UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

Ingeniería en computación

Actividad 11

Materia: Programación estructurada

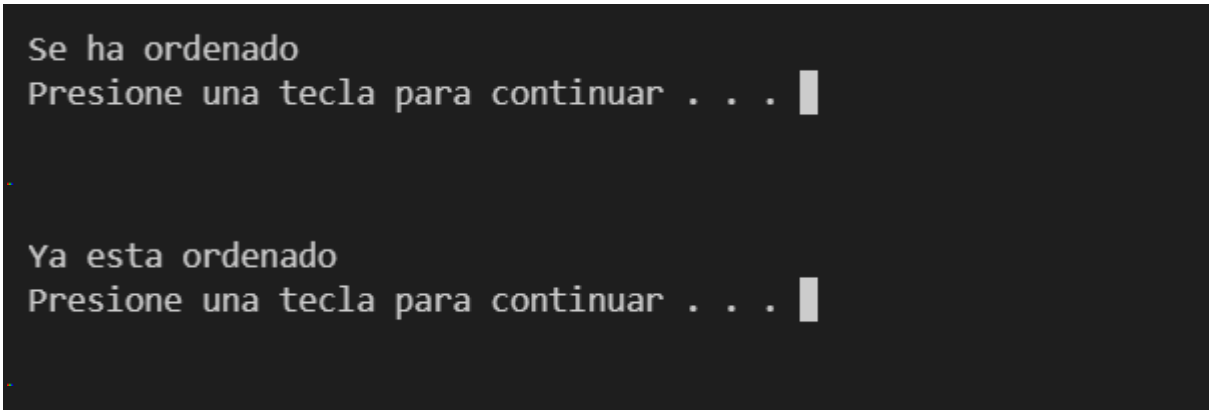
ALUMNO: Villalobos Ensaldo Luis Daniel
MATRÍCULA: 368617
GRUPO: 441

PROFESOR: Pedro Nunez Yepiz

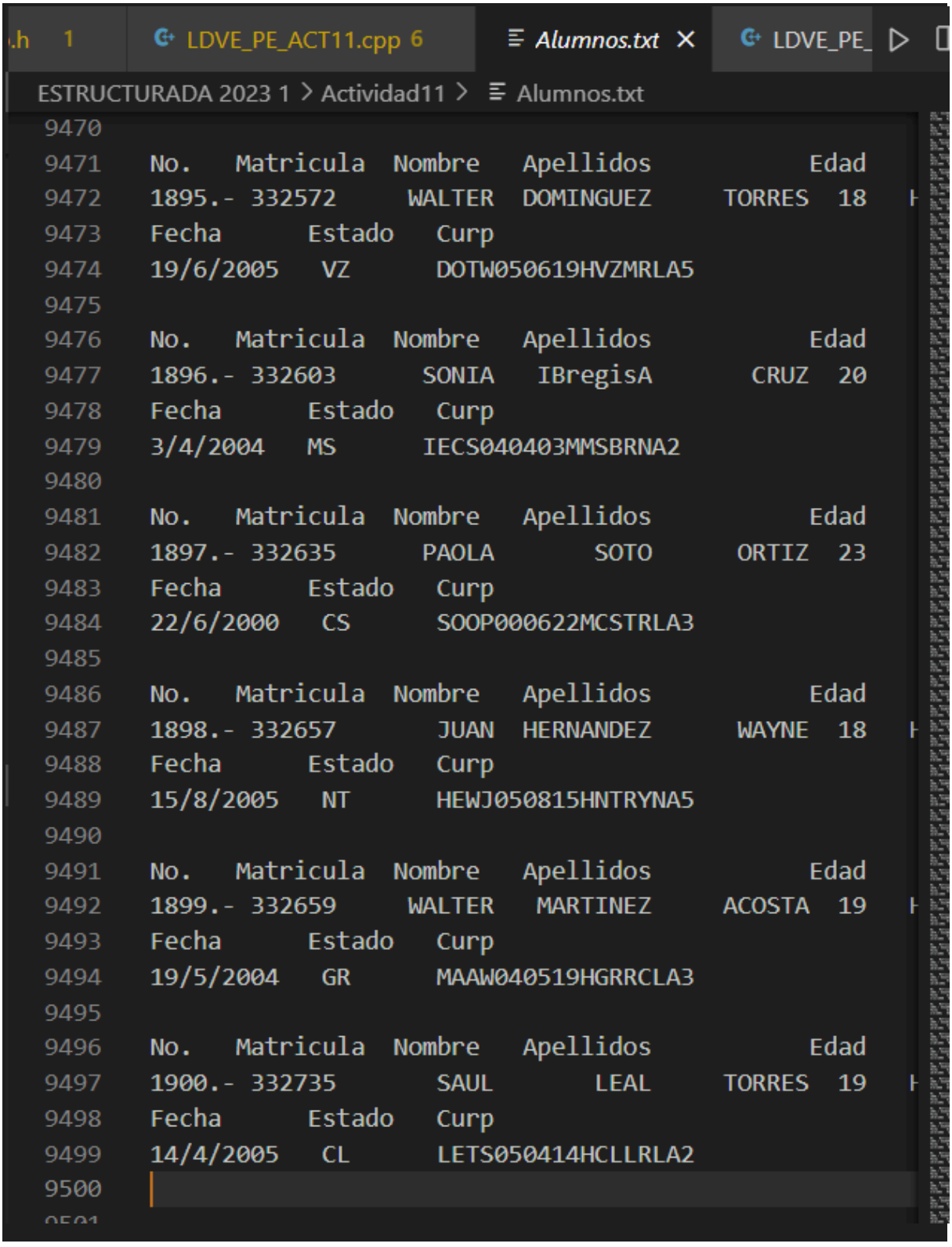
Ensenada, Baja California a 1 de mayo de 2024.

```
No.   Matricula  Nombre   Apellidos          Edad    Sexo
 38.- 318375      PAOLA   ESPINOZA     TORRES  20    MUJER
Fecha      Estado   Curp
11/10/2003   HG       EITP031011MHGSRLA2

No.   Matricula  Nombre   Apellidos          Edad    Sexo
 39.- 319572      ANGEL   MARTINEZ    CARMINE  22   HOMBRE
Fecha      Estado   Curp
19/5/2001   DF       MACA010519HDFRRNA3

No.   Matricula  Nombre   Apellidos          Edad    Sexo
 40.- 303132      SOFIA   MARTINEZ     ACOSTA  23    MUJER
Fecha      Estado   Curp
19/4/2001   PL       MAAS010419MPLRCFA2

Presione una tecla para continuar . . .
```

```
No caben mas registros
Presione una tecla para continuar . . .


Se han agregado 100 registros
Presione una tecla para continuar . . .


1898.- 317386      SONIA     PINEDA     ACOSTA  22    MUJER
Fecha      Estado   Curp
18/8/2001   AS       PIAS010818MASNCNA1

No.   Matricula  Nombre   Apellidos          Edad    Sexo
1899.- 329558       ALAN      PEREZ     TORRES  22   HOMBRE
Fecha      Estado   Curp
15/10/2001   CH       PETA011015HCHRRLA2

No.   Matricula  Nombre   Apellidos          Edad    Sexo
1900.- 307311       LUIS VILLALOBOS       CRUZ  20   HOMBRE
Fecha      Estado   Curp
17/12/2003   YN       VICL031217HYNLRSA1

Presione una tecla para continuar . . .
```


Se ha ordenado
Presione una tecla para continuar . . .
Ya esta ordenado
Presione una tecla para continuar . . .

