: %d-%d-%d",atoi(alumnos[pos].FechaNacim.dia),atoi(alumnos[pos].FechaNacim.mes),atoi(alumnos[pos].FechaNacim.anio));
                        printf("Edad: %d",alumnos[pos].edad);
                        printf("Sexo: ");
                        if(alumnos[pos].sexo=='H')
                        {
                            printf(" HOMBRE\n");
                        }
```

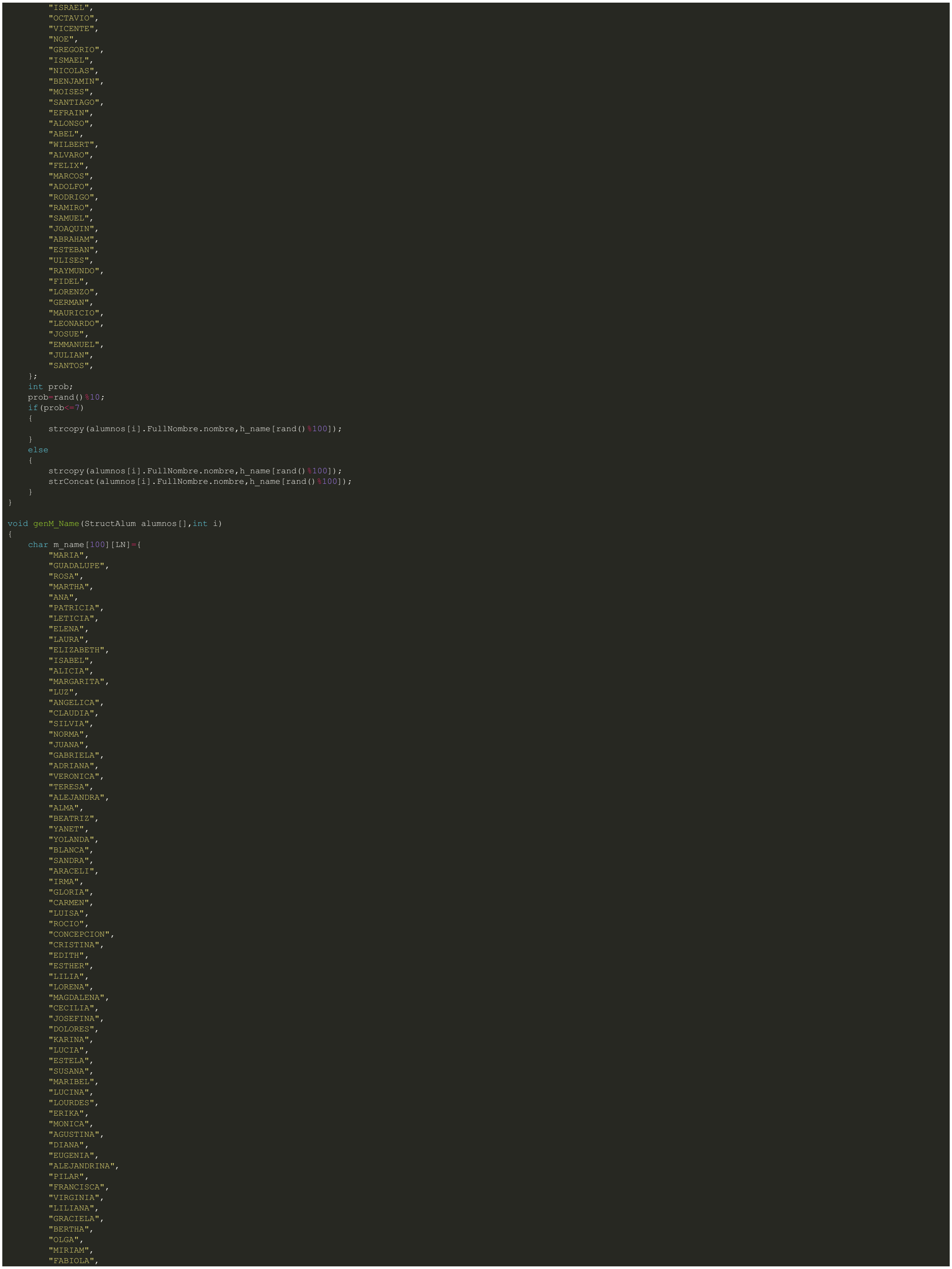


```
                        else
                        {
                            printf(" MUJER\n");
                        }
                        printf("Curp: %s\n",alumnos[pos].curp);
                        printf("\nDesea eliminar a este alumno?\n");
                        printf("1) Si\n");
                        printf("2) No\n");
                        if(validNum("Escoge una opcion: ",1,2)==1)
                        {
                            alumnos[pos].status=0;
                            printf("\nSe elimino este alumno\n");
                        }
                    }
                    else
                    {
                        printf("\nEse alumno no existe\n");
                    }
                }
                else
                {
                    printf("\nAun no hay alumnos registrados\n");
                }
                break;
            case 3:
                if(i>0)
                {
                    menuBusc(alumnos,i,orden);
                }
                else
                {
                    printf("\nAun no hay alumnos registrados\n");
                }
                break;
            case 4:
                if(!orden)
                {
                    if(i>=1000)
                    {
                        mergeSort(alumnos,0,i-1);
                    }
                    else
                    {
                        heapSort(alumnos,i);
                    }
                    printf("Se ordenaron los registros\n");
                    orden=true;
                }
                break;
            case 5:
                printf("Imprimir alumnos\n\n");
                if(i>0)
                {
                    printf("| No  | Est | Matri  | Apellido Paterno | Apellido Materno | Nombre              | Ed  | Sex |        CURP        |\n");
                    for(j=0;j<i;j++)
                    {
                        printf("| %3d ",j);
                        printAlum(alumnos[j]);
                        if((j+1)%40==0)
                        {
                            printf("\n");
                            system("pause");
                            printf("\n");
                        }
                    }
                }
                else
                {
                    printf("\nAun no hay alumnos registrados\n");
                }
                break;
            case 6:
                if(i>0)
                {
                    txtband=archivoTxt(alumnos,i,txtband);
                    printf("Se genero un archivo de texto de los registros\n");
                }
                else
                {
                    printf("\nAun no hay alumnos registrados\n");
                }
                break;
        }
        printf("\n");
        system("pause");
    }while(op!=7);
}
//case 1
int agregarMenu(StructAlum alumnos[],int i)
{
    int op,s;
    int j;
    int band=1;
    int validmatri,matri;
    printf("Agregar alumno\n\n");
    printf("a) Manual\n");
    printf("b) Automatico\n");
    printf("c) Regresar\n");
    printf("Escoge una opcion: ");
    do
    {
        op=getch();
        if(op>96)
        {
            if(op<100)
            {
                printf("%c",op);
                getch();
                band=0;
            }
        }
    }while(band);

    switch(op)
    {
        case 97:
            system("cls");
            printf("Agregar alumno manual\n\n");
            alumnos[i].status=1;
            do
```

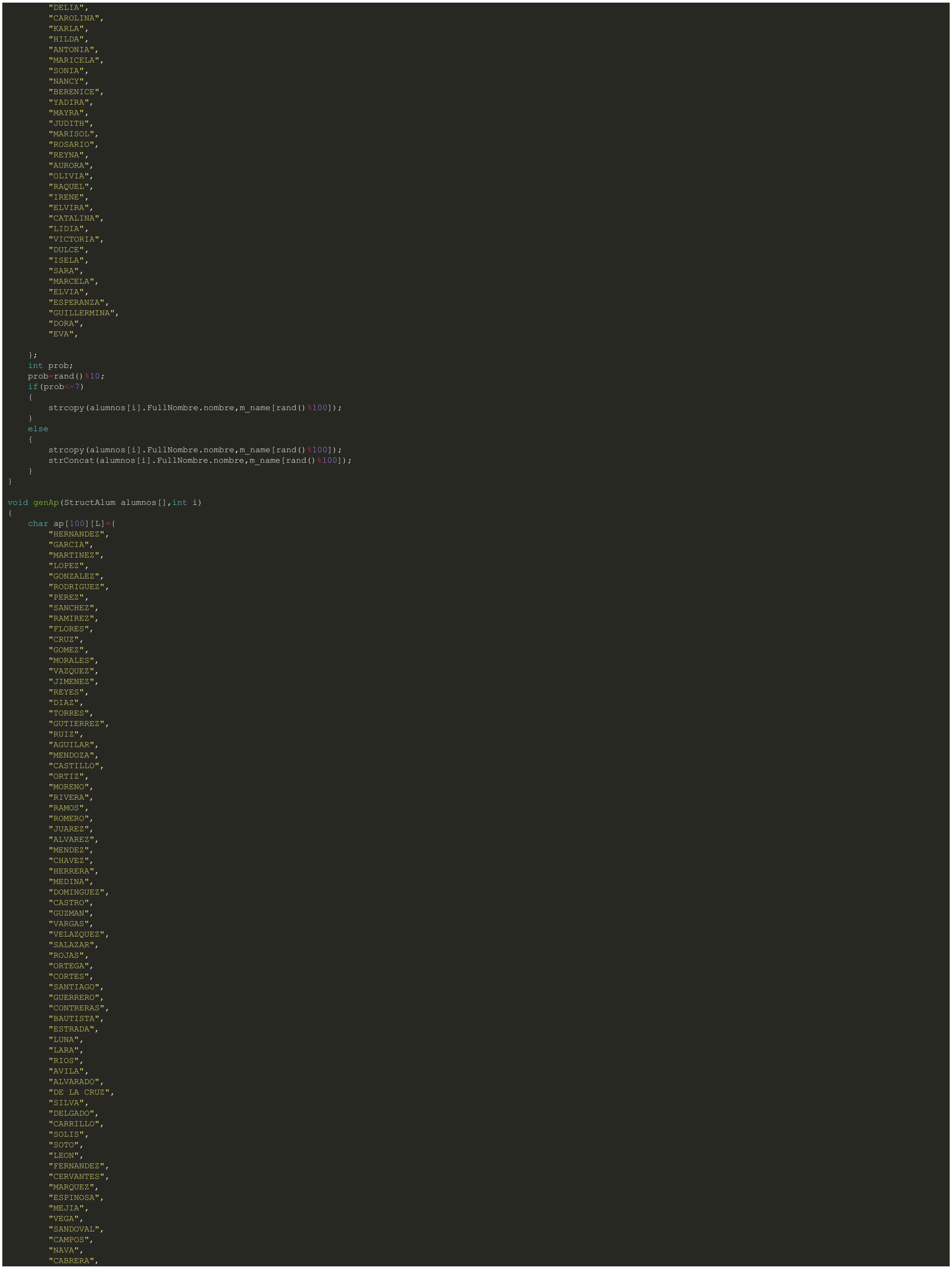


```
            {
                matri=validNum("\nIngrese su matricula: ",300000,399999);
                validmatri=buscarAlumSec(alumnos,i,matri);
                if(validmatri!=-1)
                {
                    printf("\nEsa matricula ya existe\n");
                }
            }while(validmatri!=-1);
            alumnos[i].matricula=matri;
            s=datos(alumnos[i].FullNombre.nombre,alumnos[i].FullNombre.appat,alumnos[i].FullNombre.apmat,alumnos[i].FechaNacim.anio,alumnos[i].FechaNacim.mes,alumnos[i].FechaNacim.dia
            if(s==1)
            {
                alumnos[i].sexo='H';
            }
            else
            {
                alumnos[i].sexo='M';
            }
            curpGen(alumnos[i].FullNombre.nombre,alumnos[i].FullNombre.appat,alumnos[i].FullNombre.apmat,alumnos[i].FechaNacim.anio,alumnos[i].FechaNacim.mes,alumnos[i].FechaNacim.dia
            alumnos[i].edad=calcularEdad(atoi(alumnos[i].FechaNacim.anio),atoi(alumnos[i].FechaNacim.mes),atoi(alumnos[i].FechaNacim.dia));
            i++;
            break;
        case 98:
            if(i+100<=2000)
            {
                for(j=0;j<100;j++)
                {
                    randAlum(alumnos,i++);
                }
                printf("\n\nSe agregaron 100 alumnos\n");
            }
            else
            {
                if(i!=2000)
                {
                    printf("\nEl limite se sobrepasara si se generan todos los registros\nSe generaran hasta que llegue al limite de 2000 alumnos");
                    for(i=0;i<2000;i++)
                    {
                        randAlum(alumnos,i);
                    }
                }
                else
                {
                    printf("\nNo se pueden generar mas registros\n");
                }
            }
            break;
    }
    return i;
}
//case5
void printAlum(StructAlum alumnos)
{
    printf("| %s  | %d | %-16s | %-17s| %-19s | %3d |  %c  |",alumnos.estado,alumnos.matricula,alumnos.FullNombre.appat,alumnos.FullNombre.apmat,alumnos.FullNombre.nombre,alumnos.edad
    printf(" %s |\n",alumnos.curp);
}
//case 3
void menuBusc(StructAlum alumnos[],int i,bool orden)
{
    int pos;
    printf("Busqueda\n\n");