```
No.   Matricula  Nombre   Apellidos          Edad    Sexo

1896.- 332603      SONIA   IBregisA       CRUZ  20    MUJER

Fecha      Estado   Curp
3/4/2004   MS       IECS040403MMSBRNA2

No.   Matricula  Nombre   Apellidos          Edad    Sexo
1897.- 332635      PAOLA       SOTO      ORTIZ  23    MUJER

Fecha      Estado   Curp
22/6/2000   CS       SOOP000622MCSTRLA3

No.   Matricula  Nombre   Apellidos          Edad    Sexo

1898.- 332657       JUAN  HERNANDEZ      WAYNE  18   HOMBRE

Fecha      Estado   Curp
15/8/2005   NT       HEWJ050815HNTRYNA5
No.   Matricula  Nombre   Apellidos          Edad    Sexo
1898.- 332657       JUAN  HERNANDEZ      WAYNE  18   HOMBRE
Fecha      Estado   Curp
15/8/2005   NT       HEWJ050815HNTRYNA5

No.   Matricula  Nombre   Apellidos          Edad    Sexo
1899.- 332659     WALTER   MARTINEZ     ACOSTA  19   HOMBRE
Fecha      Estado   Curp
19/5/2004   GR       MAAW040519HGRRCLA3

No.   Matricula  Nombre   Apellidos          Edad    Sexo
1900.- 332735       SAUL       LEAL     TORRES  19   HOMBRE
Fecha      Estado   Curp
14/4/2005   CL       LETS050414HCLLRLA2
```


.h 1
LDVE_PE_ACT11.cpp 6
Alumnos.txt
LDVE_PE_
ESTRUCTURADA 2023 1 > Actividad11 > Alumnos.txt
9470
9471 No. Matricula Nombre Apellidos Edad
9472 1895.- 332572 WALTER DOMINGUEZ TORRES 18
9473 Fecha Estado Curp
9474 19/6/2005 VZ DOTW050619HVZMRLA5
9475
9476 No. Matricula Nombre Apellidos Edad
9477 1896.- 332603 SONIA IBregisA CRUZ 20
9478 Fecha Estado Curp
9479 3/4/2004 MS IECS040403MMSBRNA2
9480
9481 No. Matricula Nombre Apellidos Edad
9482 1897.- 332635 PAOLA SOTO ORTIZ 23
9483 Fecha Estado Curp
9484 22/6/2000 CS SOOP000622MCSTRLA3
9485
9486 No. Matricula Nombre Apellidos Edad
9487 1898.- 332657 JUAN HERNANDEZ WAYNE 18
9488 Fecha Estado Curp
9489 15/8/2005 NT HEWJ050815HNTRYNA5
9490
9491 No. Matricula Nombre Apellidos Edad
9492 1899.- 332659 WALTER MARTINEZ ACOSTA 19
9493 Fecha Estado Curp
9494 19/5/2004 GR MAAW040519HGRRCLA3
9495
9496 No. Matricula Nombre Apellidos Edad
9497 1900.- 332735 SAUL LEAL TORRES 19
9498 Fecha Estado Curp
9499 14/4/2005 CL LETS050414HCLLRLA2
9500

```c
/* Luis Daniel Villalobos Ensaldo #368617
Actividad 11   26/04/2024    1/05/2024
Programacion estructurada   gpo:432
1-generar registro
2-generar archivo
3-Eliminar logico
4-buscar 2 tipos si esta ordenada ono
5-Busqueda 2 tipos
6-imprimir registros
*/
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "curp.h"
// cantidad de registros automaticos
#define N 100
// regsitros maximos
#define M 2000
// mostrar registros
#define S 40
typedef struct _tnombre
{
    char appat[20];
    char apmat[20];
    char nombre[20];
} Tnombre;

typedef struct _tfecha
{
    int dia;
    int anio;
    int mes;
    int edad;

} Tfecha;

typedef struct _talum
{
    int status;
    long matricula;
    Tnombre nombre;
    char curp[18];
    Tfecha fecha;
```

```
    char sexo[7];
    char estado[2];
} Talum;

//*** PROTOTIPOS DE FUNCIONES  ******
int msges();
void menu();
Tfecha Fecha(void);
Talum Generar_reg(void);
void Gen_Arch(Talum registros[], int m);
int Buscar(Talum registros[], int m, int matri, bool ordenado = 0);
void Case_4(Talum registros[], int m, bool ordenado = 0);
void Eliminar(Talum registros[], int m, bool ordenado = 0);
int Ordenar(Talum registros[], int m);
void Print_reg(Talum registros[], int m);