    if(orden)
    {
        pos=buscarAlumBin(alumnos,i,validNum("Ingresa la matricula del alumno que desea buscar: ",300000,399999));
        if(pos!=-1)
        {
            printf("Numero: %d\n",pos);
            printf("Lugar de nacimiento: %s\n",alumnos[pos].estado);
            printf("Apellido Paterno: %s\n",alumnos[pos].FullNombre.appat);
            printf("Apellido Materno: %s",alumnos[pos].FullNombre.apmat);
            printf("Nombre: %s",alumnos[pos].FullNombre.nombre);
            printf("Fecha de nacimiento: %d-%d-%d",atoi(alumnos[pos].FechaNacim.dia),atoi(alumnos[pos].FechaNacim.mes),atoi(alumnos[pos].FechaNacim.anio));
            printf("Edad: %d",alumnos[pos].edad);
            printf("Sexo: ");
            if(alumnos[pos].sexo=='H')
            {
                printf(" HOMBRE\n");
            }
            else
            {
                printf(" MUJER\n");
            }
            printf("Curp: %s\n",alumnos[pos].curp);
        }
        else
        {
            printf("\nNo se encontro al alumno\n");
        }
    }
    else
    {
        pos=buscarAlumSec(alumnos,i,validNum("Ingresa la matricula del alumno que desea buscar: ",300000,399999));
        if(pos!=-1)
        {
            printf("Numero: %d\n",pos);
            printf("Lugar de nacimiento: %s\n",alumnos[pos].estado);
            printf("Apellido Paterno: %s\n",alumnos[pos].FullNombre.appat);
            printf("Apellido Materno: %s\n",alumnos[pos].FullNombre.apmat);
            printf("Nombre: %s\n",alumnos[pos].FullNombre.nombre);
            printf("Fecha de nacimiento: %d-%d-%d\n",atoi(alumnos[pos].FechaNacim.dia),atoi(alumnos[pos].FechaNacim.mes),atoi(alumnos[pos].FechaNacim.anio));
            printf("Edad: %d\n",alumnos[pos].edad);
            printf("Sexo: ");
            if(alumnos[pos].sexo=='H')
            {
                printf(" HOMBRE\n");
            }
            else
            {
                printf(" MUJER\n");
            }
            printf("Curp: %s\n",alumnos[pos].curp);
        }
        else
        {
            printf("\nNo se encontro al alumno\n");
        }
    }

}
```

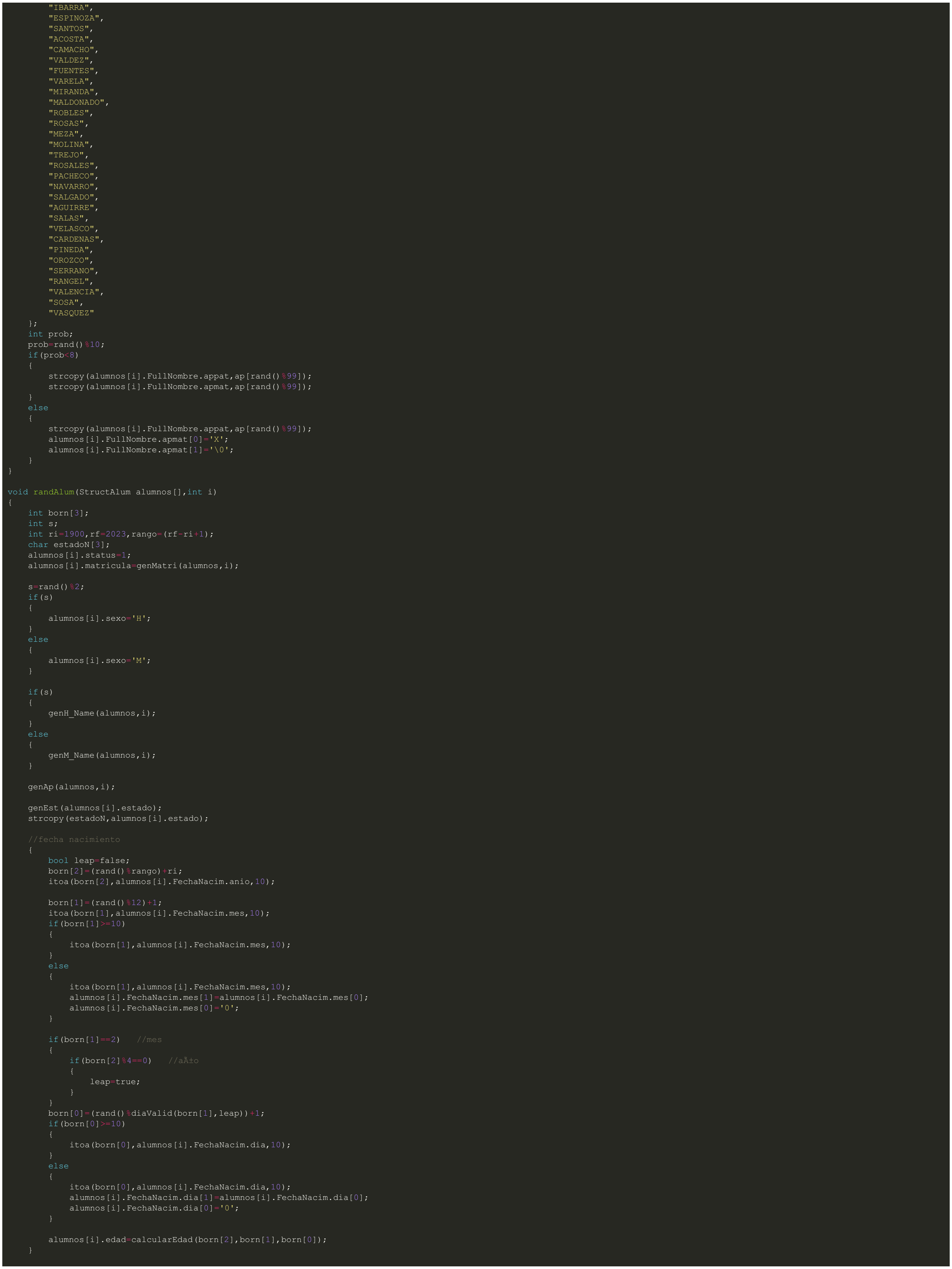
```
//case 3-1
int buscarAlumSec(StructAlum alumnos[],int i,int num)
{
    int j;
    for(j=0;j<i;j++)
    {
        if(num==alumnos[j].matricula)
        {
            return j;
        }
    }
    return -1;
}
//case 3-2
int buscarAlumBin(StructAlum alumnos[],int i,int num)
{
    int izq,drcha;
    int med;
    izq=0;
    drcha=i-1;
    while(izq<=drcha)
    {
        med=izq+(drcha-izq)/2;
        if(alumnos[med].matricula==num)
        {
            return med;
        }

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

//Generar Alumno
int genMatri(StructAlum alumnos[],int i)
{
    int num;
    do
    {
        num=(rand()%100000)+300000;
    }while(buscarAlumSec(alumnos,i,num)!=-1);
    return num;
}

void genH_Name(StructAlum alumnos[],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",
```



```c
        "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(alumnos[i].FullNombre.nombre,h_name[rand()%100]);
    }
    else
    {
        strcopy(alumnos[i].FullNombre.nombre,h_name[rand()%100]);
        strConcat(alumnos[i].FullNombre.nombre,h_name[rand()%100]);
    }
}

void genM_Name(StructAlum alumnos[],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(alumnos[i].FullNombre.nombre,m_name[rand()%100]);
    }
    else
    {
        strcopy(alumnos[i].FullNombre.nombre,m_name[rand()%100]);
        strConcat(alumnos[i].FullNombre.nombre,m_name[rand()%100]);
    }
}

void genAp(StructAlum alumnos[],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",
```

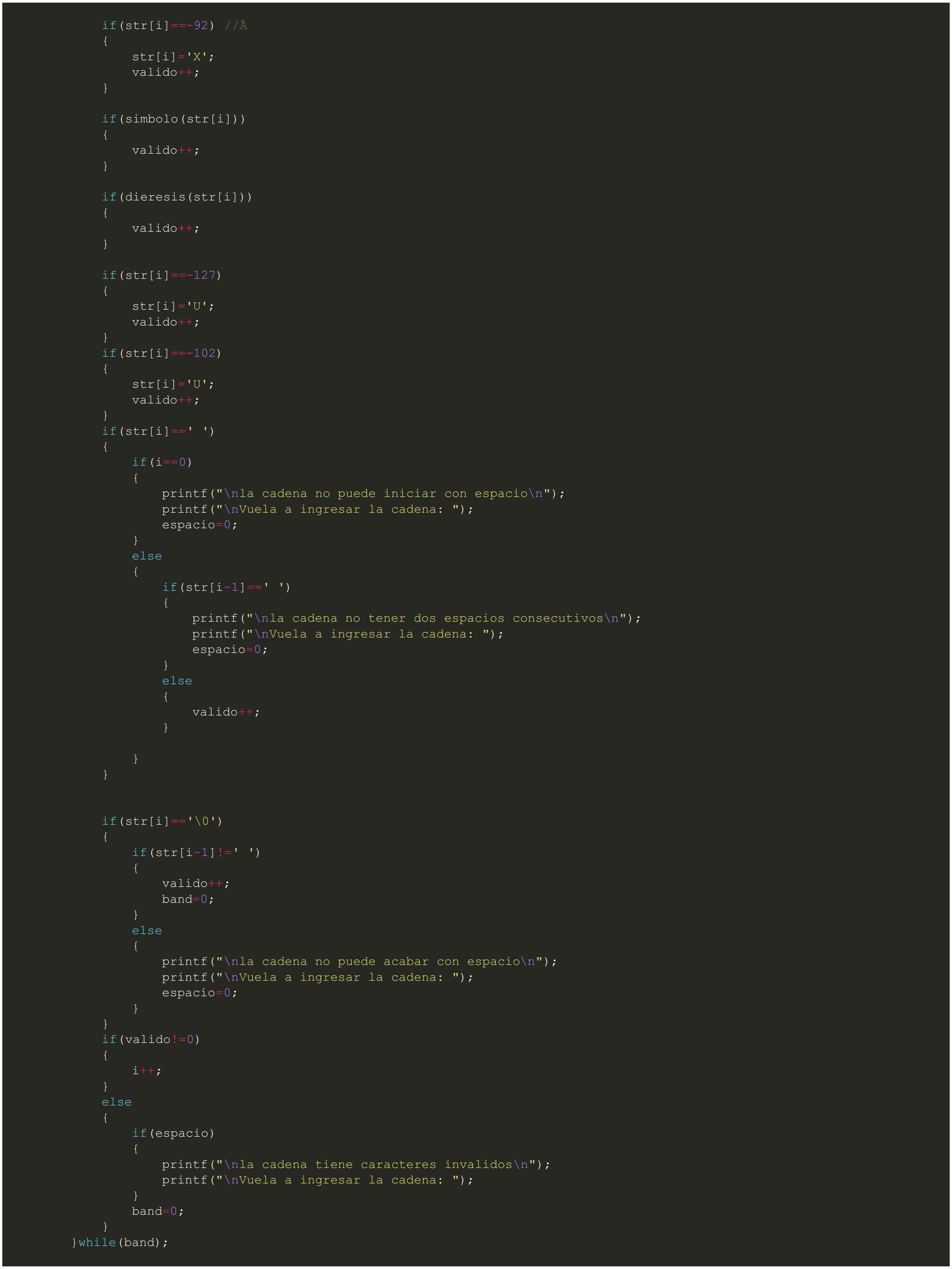


```
        "IBARRA",
        "ESPINOZA",
        "SANTOS",
        "ACOSTA",
        "CAMACHO",
        "VALDEZ",
        "FUENTES",
        "VARELA",
        "MIRANDA",
        "MALDONADO",
        "ROBLES",
        "ROSAS",
        "MEZA",
        "MOLINA",
        "TREJO",
        "ROSALES",
        "PACHECO",
        "NAVARRO",
        "SALGADO",
        "AGUIRRE",
        "SALAS",
        "VELASCO",
        "CARDENAS",
        "PINEDA",
        "OROZCO",
        "SERRANO",
        "RANGEL",
        "VALENCIA",
        "SOSA",
        "VASQUEZ"
    };
    int prob;
    prob=rand()%10;
    if(prob<8)
    {
        strcopy(alumnos[i].FullNombre.appat,ap[rand()%99]);
        strcopy(alumnos[i].FullNombre.apmat,ap[rand()%99]);
    }
    else
    {
        strcopy(alumnos[i].FullNombre.appat,ap[rand()%99]);
        alumnos[i].FullNombre.apmat[0]='X';
        alumnos[i].FullNombre.apmat[1]='\0';
    }
}

void randAlum(StructAlum alumnos[],int i)
{
    int born[3];
    int s;
    int ri=1900,rf=2023,rango=(rf-ri+1);
    char estadoN[3];
    alumnos[i].status=1;
    alumnos[i].matricula=genMatri(alumnos,i);

    s=rand()%2;
    if(s)
    {
        alumnos[i].sexo='H';
    }
    else
    {
        alumnos[i].sexo='M';
    }