//****  main principal  *********
int main()
{
    srand(time(NULL));
    menu();
    return 0;
}
//  *** DESregisOLLO DE LAS FUNCIONES  ******
//*********************
int msges()
{
    int op;
    system("CLS");
    printf("   M  E   N   U \n");
    printf("1.- Cargar \n");
    printf("2.- Archivo Texto \n");
    printf("3.- Eliminar registro \n");
    printf("4.- Buscar \n");
    printf("5.- Ordenar \n");
    printf("6.- Imprimir \n");
    printf("0.- SALIR  \n");
    op = valinum(0, 6, "ESCOGE UNA OPCION: ");

    return op;
}

//****************
```

```
void menu()
{
    int op, no_reg = 0, i;
    bool ordenado = 0;
    Talum registros[M];
    do
    {
        op = msges();
        switch (op)
        {
        case 1:
            system("CLS");
            if (no_reg < (M - N))
            {
                for (i = 0; i < N; i++)
                {
                    registros[no_reg] = Generar_reg();
                    no_reg++;
                }
                ordenado = 0;
                printf("Se han agregado %d registros\n", N);
            }
            else
            {
                printf("No caben mas registros\n");
            }
            system("PAUSE");
            break;
        case 2:
            system("CLS");
            Gen_Arch(registros, no_reg);
            break;
        case 3:
            system("CLS");
            Eliminar(registros, no_reg, ordenado);
            system("PAUSE");

            break;
        case 4:
            Case_4(registros, no_reg, ordenado);
            break;
        case 5:
            system("CLS");
```

```
            if (ordenado)
            {
                printf("Ya esta ordenado\n");
            }
            else
            {
                ordenado = Ordenar(registros, no_reg);
                printf("Se ha ordenado\n");
            }
            system("PAUSE");
            break;
        case 6:
            system("CLS");
            Print_reg(registros, no_reg);
            break;
        }

    } while (op != 0);
}
Tfecha Fecha()
{
    Tfecha fecha;
    fecha.anio = (rand() % 6) + 2000;
    fecha.mes = (rand() % 12) + 1;
    if (fecha.mes == 2)
    {
        if ((fecha.anio % 4) == 0)
        {
            fecha.dia = (rand() % 28) + 1;
        }
        else
        {
            fecha.dia = (rand() % 27) + 1;
        }
    }
    else
    {
        if (fecha.mes == 4 or fecha.mes == 6 or fecha.mes == 9 or
fecha.mes == 11)
        {
            fecha.dia = (rand() % 29) + 1;
        }
        else
```

```
        {
            fecha.dia = (rand() % 30) + 1;
        }
    }
    // fechas actuales
    time_t tiempo_actual;
    time(&tiempo_actual);
    struct tm *info_tiempo;
    info_tiempo = localtime(&tiempo_actual);
    int anio = info_tiempo->tm_year + 1900; // Año actual
    int mes = info_tiempo->tm_mon + 1;      // Mes actual
    int dia = info_tiempo->tm_mday;
    // edad
    if (mes <= fecha.mes)
    {
        if (mes == fecha.mes)
        {
            if (dia >= fecha.dia)
            {
                fecha.edad = anio - fecha.anio;
            }
            else
            {
                fecha.edad = anio - fecha.anio - 1;
            }
        }
        else
        {
            fecha.edad = anio - fecha.anio - 1;
        }
    }
    else
    {
        fecha.edad = anio - fecha.anio;
    }
    return fecha;
}
void merge(Talum regis[], int izq, int mitad, int ult)
{
    int i, j, k;
    int n1 = mitad - izq + 1;
    int n2 = ult - mitad;
```

```
    Talum L[n1], R[n2];

    for (i = 0; i < n1; ++i)
        L[i] = regis[izq + i];
    for (j = 0; j < n2; ++j)
        R[j] = regis[mitad + 1 + j];
    i = 0;
    j = 0;
    k = izq;

    while (i < n1 && j < n2)
    {
        if (L[i].matricula <= R[j].matricula)
        {
            regis[k] = L[i];
            ++i;
        }
        else
        {
            regis[k] = R[j];
            ++j;
        }
        ++k;
    }

    while (i < n1)
    {
        regis[k] = L[i];
        ++i;
        ++k;
    }

    while (j < n2)
    {
        regis[k] = R[j];
        ++j;
        ++k;
    }
}
void mergeSort(Talum regis[], int izq, int ult)
{
    if (izq < ult)
    {
```

```
        int mitad = izq + (ult - izq) / 2;
        mergeSort(regis, izq, mitad);
        mergeSort(regis, mitad + 1, ult);
        merge(regis, izq, mitad, ult);
    }
}
//*********************
Talum Generar_reg(void)
{
    Talum reg;
    Tnombre copia;

    reg.status = 1;
    char nombres[12][20] = {"JUAN", "LUIS", "DANIEL", "SAUL", "JACOB",
"ANGEL", "ALAN", "YAHIR", "GABRIEL", "ISRAEL", "WALTER"};
    char mujeres[12][20] = {"ANGELA", "SOFIA", "VIVIANA", "ANA",
"MARIA", "PAOLA", "PAULA", "SONIA", "VANESSA", "HANNA", "LOLA"};
    char appat[12][20] = {"SOTO", "VILLALOBOS", "VICTORIO", "PEREZ",
"PINEDA", "HERNANDEZ", "LEAL", "ESPINOZA", "IBregisA", "DIAZ",
"DOMINGUEZ", "MARTINEZ"};
    char apmat[12][20] = {"WAYNE", "CRUZ", "CHAVEZ", "CARMINE", "RUIZ",
"ORTIZ", "TORRES", "OROZCO", "GARCIA", "PAREDES", "ACOSTA", "CASAS"};
    int sexo, estado;
    // fecha
    reg.fecha = Fecha();
    // nombres y apellidos
    copystr(reg.nombre.appat, appat[rand() % 12]);
    copystr(reg.nombre.apmat, apmat[rand() % 12]);
    sexo = (rand() % 2) + 1;
    if (sexo == 2)
    {
        copystr(reg.sexo, "HOMBRE");
        copystr(reg.nombre.nombre, nombres[rand() % 11]);
    }
    else
    {
        copystr(reg.sexo, "MUJER");
        copystr(reg.nombre.nombre, mujeres[rand() % 11]);
    }
    reg.matricula = (rand() % 99999) + 300000;
    estado = (rand() % 33) + 1;
    copia = reg.nombre;
```

```
    Curp(reg.curp, copia.nombre, copia.appat, copia.apmat, sexo,
estado, reg.fecha.anio, reg.fecha.mes, reg.fecha.dia);
    reg.estado[0] = reg.curp[11];
    reg.estado[1] = reg.curp[12];
    reg.estado[2] = '\0';

    return reg;
}
//****************************
int Buscar(Talum registros[], int m, int matri, bool ordenado)
{
    int i;

    if (ordenado)
    {
        int bajo = 0, centro, alto = m - 1;
        do
        {
            centro = (bajo + alto) / 2;
            if (registros[centro].matricula < matri)
            {
                bajo = centro + 1;
            }
            else
            {
                if (registros[centro].matricula == matri)
                {
                    return centro;
                }
                else
                {
                    alto = centro - 1;
                }
            }
        } while (alto >= bajo);
    }
    else
    {
        for (i = 0; i < m; i++)
        {
            if (registros[i].matricula == matri)
            {
                return i;
```

```
            }