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

    genAp(alumnos,i);

    genEst(alumnos[i].estado);
    strcopy(estadoN,alumnos[i].estado);

    //fecha nacimiento
    {
        bool leap=false;
        born[2]=(rand()%rango)+ri;
        itoa(born[2],alumnos[i].FechaNacim.anio,10);

        born[1]=(rand()%12)+1;
        itoa(born[1],alumnos[i].FechaNacim.mes,10);
        if(born[1]>=10)
        {
            itoa(born[1],alumnos[i].FechaNacim.mes,10);
        }
        else
        {
            itoa(born[1],alumnos[i].FechaNacim.mes,10);
            alumnos[i].FechaNacim.mes[1]=alumnos[i].FechaNacim.mes[0];
            alumnos[i].FechaNacim.mes[0]='0';
        }

        if(born[1]==2)   //mes
        {
            if(born[2]%4==0)   //aÃ±o
            {
                leap=true;
            }
        }
        born[0]=(rand()%diaValid(born[1],leap))+1;
        if(born[0]>=10)
        {
            itoa(born[0],alumnos[i].FechaNacim.dia,10);
        }
        else
        {
            itoa(born[0],alumnos[i].FechaNacim.dia,10);
            alumnos[i].FechaNacim.dia[1]=alumnos[i].FechaNacim.dia[0];
            alumnos[i].FechaNacim.dia[0]='0';
        }

        alumnos[i].edad=calcularEdad(born[2],born[1],born[0]);
    }
```

```c
    curpGen(alumnos[i].FullNombre.nombre,alumnos[i].FullNombre.appat,alumnos[i].FullNombre.apmat,alumnos[i].FechaNacim.anio,alumnos[i].FechaNacim.mes,alumnos[i].FechaNacim.dia,alumnos

    alumnos[i].curp[11]=estadoN[0];
    alumnos[i].curp[12]=estadoN[1];

}

void genEst(char estado[])
{
    int E;
    char estList[33][3]={
    "AS","BC","BS","CC","CS","CH","CM","CL","MX","DG",
    "GT","GR","HG","JC","EM","MN","MS","NT","NL","OC",
    "PL","QT","QR","SP","SL","SR","TC","TS","TL","VZ",
    "YN","ZS","NE"
    };


    E=rand()%33;

    estado[0]=estList[E][0];
    estado[1]=estList[E][1];
    estado[2]='\0';
}

int calcularEdad(int anio,int mes,int dia)
{
    int edad;
    time_t t;
    t=time(NULL);
    struct tm *fecha;
    fecha=localtime(&t);

    edad=(fecha->tm_year+1900)-anio;
    if(mes>fecha->tm_mon+1)
    {
        edad--;
    }
    else
    {
        if(mes==fecha->tm_mon+1)
        {
            if(dia>fecha->tm_mday)
            {
                edad--;
            }
        }
    }
    return edad;
}

int archivoTxt(StructAlum alumnos[],int i,int band)
{
    int j;
    FILE *pa;
    if(band)
    {
        pa=fopen("registros.txt","a");
        char str[119]={"| No  | Est | Matri  | Apellido Paterno | Apellido Materno | Nombre              | Ed  | Sex |        CURP        |\n"};
        fprintf(pa,"%s",str);

        for(j=0;j<i;j++)
        {
            if(alumnos[j].status)
            {
                fprintf(pa,"| %4d ",j);
                fprintf(pa,"| %s  | %d | %-16s | %-17s| %-19s | %3d |  %c  |",alumnos[j].estado,alumnos[j].matricula,alumnos[j].FullNombre.appat,alumnos[j].FullNombre.apmat,alumnos[j]
                fprintf(pa," %s |\n",alumnos[j].curp);
            }
        }
    }
    else
    {
        pa=fopen("registros.txt","w");
        char str[119]={"| No  | Est | Matri  | Apellido Paterno | Apellido Materno | Nombre              | Ed  | Sex |        CURP        |\n"};
        fprintf(pa,"%s",str);

        for(j=0;j<i;j++)
        {
            if(alumnos[j].status)
            {
                fprintf(pa,"| %4d ",j);
                fprintf(pa,"| %s  | %d | %-16s | %-17s| %-19s | %3d |  %c  |",alumnos[j].estado,alumnos[j].matricula,alumnos[j].FullNombre.appat,alumnos[j].FullNombre.apmat,alumnos[j]
                fprintf(pa," %s |\n",alumnos[j].curp);
            }
        }
    }

    fclose(pa);

    return 0;
}

//MergeSort
void mergeSort(StructAlum alumnos[],int izq,int derch)
{
    int med;

    if(izq<derch)
    {
        med=izq+(derch-izq)/2;
        mergeSort(alumnos,izq,med);
        mergeSort(alumnos,med+1,derch);
        fusionMergeSort(alumnos,izq,med,derch);
    }
}

void fusionMergeSort(StructAlum alumnos[],int izq,int medio,int derch)
{
    int i,j,k;
    int n1=medio-izq+1;
    int n2=derch-medio;
    StructAlum Lf[n1], R[n2];

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

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

//HeapSort
void swap(StructAlum *x,StructAlum *y)
{
    StructAlum temp = *x;
    *x = *y;
    *y = temp;
}

void adjustHeap(StructAlum alumnos[],int n,int raiz)
{
    int may=raiz;
    int izq =2 *raiz+1;
    int derch =2*raiz+2;

    if(izq<n)
    {
        if(alumnos[izq].matricula>alumnos[may].matricula)
        {
            may=izq;
        }
    }

    if(derch<n)
    {
        if(alumnos[derch].matricula>alumnos[may].matricula)
        {
            may=derch;
        }

    }

    if(may!=raiz)
    {
        swap(&alumnos[raiz],&alumnos[may]);
        adjustHeap(alumnos,n,may);
    }
}

void heapSort(StructAlum alumnos[], int n)
{
    int i;
    for (i=n/2-1;i>=0;i--)
    {
        adjustHeap(alumnos,n,i);
    }

    for (i=n-1;i>0;i--)
    {
        swap(&alumnos[0],&alumnos[i]);
        adjustHeap(alumnos,i,0);
    }
}
```

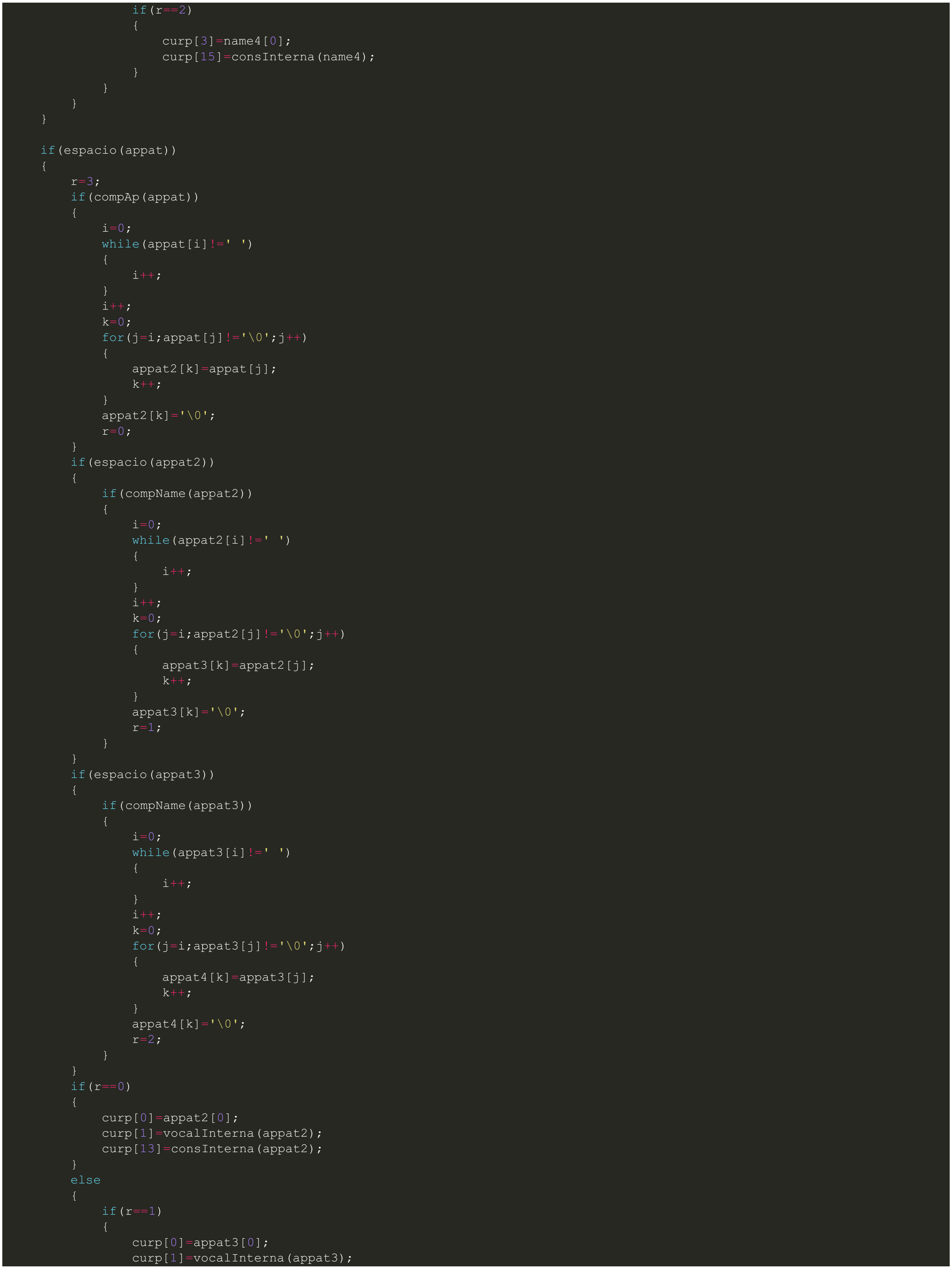
```c
#include "C:\Users\ramon\OneDrive\Escritorio\MMR_ACT1_932\MMRLIB.h"

#define STR 100
#define L 30
#define LN 50
#define STC 2000

void validStrCurp(char str[]);
int datos(char name[],char appat[],char apmat[],char anio[],char mes[],char dia[],char estado[]);
void fecha(char anio[],char mes[],char dia[]);
void Est(char estado[]);
int diaValid(int mes,bool leap);
char vocalInterna(char appat[]);
char consInterna(char str[]);
int dieresis(char c);
void validCurp(char curp[],char name[],char appat[],char apmat[]);
void validName(char curp[],char name[],char appat[],char apmat[]);
int compName(char name[]);
int compAp(char ap[]);
int espacio(char str[]);
void curpGen(char namem[],char appatm[],char apmatm[],char anio[],char mes[],char dia[],char estado[],char curp[],char sex);
char curp16(char anio[]);
int simbolo(char c1);
void validSimbolo(char curp[]);
int validPalabraInconven(char curp[]);

int diaValid(int mes,bool leap)
{
    switch(mes)
    {
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:
            return 31;
            break;
        case 4:
        case 6:
        case 9:
        case 11:
            return 30;
            break;
        case 2:
            if(leap==true)
            {
                return 29;
            }
            else
            {
                return 28;
            }
            break;
    }
}

int datos(char name[],char appat[],char apmat[],char anio[],char mes[],char dia[],char estado[])
{
    int sex;
    printf("\nIngresa tu nombre: ");
    validStrCurp(name);
    strM(name);


    printf("\nIngresa tu apellido paterno: ");
    validStrCurp(appat);
    strM(appat);
    if(strlen(appat)==0)
    {
        appat[0]='X';
        appat[1]='\0';
    }


    printf("\nIngresa tu apellido materno: ");
    validStrCurp(apmat);
    strM(apmat);
    if(strlen(apmat)==0)
    {
        apmat[0]='X';
        apmat[1]='\0';
    }

    printf("\n\nFecha de nacimiento\n\n");
    fecha(anio,mes,dia);

    printf("\n");
    Est(estado);

    sex=validNum("\n\nIngresa tu sexo\n1) Masculino\n2) Femenino\nElige una opcion: ",1,2);
    return sex;
}

void curpGen(char namem[],char appatm[],char apmatm[],char anio[],char mes[],char dia[],char estado[],char curp[],char sex)
{
    //datos
    char rnd[2];
    char name[LN],appat[L],apmat[L];
    strcopy(name,namem);
    strcopy(appat,appatm);
    strcopy(apmat,apmatm);

    curp[0]=appat[0];
    curp[1]=vocalInterna(appat);
    curp[2]=apmat[0];
    curp[3]=name[0];
    curp[4]=anio[2];
    curp[5]=anio[3];
    curp[6]=mes[0];
    curp[7]=mes[1];
    curp[8]=dia[0];
    curp[9]=dia[1];
    curp[10]=sex;
    curp[11]=estado[0];
    curp[12]=estado[1];
    curp[13]=consInterna(appat);
    curp[14]=consInterna(apmat);
    curp[15]=consInterna(name);
    curp[16]=curp16(anio);
    itoa(rand()%10,rnd,10);
    curp[17]=rnd[0];
    curp[18]='\0';

    validCurp(curp,name,appat,apmat);
}

void fecha(char anio[],char mes[],char dia[])
{
    int born[3];
    bool leap=false;
    born[2]=validNum("\nIngresa tu aÃ±o de nacimiento: ",1900,2023);
    itoa(born[2],anio,10);
    anio[4]='\0';

    born[1]=validNum("\nIngresa tu mes de nacimiento: ",1,12);
    itoa(born[1],mes,10);
    if(born[1]>=10)
    {
        itoa(born[1],mes,10);
    }
    else
    {
        itoa(born[1],mes,10);
        mes[1]=mes[0];
        mes[0]='0';
    }
    mes[2]='\0';

    if(born[1]==2)    //mes
    {
        if(born[2]%4==0)    //aÃ±o
        {
            leap=true;
        }
    }
    born[0]=validNum("\nIngresa tu dia de nacimiento: ",1,diaValid(born[1],leap));
    if(born[0]>=10)
    {
        itoa(born[0],dia,10);
    }
    else
    {
        itoa(born[0],dia,10);
        dia[1]=dia[0];
        dia[0]='0';
    }
    dia[2]='\0';
}

char vocalInterna(char appat[])
{
    int i;
    for(i=1;appat[i]!='\0';i++)
    {
        if(simbolo(appat[i]))
        {
            return appat[i];
        }
        if(appat[i]=='A')
        {
            return appat[i];
        }
        if(appat[i]=='E')
        {
            return appat[i];
        }
        if(appat[i]=='I')
        {
            return appat[i];
        }
        if(appat[i]=='O')
        {
            return appat[i];
        }
        if(appat[i]=='U')
        {
            return appat[i];
        }

        if(dieresis(appat[i]))
        {
            return 'X';
        }
    }
    return 'X';
}

char consInterna(char str[])
{
    int i;
    for(i=1;str[i]!='\0';i++)
    {
        if(str[i]!='A')
        {
            if(str[i]!='E')
            {
                if(str[i]!='I')
                {
                    if(str[i]!='O')
                    {
                        if(str[i]!='U')
                        {
                            if(!dieresis(str[i]))
                            {
                                return str[i];
                            }
                        }
                    }
                }
            }
        }
    }
    return 'X';
}

void Est(char estado[])
{
    int E;
    char estList[33][3]={
    "AS","BC","BS","CC","CS","CH","CM","CL","MX","DG",
    "GT","GR","HG","JC","EM","MN","MS","NT","NL","OC",
```

```c
    "PL","QT","QR","SP","SL","SR","TC","TS","TL","VZ",
    "YN","ZS","NE"
    };

    system("cls");
    printf("Estado\n\n");
    printf("1) Aguascalientes\n");
    printf("2) Baja California\n");
    printf("3) Baja California Sur\n");
    printf("4) Campeche\n");
    printf("5) Chiapas\n");
    printf("6) Chihuahua\n");
    printf("7) Ciudad de MÃ©xico\n");
    printf("8) Coahuila\n");
    printf("9) Colima\n");
    printf("10) Durango\n");
    printf("11) Guanajuato\n");
    printf("12) Guerrero\n");
    printf("13) Hidalgo\n");
    printf("14) Jalisco\n");
    printf("15) Estado de MÃ©xico\n");
    printf("16) MichoacÃ¡n\n");
    printf("17) Morelos\n");
    printf("18) Nayarit\n");
    printf("19) Nuevo LeÃ³n\n");
    printf("20) Oaxaca\n");
    printf("21) Puebla\n");
    printf("22) QuerÃ©taro\n");
    printf("23) Quintana Roo\n");
    printf("24) San Luis PotosÃ \n");
    printf("25) Sinaloa\n");
    printf("26) Sonora\n");
    printf("27) Tabasco\n");
    printf("28) Tamaulipas\n");
    printf("29) Tlaxcala\n");
    printf("30) Veracruz\n");
    printf("31) YucatÃ¡n\n");
    printf("32) Zacatecas\n");
    printf("33) Extranjero\n");
    E=validNum("Escoge tu estado: ",1,33);

    estado[0]=estList[E-1][0];
    estado[1]=estList[E-1][1];
}

void validStrCurp(char str[])
{
    int i,valido,band,espacio;
    do
    {
        fflush(stdin);
        gets(str);
        band=1;
        espacio=1;
        i=0;
        do
        {
            valido=0;
            //fflush(stdin);
            //c=getch();
            if(str[i]>='A')
            {
                if(str[i]<='Z')
                {
                    valido++;
                }
            }

            if(str[i]>='a')
            {
                if(str[i]<='z')
                {
                    valido++;
                }
            }

            if(str[i]==-91) //Ã±
            {
                str[i]='X';
                valido++;
            }
```

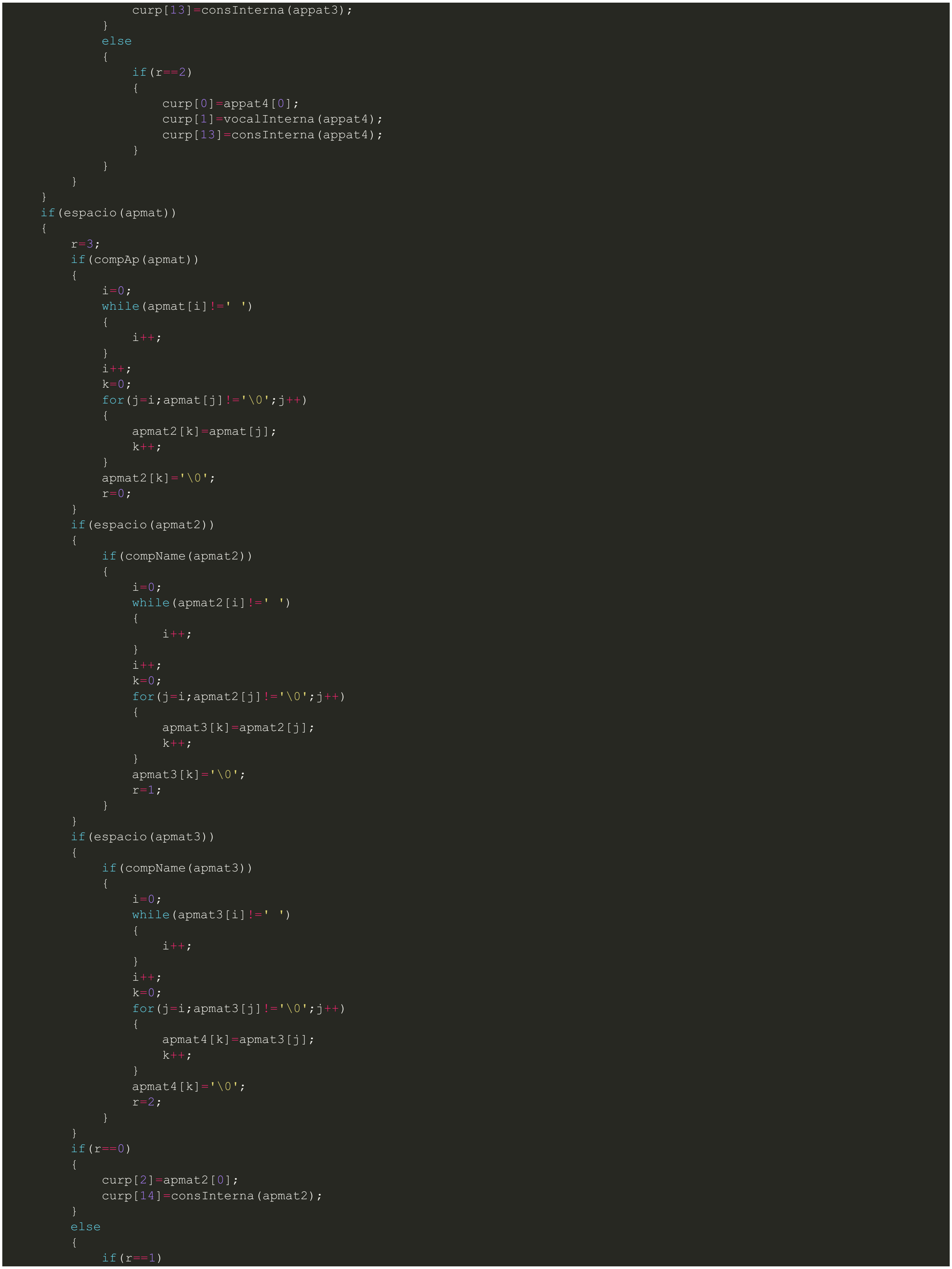


```
        if(str[i]==-92) //Ã
        {
            str[i]='X';
            valido++;
        }

        if(simbolo(str[i]))
        {
            valido++;
        }

        if(dieresis(str[i]))
        {
            valido++;
        }

        if(str[i]==-127)
        {
            str[i]='U';
            valido++;
        }
        if(str[i]==-102)
        {
            str[i]='U';
            valido++;
        }
        if(str[i]==' ')
        {
            if(i==0)
            {
                printf("\nla cadena no puede iniciar con espacio\n");
                printf("\nVuela a ingresar la cadena: ");
                espacio=0;
            }
            else
            {
                if(str[i-1]==' ')
                {
                    printf("\nla cadena no tener dos espacios consecutivos\n");
                    printf("\nVuela a ingresar la cadena: ");
                    espacio=0;
                }
                else
                {
                    valido++;
                }

            }
        }


        if(str[i]=='\0')
        {
            if(str[i-1]!=' ')
            {
                valido++;
                band=0;
            }
            else
            {
                printf("\nla cadena no puede acabar con espacio\n");
                printf("\nVuela a ingresar la cadena: ");
                espacio=0;
            }
        }
        if(valido!=0)
        {
            i++;
        }
        else
        {
            if(espacio)
            {
                printf("\nla cadena tiene caracteres invalidos\n");
                printf("\nVuela a ingresar la cadena: ");
            }
            band=0;
        }
    }while(band);
```

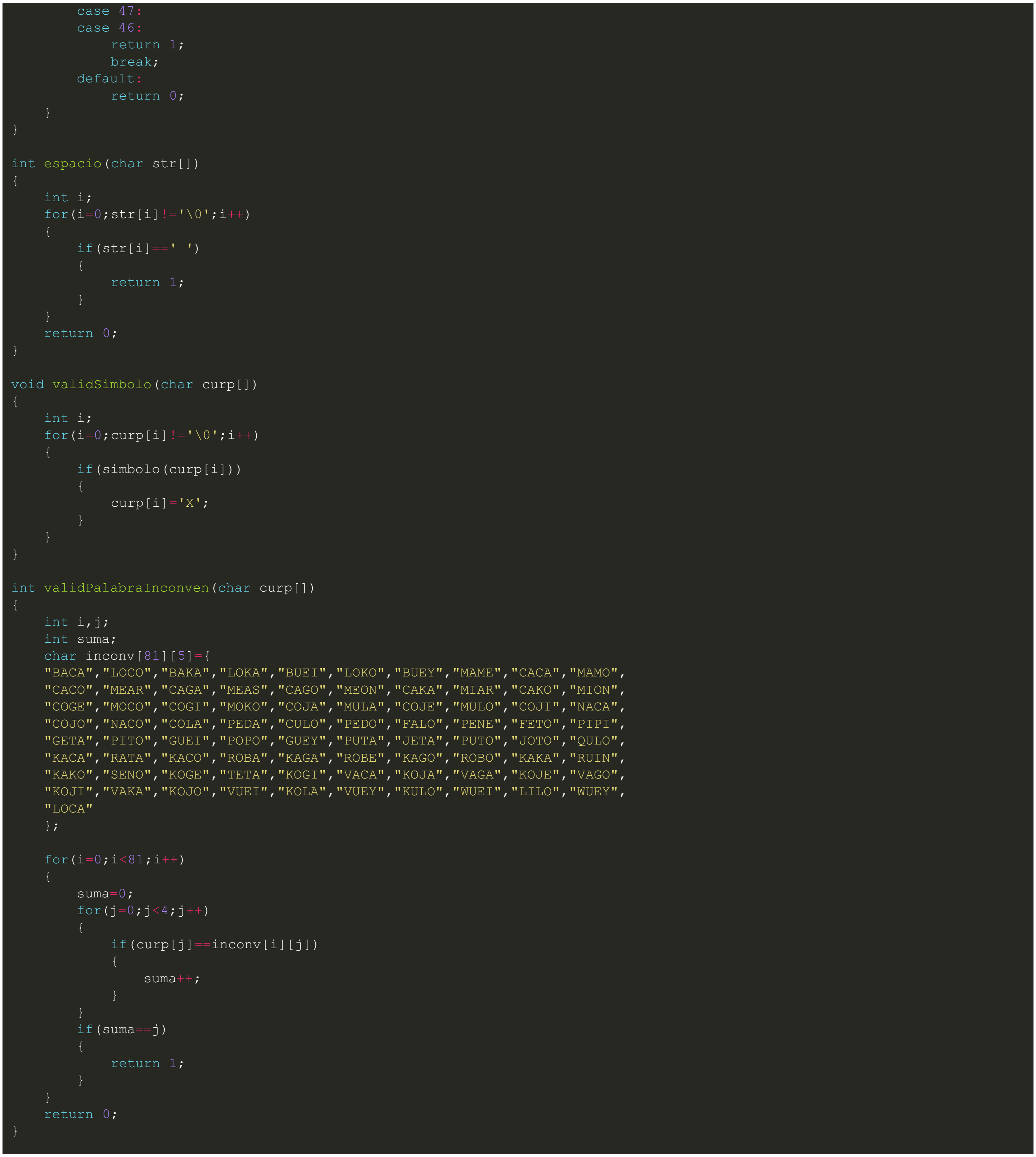
```c
    }while(valido==0);
}