        }
    }
    return -1;
}
void Case_4(Talum registros[], int m, bool ordenado)
{
    system("CLS");
    int num, resul;
    num = valilong(300000, 399999, "Matricula que busca: ");
    resul = Buscar(registros, m, num, ordenado);
    if (resul == -1)
    {
        printf("No se encontro la matricula\n");
    }
    else
    {
        printf("La matricula esta en: %d\n", (resul + 1));
        printf("No.   Matricula  Nombre   Apellidos\n");
        printf("%3d.- %6ld %10s %10s %10s\n", (resul + 1),
registros[resul].matricula, registros[resul].nombre.nombre,
registros[resul].nombre.appat, registros[resul].nombre.apmat);
    }
    system("PAUSE");
}
void Eliminar(Talum registros[], int m, bool ordenado)
{
    long num;
    int resul;
    bool eliminar;

    num = valilong(300000, 399999, "Matricula que busca: ");
    resul = Buscar(registros, m, num, ordenado);
    if (resul == -1)
    {
        printf("No se encontro la matricula\n");
    }
    else
    {
        printf("La matricula esta en: %d\n", (resul + 1));
        printf("No.   Matricula  Nombre   Apellidos\n");
```

```
        printf("%3d.- %6ld %10s %10s %10s\n", (resul + 1),
registros[resul].matricula, registros[resul].nombre.nombre,
registros[resul].nombre.appat, registros[resul].nombre.apmat);
        eliminar = valinum(0, 1, "(0)Cancelar\n(1)Eliminar\nSeleccione:
");
        if (eliminar)
        {
            registros[resul].status = 0;
            printf("Se elimino el registro\n");
        }
        else
        {
            printf("Cancelando\n");
        }
    }
}
int Ordenar(Talum registros[], int m)
{
    int j;
    Talum temp;
    if (m < 500)
    {
        for (int i = 0; i < m - 1; i++)
        {
            for (j = i + 1; j < m; j++)
            {
                if (registros[j].matricula < registros[i].matricula)
                {
                    temp = registros[i];
                    registros[i] = registros[j];
                    registros[j] = temp;
                }
            }
        }
    }
    else
    {
        mergeSort(registros, 0, m - 1);
    }
    return 1;
}
void Print_reg(Talum registros[], int m)
{
```

```
    for (int i = 0; i < m; i++)
    {
        if (registros[i].status == 1)
        {
            printf("No.   Matricula  Nombre   Apellidos           Edad
Sexo\n");
            printf("%3d.- %6ld %10s %10s %10s %3d %8s\n", (i + 1),
registros[i].matricula, registros[i].nombre.nombre,
registros[i].nombre.appat, registros[i].nombre.apmat,
registros[i].fecha.edad, registros[i].sexo);
            printf("Fecha      Estado   Curp\n");
            printf("%d/%d/%d   %s       %s\n\n",
registros[i].fecha.dia, registros[i].fecha.mes,
registros[i].fecha.anio, registros[i].estado, registros[i].curp);
        }
        if ((i + 1) % S == 0)
        {
            system("PAUSE");
        }
    }
    system("PAUSE");
}
void Gen_Arch(Talum registros[], int m)
{
    FILE *archivo = fopen("Alumnos.txt", "w");

    if (archivo)
    {
        // Escribir en el archivo
        for (int i = 0; i < m; i++)
        {
            if (registros[i].status == 1)
            {
                fprintf(archivo, "No.   Matricula  Nombre   Apellidos
Edad    Sexo\n");
                fprintf(archivo, "%3d.- %6ld %10s %10s %10s %3d %8s\n",
(i + 1), registros[i].matricula, registros[i].nombre.nombre,