char curp16(char anio[])
{
    if(anio[0]=='1')
    {
        return anio[2];
    }
    else
    {
        switch(anio[2])
        {
            case '0':
                return 'A';
                break;
            case '1':
                return 'B';
                break;
            case '2':
                return 'C';
                break;
        }
    }
}

int dieresis(char c)
{
    switch(c)
    {
        case -114:  //a
        case -45:   //e
        case -40:   //i
        case -103:  //o
        case -124:  //a
        case -119:  //e
        case -117:  //i
        case -108:  //o
            return 1;
            break;
        default:
            return 0;
    }
}

void validCurp(char curp[],char name[],char appat[],char apmat[])
{
    validName(curp,name,appat,apmat);

    //dieresis
    if(dieresis(curp[0]))
    {
        curp[0]='X';
    }
    if(dieresis(curp[2]))
    {
        curp[2]='X';
    }
    if(dieresis(curp[3]))
    {
        curp[3]='X';
    }

    validSimbolo(curp);

    if(validPalabraInconven(curp))
    {
        curp[1]='X';
    }

}

void validName(char curp[],char name[],char appat[],char apmat[])
{
    int i,j,k;
    int r;
    char name2[LN];
    char name3[LN];
    char name4[LN];
    char appat2[L];
    char appat3[L];
    char appat4[L];
    char apmat2[L];
    char apmat3[L];
    char apmat4[L];

    if(espacio(name))
    {
        r=3;
        if(compName(name))
        {
            i=0;
            while(name[i]!=' ')
            {
                i++;
            }
            i++;
            k=0;
            for(j=i;name[j]!='\0';j++)
            {
                name2[k]=name[j];
                k++;
            }
            name2[k]='\0';
            r=0;
        }
        if(espacio(name2))
        {
            if(compName(name2))
            {
                i=0;
                while(name2[i]!=' ')
                {
                    i++;
                }
                i++;
                k=0;
                for(j=i;name2[j]!='\0';j++)
                {
                    name3[k]=name2[j];
                    k++;
                }
                name3[k]='\0';
                r=1;
            }
        }
        if(espacio(name3))
        {
            if(compName(name3))
            {
                i=0;
                while(name3[i]!=' ')
                {
                    i++;
                }
                i++;
                k=0;
                for(j=i;name3[j]!='\0';j++)
                {
                    name4[k]=name3[j];
                    k++;
                }
                name4[k]='\0';
                r=2;
            }
        }

        if(r==0)
        {
            curp[3]=name2[0];
            curp[15]=consInterna(name2);

        }
        else
        {
            if(r==1)
            {
                curp[3]=name3[0];
                curp[15]=consInterna(name3);
            }
            else
            {
```



```
                if(r==2)
                {
                    curp[3]=name4[0];
                    curp[15]=consInterna(name4);
                }
            }
        }
    }

    if(espacio(appat))
    {
        r=3;
        if(compAp(appat))
        {
            i=0;
            while(appat[i]!=' ')
            {
                i++;
            }
            i++;
            k=0;
            for(j=i;appat[j]!='\0';j++)
            {
                appat2[k]=appat[j];
                k++;
            }
            appat2[k]='\0';
            r=0;
        }
        if(espacio(appat2))
        {
            if(compName(appat2))
            {
                i=0;
                while(appat2[i]!=' ')
                {
                    i++;
                }
                i++;
                k=0;
                for(j=i;appat2[j]!='\0';j++)
                {
                    appat3[k]=appat2[j];
                    k++;
                }
                appat3[k]='\0';
                r=1;
            }
        }
        if(espacio(appat3))
        {
            if(compName(appat3))
            {
                i=0;
                while(appat3[i]!=' ')
                {
                    i++;
                }
                i++;
                k=0;
                for(j=i;appat3[j]!='\0';j++)
                {
                    appat4[k]=appat3[j];
                    k++;
                }
                appat4[k]='\0';
                r=2;
            }
        }
        if(r==0)
        {
            curp[0]=appat2[0];
            curp[1]=vocalInterna(appat2);
            curp[13]=consInterna(appat2);
        }
        else
        {
            if(r==1)
            {
                curp[0]=appat3[0];
                curp[1]=vocalInterna(appat3);
```



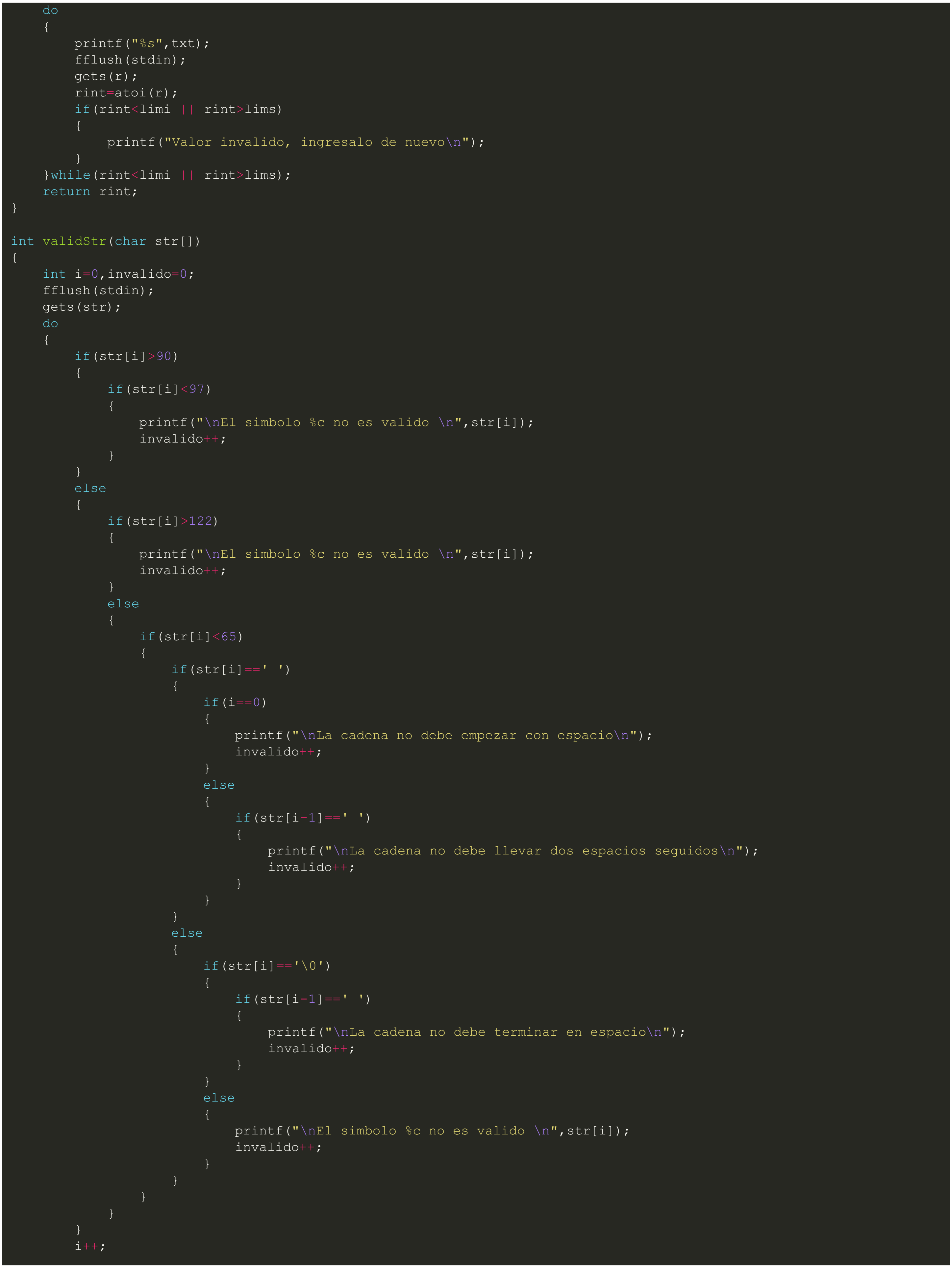
```
                curp[13]=consInterna(appat3);
            }
            else
            {
                if(r==2)
                {
                    curp[0]=appat4[0];
                    curp[1]=vocalInterna(appat4);
                    curp[13]=consInterna(appat4);
                }
            }
        }
    }
    if(espacio(apmat))
    {
        r=3;
        if(compAp(apmat))
        {
            i=0;
            while(apmat[i]!=' ')
            {
                i++;
            }
            i++;
            k=0;
            for(j=i;apmat[j]!='\0';j++)
            {
                apmat2[k]=apmat[j];
                k++;
            }
            apmat2[k]='\0';
            r=0;
        }
        if(espacio(apmat2))
        {
            if(compName(apmat2))
            {
                i=0;
                while(apmat2[i]!=' ')
                {
                    i++;
                }
                i++;
                k=0;
                for(j=i;apmat2[j]!='\0';j++)
                {
                    apmat3[k]=apmat2[j];
                    k++;
                }
                apmat3[k]='\0';
                r=1;
            }
        }
        if(espacio(apmat3))
        {
            if(compName(apmat3))
            {
                i=0;
                while(apmat3[i]!=' ')
                {
                    i++;
                }
                i++;
                k=0;
                for(j=i;apmat3[j]!='\0';j++)
                {
                    apmat4[k]=apmat3[j];
                    k++;
                }
                apmat4[k]='\0';
                r=2;
            }
        }
        if(r==0)
        {
            curp[2]=apmat2[0];
            curp[14]=consInterna(apmat2);
        }
        else
        {
            if(r==1)
            {
                curp[2]=apmat3[0];
                curp[14]=consInterna(apmat3);
            }
            else
            {
                if(r==2)
                {
                    curp[2]=apmat4[0];
                    curp[14]=consInterna(apmat4);
                }
            }
        }
    }
}

int compName(char name[])
{
    int i,j,suma;
    char no_validName[27][strlen(name)]={
        "MARIA","MA","MA.","M.","M","JOSE","J.","J","DA","DAS","DE","DEL","DER",
        "DI","DIE","DD","Y","EL","LA","LOS","LAS","LE","LES","MAC","MC","VAN","VON"
    };

    for(j=0;j<27;j++)
    {
        suma=0;
        for(i=0;name[i]!=' ';i++)
        {

            if(name[i]==no_validName[j][i])
            {
                suma++;
            }
        }

        if(suma==i)
        {
            suma++;
            return 1;
        }
    }
    return 0;
}

int compAp(char ap[])
{
    int i,j,suma=0;
    char no_validAp[19][strlen(ap)]={
        "DA","DAS","DE","DEL","DER","DI","DIE","DD","Y",
        "EL","LA","LOS","LAS","LE","LES","MAC","MC","VAN",
        "VON"
    };