registros[i].nombre.appat, registros[i].nombre.apmat,
registros[i].fecha.edad, registros[i].sexo);
                fprintf(archivo, "Fecha      Estado   Curp\n");
                fprintf(archivo, "%d/%d/%d   %s      %s\n\n",
registros[i].fecha.dia, registros[i].fecha.mes,
registros[i].fecha.anio, registros[i].estado, registros[i].curp);
```

```
            }
        }
    }

    // Cerrar el archivo
    fclose(archivo);

    printf("Se ha escrito el archivo\n");
    system("PAUSE");
}
```

Programa1

```
Talum Generar_reg(void)
{
    Talum reg;
    Tnombre copia;

    reg.status = 1;
    char nombres[12][20] = {"JUAN", "LUIS", "DANIEL", "SAUL", "JACOB",
"ANGEL", "ALAN", "YAHIR", "GABRIEL", "ISRAEL", "WALTER"};
    char mujeres[12][20] = {"ANGELA", "SOFIA", "VIVIANA", "ANA",
"MARIA", "PAOLA", "PAULA", "SONIA", "VANESSA", "HANNA", "LOLA"};
    char appat[12][20] = {"SOTO", "VILLALOBOS", "VICTORIO", "PEREZ",
"PINEDA", "HERNANDEZ", "LEAL", "ESPINOZA", "IBregisA", "DIAZ",
"DOMINGUEZ", "MARTINEZ"};
    char apmat[12][20] = {"WAYNE", "CRUZ", "CHAVEZ", "CARMINE", "RUIZ",
"ORTIZ", "TORRES", "OROZCO", "GARCIA", "PAREDES", "ACOSTA", "CASAS"};
    int sexo, estado;
    // fecha
    reg.fecha = Fecha();
    // nombres y apellidos
    copystr(reg.nombre.appat, appat[rand() % 12]);
    copystr(reg.nombre.apmat, apmat[rand() % 12]);
    sexo = (rand() % 2) + 1;
    if (sexo == 2)
    {
        copystr(reg.sexo, "HOMBRE");
        copystr(reg.nombre.nombre, nombres[rand() % 11]);
    }
    else
    {
        copystr(reg.sexo, "MUJER");
        copystr(reg.nombre.nombre, mujeres[rand() % 11]);
    }
    reg.matricula = (rand() % 99999) + 300000;
    estado = (rand() % 33) + 1;
    copia = reg.nombre;
    Curp(reg.curp, copia.nombre, copia.appat, copia.apmat, sexo,
estado, reg.fecha.anio, reg.fecha.mes, reg.fecha.dia);
    reg.estado[0] = reg.curp[11];
    reg.estado[1] = reg.curp[12];
    reg.estado[2] = '\0';

    return reg;
```

```
}
```

Programa2

```
void Gen_Arch(Talum registros[], int m)
{
    FILE *archivo = fopen("Alumnos.txt", "w");

    if (archivo)
    {
        // Escribir en el archivo
        for (int i = 0; i < m; i++)
        {
            if (registros[i].status == 1)
            {
                fprintf(archivo, "No.   Matricula  Nombre   Apellidos
Edad    Sexo\n");
                fprintf(archivo, "%3d.- %6ld %10s %10s %10s %3d %8s\n",
(i + 1), registros[i].matricula, registros[i].nombre.nombre,
registros[i].nombre.appat, registros[i].nombre.apmat,
registros[i].fecha.edad, registros[i].sexo);
                fprintf(archivo, "Fecha      Estado   Curp\n");
                fprintf(archivo, "%d/%d/%d   %s      %s\n\n",
registros[i].fecha.dia, registros[i].fecha.mes,
registros[i].fecha.anio, registros[i].estado, registros[i].curp);
            }
        }
    }

    // Cerrar el archivo
    fclose(archivo);

    printf("Se ha escrito el archivo\n");
    system("PAUSE");
}
```

Programa3

```
void Eliminar(Talum registros[], int m, bool ordenado)
{
    long num;
    int resul;
    bool eliminar;

    num = valilong(300000, 399999, "Matricula que busca: ");
    resul = Buscar(registros, m, num, ordenado);
    if (resul == -1)
    {
        printf("No se encontro la matricula\n");
    }
    else
    {
        printf("La matricula esta en: %d\n", (resul + 1));
        printf("No.   Matricula  Nombre   Apellidos\n");
        printf("%3d.- %6ld %10s %10s %10s\n", (resul + 1),
registros[resul].matricula, registros[resul].nombre.nombre,
registros[resul].nombre.appat, registros[resul].nombre.apmat);
        eliminar = valinum(0, 1, "(0)Cancelar\n(1)Eliminar\nSeleccione:
");
        if (eliminar)
        {
            registros[resul].status = 0;
            printf("Se elimino el registro\n");
        }
        else
        {
            printf("Cancelando\n");
        }
    }
}
```

Programa4

```
int Buscar(Talum registros[], int m, int matri, bool ordenado)
{
    int i;

    if (ordenado)
    {
        int bajo = 0, centro, alto = m - 1;
        do
        {
            centro = (bajo + alto) / 2;
            if (registros[centro].matricula < matri)
            {
                bajo = centro + 1;
            }
            else
            {
                if (registros[centro].matricula == matri)
                {
                    return centro;
                }
                else
                {
                    alto = centro - 1;
                }
            }
        } while (alto >= bajo);
    }
    else
    {
        for (i = 0; i < m; i++)
        {
            if (registros[i].matricula == matri)
            {
                return i;
            }
        }
    }
    return -1;
}
void Case_4(Talum registros[], int m, bool ordenado)
{
    system("CLS");
```

```
    int num, resul;
    num = valilong(300000, 399999, "Matricula que busca: ");
    resul = Buscar(registros, m, num, ordenado);
    if (resul == -1)
    {
        printf("No se encontro la matricula\n");
    }
    else
    {
        printf("La matricula esta en: %d\n", (resul + 1));
        printf("No.   Matricula  Nombre   Apellidos\n");
        printf("%3d.- %6ld %10s %10s %10s\n", (resul + 1),
registros[resul].matricula, registros[resul].nombre.nombre,
registros[resul].nombre.appat, registros[resul].nombre.apmat);
    }
    system("PAUSE");
}
```

Programa5

```
int Ordenar(Talum registros[], int m)
{
    int j;
    Talum temp;
    if (m < 500)
    {
        for (int i = 0; i < m - 1; i++)
        {
            for (j = i + 1; j < m; j++)
            {
                if (registros[j].matricula < registros[i].matricula)
                {
                    temp = registros[i];
                    registros[i] = registros[j];
                    registros[j] = temp;
                }
            }
        }
    }
    else
    {
        mergeSort(registros, 0, m - 1);
    }
    return 1;
}
```

Programa6

```
void Print_reg(Talum registros[], int m)
{
    for (int i = 0; i < m; i++)
    {
        if (registros[i].status == 1)
        {
            printf("No.   Matricula  Nombre   Apellidos           Edad
Sexo\n");
            printf("%3d.- %6ld %10s %10s %10s %3d %8s\n", (i + 1),
registros[i].matricula, registros[i].nombre.nombre,
registros[i].nombre.appat, registros[i].nombre.apmat,
registros[i].fecha.edad, registros[i].sexo);
            printf("Fecha      Estado   Curp\n");
            printf("%d/%d/%d   %s       %s\n\n",
registros[i].fecha.dia, registros[i].fecha.mes,
registros[i].fecha.anio, registros[i].estado, registros[i].curp);
        }
        if ((i + 1) % S == 0)
        {
            system("PAUSE");
        }
    }
    system("PAUSE");
}
```