    for(j=0;j<19;j++)
    {
        suma=0;
        for(i=0;ap[i]!=' ';i++)
        {

            if(ap[i]==no_validAp[j][i])
            {
                suma++;
            }
        }

        if(suma==i)
        {
            suma++;
            return 1;
        }
    }
    return 0;
}

int simbolo(char c1)
{
    switch(c1)
    {
        case 39:
        case 45:
```



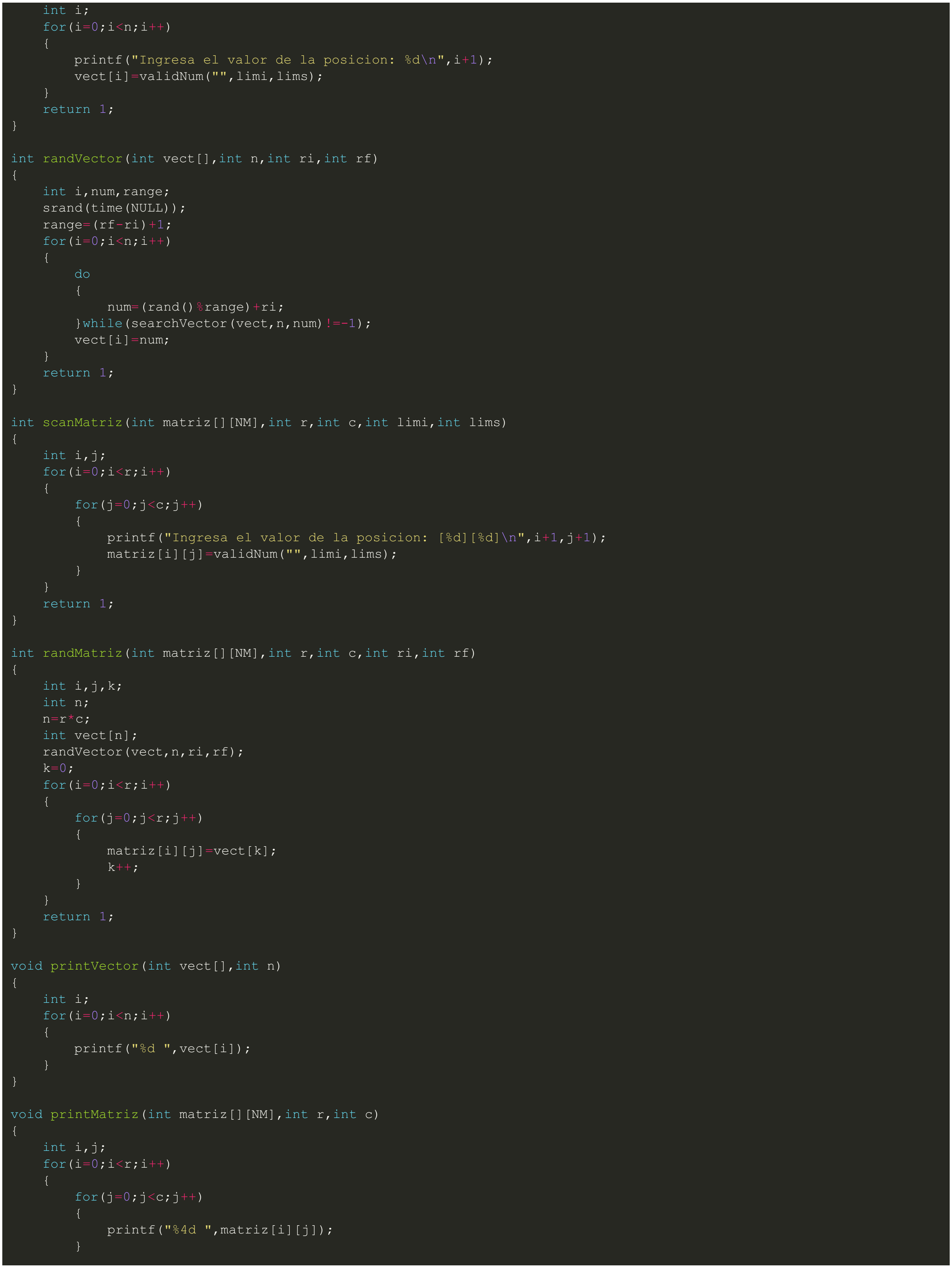
```
        case 47:
        case 46:
            return 1;
            break;
        default:
            return 0;
    }
}

int espacio(char str[])
{
    int i;
    for(i=0;str[i]!='\0';i++)
    {
        if(str[i]==' ')
        {
            return 1;
        }
    }
    return 0;
}

void validSimbolo(char curp[])
{
    int i;
    for(i=0;curp[i]!='\0';i++)
    {
        if(simbolo(curp[i]))
        {
            curp[i]='X';
        }
    }
}

int validPalabraInconven(char curp[])
{
    int i,j;
    int suma;
    char inconv[81][5]={
    "BACA","LOCO","BAKA","LOKA","BUEI","LOKO","BUEY","MAME","CACA","MAMO",
    "CACO","MEAR","CAGA","MEAS","CAGO","MEON","CAKA","MIAR","CAKO","MION",
    "COGE","MOCO","COGI","MOKO","COJA","MULA","COJE","MULO","COJI","NACA",
    "COJO","NACO","COLA","PEDA","CULO","PEDO","FALO","PENE","FETO","PIPI",
    "GETA","PITO","GUEI","POPO","GUEY","PUTA","JETA","PUTO","JOTO","QULO",
    "KACA","RATA","KACO","ROBA","KAGA","ROBE","KAGO","ROBO","KAKA","RUIN",
    "KAKO","SENO","KOGE","TETA","KOGI","VACA","KOJA","VAGA","KOJE","VAGO",
    "KOJI","VAKA","KOJO","VUEI","KOLA","VUEY","KULO","WUEI","LILO","WUEY",
    "LOCA"
    };

    for(i=0;i<81;i++)
    {
        suma=0;
        for(j=0;j<4;j++)
        {
            if(curp[j]==inconv[i][j])
            {
                suma++;
            }
        }
        if(suma==j)
        {
            return 1;
        }
    }
    return 0;
}
```

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

#define NM 4
#define STR 100

//** 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);

int validNum(char txt[],int limi,int lims)
{
    char r[STR];
    int rint;
```



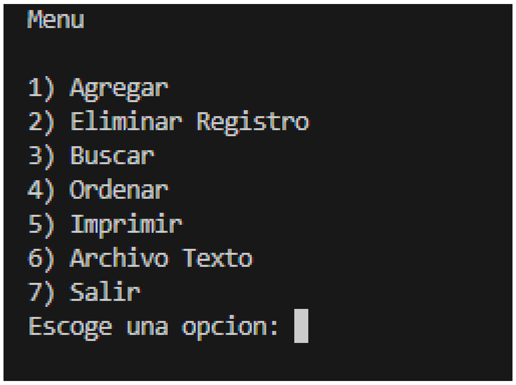
```
    do
    {
        printf("%s",txt);
        fflush(stdin);
        gets(r);
        rint=atoi(r);
        if(rint<limi || rint>lims)
        {
            printf("Valor invalido, ingresalo de nuevo\n");
        }
    }while(rint<limi || rint>lims);
    return rint;
}

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';
}
```

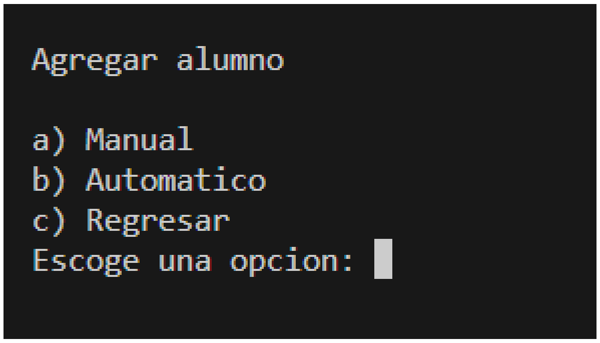
Pruebas de escritorio:

Menu:

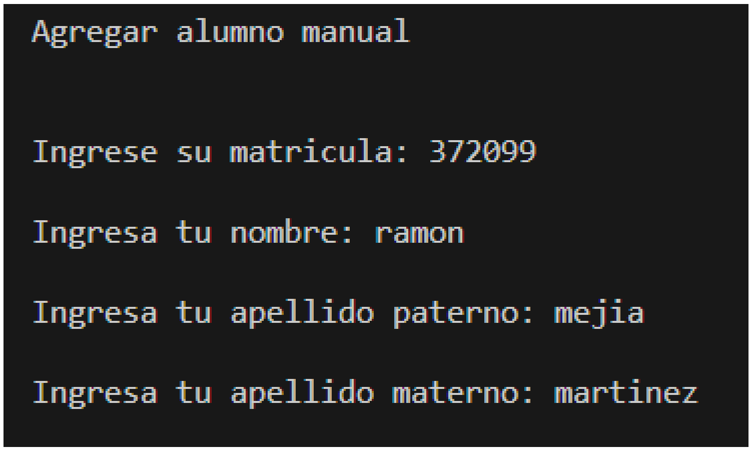


```
Menu

1) Agregar
2) Eliminar Registro
3) Buscar
4) Ordenar
5) Imprimir
6) Archivo Texto
7) Salir
Escoge una opcion:
```

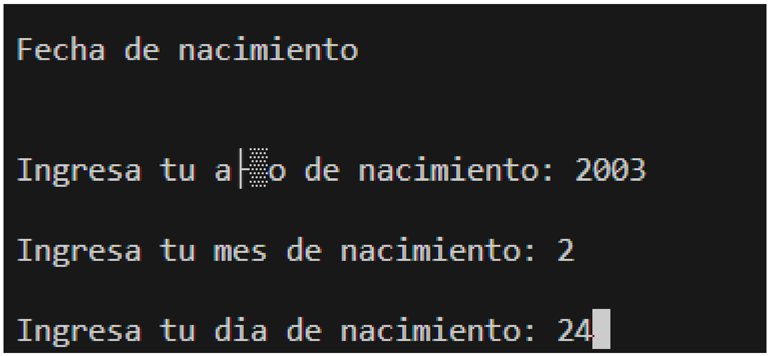
1) Agregar:



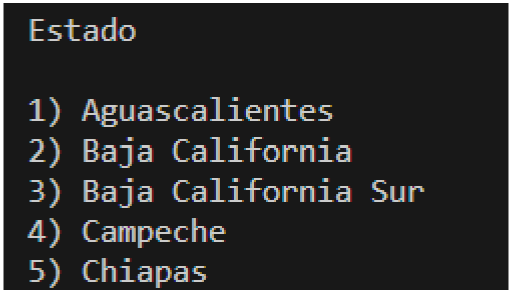
```
Agregar alumno

a) Manual
b) Automatico
c) Regresar
Escoge una opcion:
```



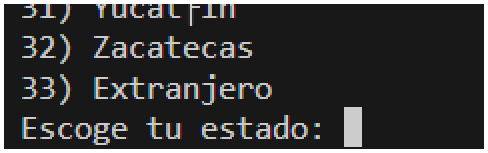
```
Agregar alumno manual


Ingrese su matricula: 372099

Ingresa tu nombre: ramon

Ingresa tu apellido paterno: mejia

Ingresa tu apellido materno: martinez
```



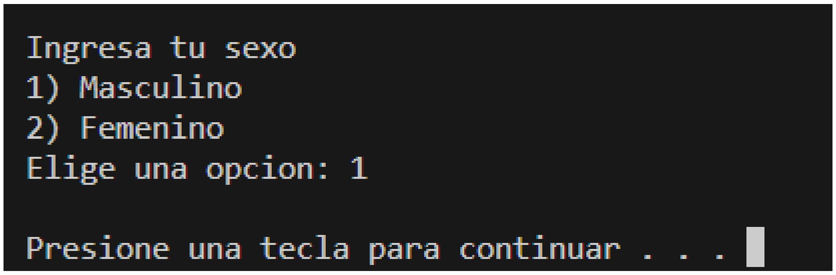
```
Fecha de nacimiento


Ingresa tu a├▒o de nacimiento: 2003

Ingresa tu mes de nacimiento: 2

Ingresa tu dia de nacimiento: 24
```



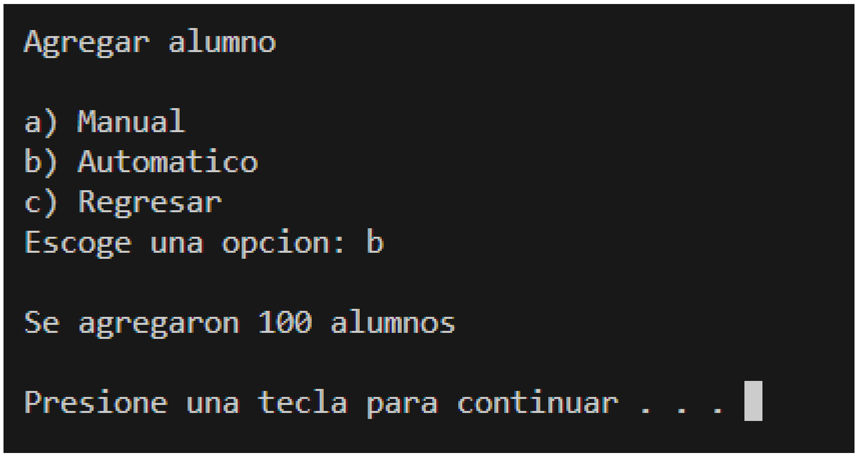
```
Estado

1) Aguascalientes
2) Baja California
3) Baja California Sur
4) Campeche
5) Chiapas
```



```
31) Yucat├ín
32) Zacatecas
33) Extranjero
Escoge tu estado:
```

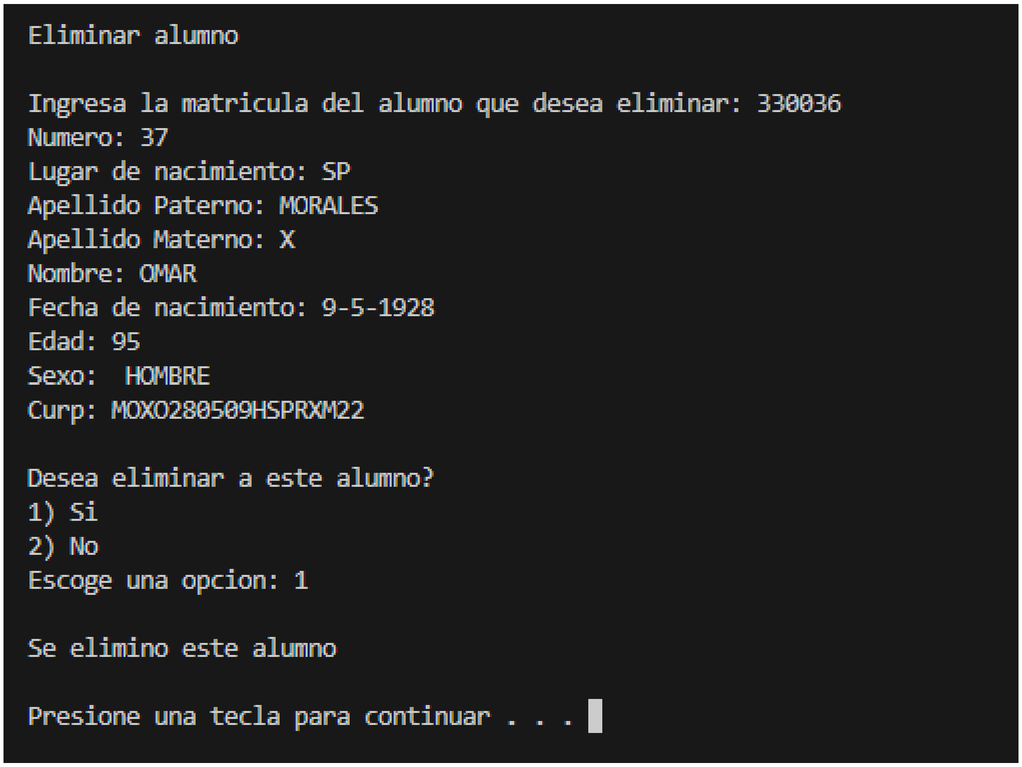


```
Ingresa tu sexo
1) Masculino
2) Femenino
Elige una opcion: 1

Presione una tecla para continuar . . .
```



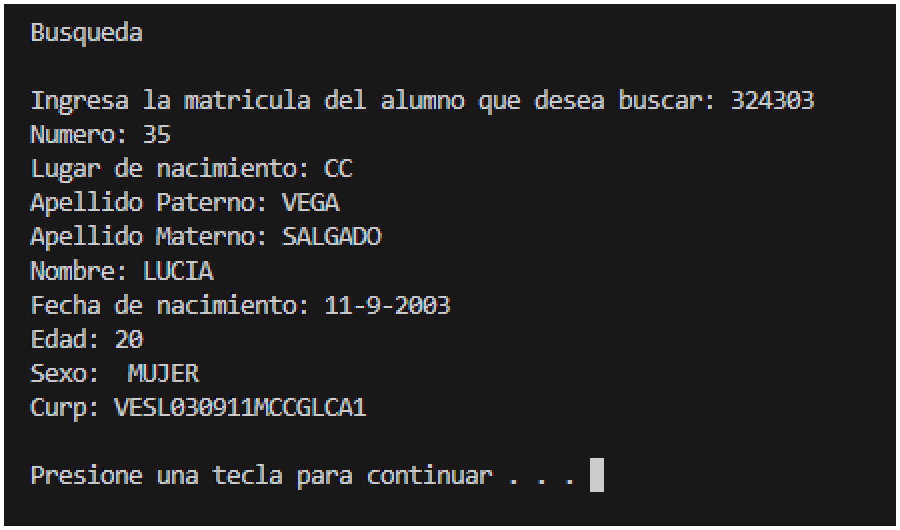
```
Agregar alumno

a) Manual
b) Automatico
c) Regresar
Escoge una opcion: b

Se agregaron 100 alumnos

Presione una tecla para continuar . . .
```

2) Eliminar Registro:



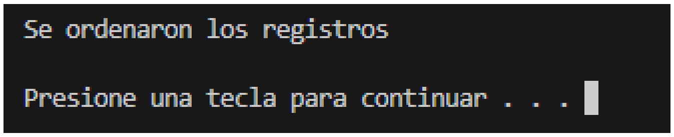
```
Eliminar alumno

Ingresa la matricula del alumno que desea eliminar: 330036
Numero: 37
Lugar de nacimiento: SP
Apellido Paterno: MORALES
Apellido Materno: X
Nombre: OMAR
Fecha de nacimiento: 9-5-1928
Edad: 95
Sexo:  HOMBRE
Curp: MOXO280509HSPRXM22

Desea eliminar a este alumno?
1) Si
2) No
Escoge una opcion: 1

Se elimino este alumno

Presione una tecla para continuar . . .
```

3) Buscar:

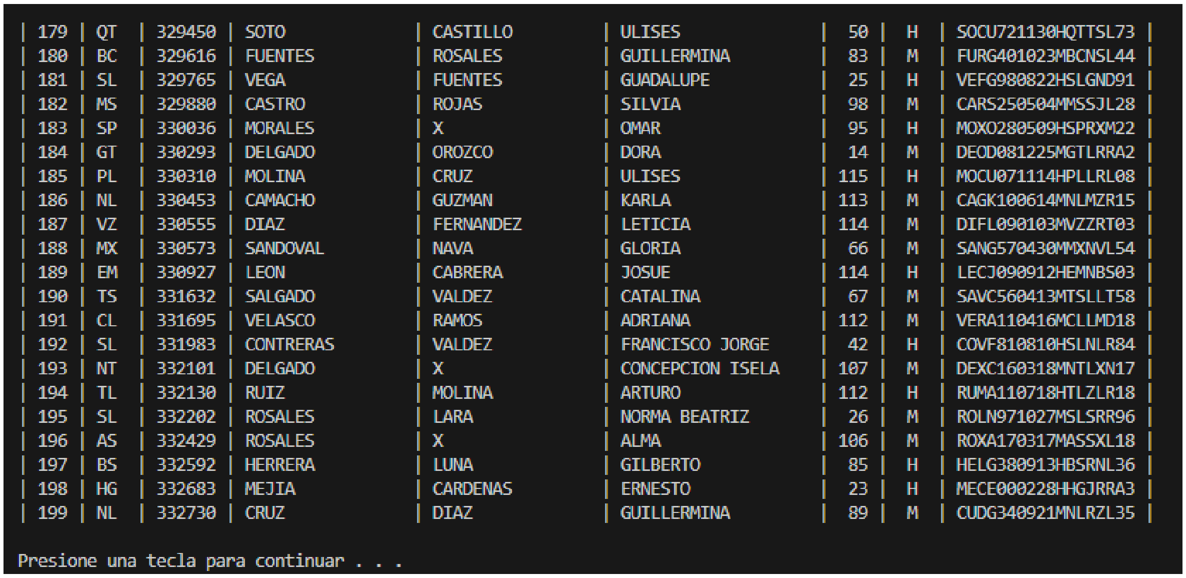


```
Busqueda

Ingresa la matricula del alumno que desea buscar: 324303
Numero: 35
Lugar de nacimiento: CC
Apellido Paterno: VEGA
Apellido Materno: SALGADO
Nombre: LUCIA
Fecha de nacimiento: 11-9-2003
Edad: 20
Sexo:  MUJER
Curp: VESL030911MCCGLCA1

Presione una tecla para continuar . . .
```

4) Ordenar:

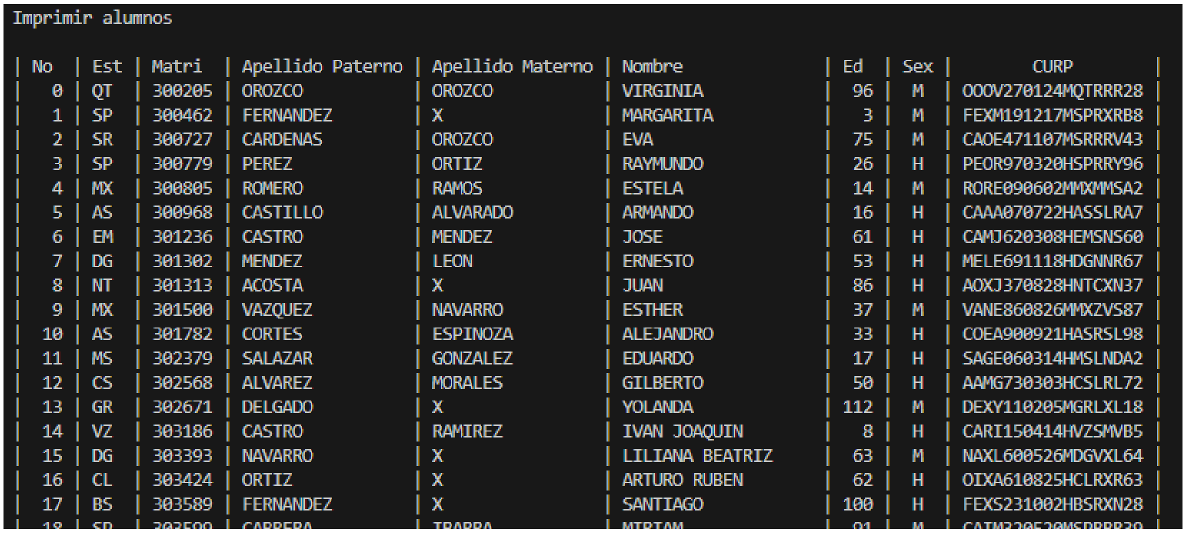


```
Se ordenaron los registros

Presione una tecla para continuar . . .
```



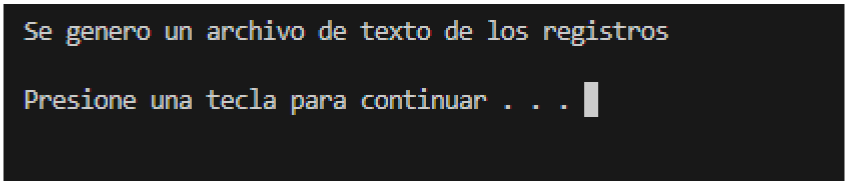
```
| 179 | QT  | 329450 | SOTO             | CASTILLO         | ULISES             |  50 |  H  | SOCU721130HQTTSL73 |
| 180 | BC  | 329616 | FUENTES          | ROSALES          | GUILLERMINA        |  83 |  M  | FURG401023MBCNSL44 |
| 181 | SL  | 329765 | VEGA             | FUENTES          | GUADALUPE          |  25 |  H  | VEFG980822HSLGND91 |
| 182 | MS  | 329880 | CASTRO           | ROJAS            | SILVIA             |  98 |  M  | CARS250504MMSSJL28 |
| 183 | SP  | 330036 | MORALES          | X                | OMAR               |  95 |  H  | MOXO280509HSPRXM22 |
| 184 | GT  | 330293 | DELGADO          | OROZCO           | DORA               |  14 |  M  | DEOD081225MGTLRRA2 |
| 185 | PL  | 330310 | MOLINA           | CRUZ             | ULISES             | 115 |  H  | MOCU071114HPLLRL08 |
| 186 | NL  | 330453 | CAMACHO          | GUZMAN           | KARLA              | 113 |  M  | CAGK100614MNLMZR15 |
| 187 | VZ  | 330555 | DIAZ             | FERNANDEZ        | LETICIA            | 114 |  M  | DIFL090103MVZZRT03 |
| 188 | MX  | 330573 | SANDOVAL         | NAVA             | GLORIA             |  66 |  M  | SANG570430MMXNVL54 |
| 189 | EM  | 330927 | LEON             | CABRERA          | JOSUE              | 114 |  H  | LECJ090912HEMNBS03 |
| 190 | TS  | 331632 | SALGADO          | VALDEZ           | CATALINA           |  67 |  M  | SAVC560413MTSLLT58 |
| 191 | CL  | 331695 | VELASCO          | RAMOS            | ADRIANA            | 112 |  M  | VERA110416MCLLMD18 |
| 192 | SL  | 331983 | CONTRERAS        | VALDEZ           | FRANCISCO JORGE    |  42 |  H  | COVF810810HSLNLR84 |
| 193 | NT  | 332101 | DELGADO          | X                | CONCEPCION ISELA   | 107 |  M  | DEXC160318MNTLXN17 |
| 194 | TL  | 332130 | RUIZ             | MOLINA           | ARTURO             | 112 |  H  | RUMA110718HTLZLR18 |
| 195 | SL  | 332202 | ROSALES          | LARA             | NORMA BEATRIZ      |  26 |  M  | ROLN971027MSLSRR96 |
| 196 | AS  | 332429 | ROSALES          | X                | ALMA               | 106 |  M  | ROXA170317MASSXL18 |
| 197 | BS  | 332592 | HERRERA          | LUNA             | GILBERTO           |  85 |  H  | HELG380913HBSRNL36 |
| 198 | HG  | 332683 | MEJIA            | CARDENAS         | ERNESTO            |  23 |  H  | MECE000228HHGJRRA3 |
| 199 | NL  | 332730 | CRUZ             | DIAZ             | GUILLERMINA        |  89 |  M  | CUDG340921MNLRZL35 |

Presione una tecla para continuar . . .
```

5) Imprimir:

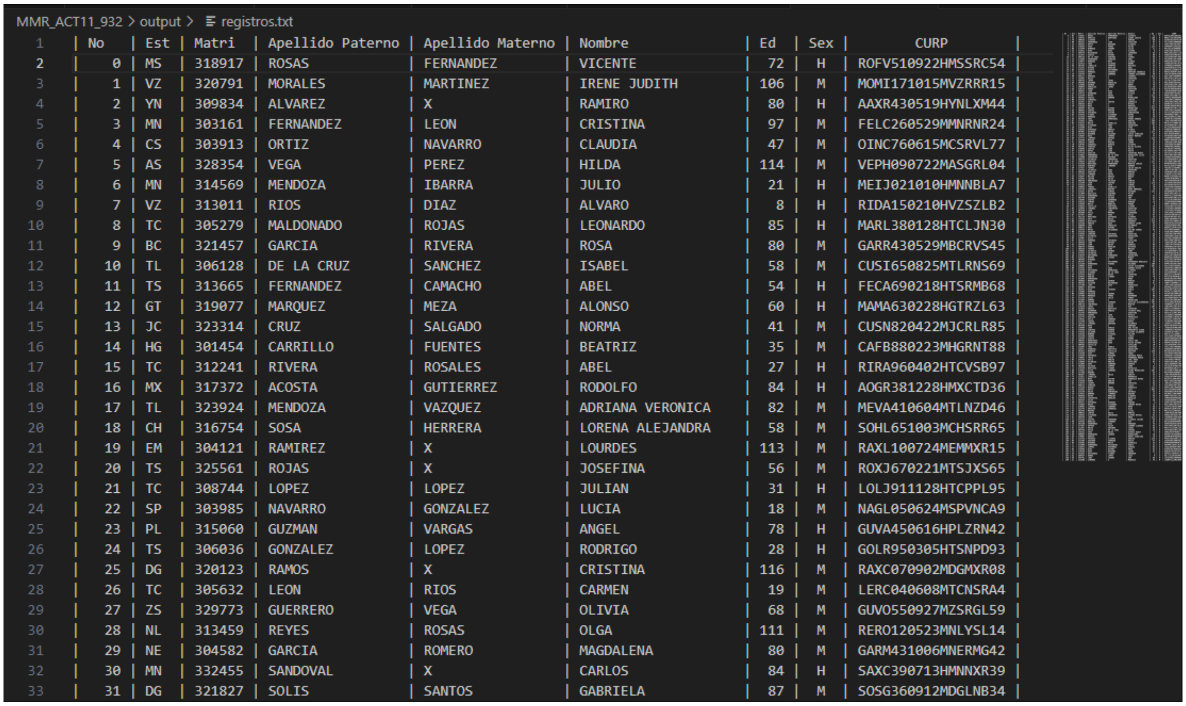


```
Imprimir alumnos

| No  | Est | Matri  | Apellido Paterno | Apellido Materno | Nombre             | Ed  | Sex |        CURP        |
|   0 | QT  | 300205 | OROZCO           | OROZCO           | VIRGINIA           |  96 |  M  | OOOV270124MQTRRR28 |
|   1 | SP  | 300462 | FERNANDEZ        | X                | MARGARITA          |   3 |  M  | FEXM191217MSPRXRB8 |
|   2 | SR  | 300727 | CARDENAS         | OROZCO           | EVA                |  75 |  M  | CAOE471107MSRRRV43 |
|   3 | SP  | 300779 | PEREZ            | ORTIZ            | RAYMUNDO           |  26 |  H  | PEOR970320HSPRRY96 |
|   4 | MX  | 300805 | ROMERO           | RAMOS            | ESTELA             |  14 |  M  | RORE090602MMXMMSA2 |
|   5 | AS  | 300968 | CASTILLO         | ALVARADO         | ARMANDO            |  16 |  H  | CAAA070722HASSLRA7 |
|   6 | EM  | 301236 | CASTRO           | MENDEZ           | JOSE               |  61 |  H  | CAMJ620308HEMSNS60 |
|   7 | DG  | 301302 | MENDEZ           | LEON             | ERNESTO            |  53 |  H  | MELE691118HDGNNR67 |
|   8 | NT  | 301313 | ACOSTA           | X                | JUAN               |  86 |  H  | AOXJ370828HNTCXN37 |
|   9 | MX  | 301500 | VAZQUEZ          | NAVARRO          | ESTHER             |  37 |  M  | VANE860826MMXZVS87 |
|  10 | AS  | 301782 | CORTES           | ESPINOZA         | ALEJANDRO          |  33 |  H  | COEA900921HASRSL98 |
|  11 | MS  | 302379 | SALAZAR          | GONZALEZ         | EDUARDO            |  17 |  H  | SAGE060314HMSLNDA2 |
|  12 | CS  | 302568 | ALVAREZ          | MORALES          | GILBERTO           |  50 |  H  | AAMG730303HCSLRL72 |
|  13 | GR  | 302671 | DELGADO          | X                | YOLANDA            | 112 |  M  | DEXY110205MGRLXL18 |
|  14 | VZ  | 303186 | CASTRO           | RAMIREZ          | IVAN JOAQUIN       |   8 |  H  | CARI150414HVZSMVB5 |
|  15 | DG  | 303393 | NAVARRO          | X                | LILIANA BEATRIZ    |  63 |  M  | NAXL600526MDGVXL64 |
|  16 | CL  | 303424 | ORTIZ            | X                | ARTURO RUBEN       |  62 |  H  | OIXA610825HCLRXR63 |
|  17 | BS  | 303589 | FERNANDEZ        | X                | SANTIAGO           | 100 |  H  | FEXS231002HBSRXN28 |
|  18 | SP  | 303599 | CABRERA          | IBARRA           | MIRIAM             |  91 |  M  | CAIM320520MSPRRR39 |
```

6) Archivo Texto:



```
Se genero un archivo de texto de los registros

Presione una tecla para continuar . . .
```



```
| No   | Est | Matri  | Apellido Paterno | Apellido Materno | Nombre             | Ed  | Sex |        CURP        |
|    0 | MS  | 318917 | ROSAS            | FERNANDEZ        | VICENTE            |  72 |  H  | ROFV510922HMSSRC54 |
|    1 | VZ  | 320791 | MORALES          | MARTINEZ         | IRENE JUDITH       | 106 |  M  | MOMI171015MVZRRR15 |
|    2 | YN  | 309834 | ALVAREZ          | X                | RAMIRO             |  80 |  H  | AAXR430519HYNLXM44 |
|    3 | MN  | 303161 | FERNANDEZ        | LEON             | CRISTINA           |  97 |  M  | FELC260529MMNRNR24 |
|    4 | CS  | 303913 | ORTIZ            | NAVARRO          | CLAUDIA            |  47 |  M  | OINC760615MCSRVL77 |
|    5 | AS  | 328354 | VEGA             | PEREZ            | HILDA              | 114 |  M  | VEPH090722MASGRL04 |
|    6 | MN  | 314569 | MENDOZA          | IBARRA           | JULIO              |  21 |  H  | MEIJ021010HMNNBLA7 |
|    7 | VZ  | 313011 | RIOS             | DIAZ             | ALVARO             |   8 |  H  | RIDA150210HVZSZLB2 |
|    8 | TC  | 305279 | MALDONADO        | ROJAS            | LEONARDO           |  85 |  H  | MARL380128HTCLJN30 |
|    9 | BC  | 321457 | GARCIA           | RIVERA           | ROSA               |  80 |  M  | GARR430529MBCRVS45 |
|   10 | TL  | 306128 | DE LA CRUZ       | SANCHEZ          | ISABEL             |  58 |  M  | CUSI650825MTLRNS69 |
|   11 | TS  | 313665 | FERNANDEZ        | CAMACHO          | ABEL               |  54 |  H  | FECA690218HTSRMB68 |
|   12 | GT  | 319077 | MARQUEZ          | MEZA             | ALONSO             |  60 |  H  | MAMA630228HGTRZL63 |
|   13 | JC  | 323314 | CRUZ             | SALGADO          | NORMA              |  41 |  M  | CUSN820422MJCRLR85 |
|   14 | HG  | 301454 | CARRILLO         | FUENTES          | BEATRIZ            |  35 |  M  | CAFB880223MHGRNT88 |
|   15 | TC  | 312241 | RIVERA           | ROSALES          | ABEL               |  27 |  H  | RIRA960402HTCVSB97 |
|   16 | MX  | 317372 | ACOSTA           | GUTIERREZ        | RODOLFO            |  84 |  H  | AOGR381228HMXCTD36 |
|   17 | TL  | 323924 | MENDOZA          | VAZQUEZ          | ADRIANA VERONICA   |  82 |  M  | MEVA410604MTLNZD46 |
|   18 | CH  | 316754 | SOSA             | HERRERA          | LORENA ALEJANDRA   |  58 |  M  | SOHL651003MCHSRR65 |
|   19 | EM  | 304121 | RAMIREZ          | X                | LOURDES            | 113 |  M  | RAXL100724MEMMXR15 |
|   20 | TS  | 325561 | ROJAS            | X                | JOSEFINA           |  56 |  M  | ROXJ670221MTSJXS65 |
|   21 | TC  | 308744 | LOPEZ            | LOPEZ            | JULIAN             |  31 |  H  | LOLJ911128HTCPPL95 |
|   22 | SP  | 303985 | NAVARRO          | GONZALEZ         | LUCIA              |  18 |  M  | NAGL050624MSPVNCA9 |
|   23 | PL  | 315060 | GUZMAN           | VARGAS           | ANGEL              |  78 |  H  | GUVA450616HPLZRN42 |
|   24 | TS  | 306036 | GONZALEZ         | LOPEZ            | RODRIGO            |  28 |  H  | GOLR950305HTSNPD93 |
|   25 | DG  | 320123 | RAMOS            | X                | CRISTINA           | 116 |  M  | RAXC070902MDGMXR08 |
|   26 | TC  | 305632 | LEON             | RIOS             | CARMEN             |  19 |  M  | LERC040608MTCNSRA4 |
|   27 | ZS  | 329773 | GUERRERO         | VEGA             | OLIVIA             |  68 |  M  | GUVO550927MZSRGL59 |
|   28 | NL  | 313459 | REYES            | ROSAS            | OLGA               | 111 |  M  | RERO120523MNLYSL14 |
|   29 | NE  | 304582 | GARCIA           | ROMERO           | MAGDALENA          |  80 |  M  | GARM431006MNERMG42 |
|   30 | MN  | 332455 | SANDOVAL         | X                | CARLOS             |  84 |  H  | SAXC390713HMNNXR39 |
|   31 | DG  | 321827 | SOLIS            | SANTOS           | GABRIELA           |  87 |  M  | SOSG360912MDGLNB34 |
```