Universidad Autónoma de Baja California
Facultad de Ingeniería, Arquitectura y Diseño

Ingenieria en computación

Programación Estructurara/36276

Santos Tirado Martin/00369705 Pedro

Nuñez Yepiz

Actividad N. 11

ARCHIVOS

Ensenada Baja California, 2 de novienbre del 2023

Introducción:

El alumno aprenderá a crear por su cuenta un formato archivo el cual usara en C.

Competencia:

El alumno llevará a cabo los ejercicios planteados con ayuda del profesor que dejo en un documento sobre los Archivos.

Fundamentos:

El alumno aprenderá a manejar y acostumbrarse a los códigos largos y complicados con la ayuda de funciones, vectores, matrices, librerías propias, entre otras; también, el alumno desarrollará el como usar fuentes externas via archivos.

Procedimiento:

1.- Realiza un programa en C que utilice una librerías propias
(Funciones de validar números y cadenas)
2.- Realiza reporte de práctica
3.- Sube a Blackboard, programa, librería, y reporte de practica y PDF anexo con capturas y código

Realiza el programa que contenga el siguiente menú

M E N Ú
1.- Cargar
2.- Eliminar
3.- Buscar
4.- Ordenar
5.- Imprimir
6.- Archivo Texto

0.- Salir

El programa deberá poder almacenar en un arreglo (máximo 2,000 registros) los datos para generar el CURP la estructura debe contener 2 estructuras anidadas, nombre y fecha nacimiento y un campo donde se escribirá automáticamente el curp basado en los datos proporcionados

MENÚ DESCRIPCIÓN:

1.- Cargar: Se deberá agregar 100 registros en forma automáticamente y aleatorios (cuidar no se desborde Arreglo)

2.- Eliminar: La búsqueda se realizará por matrícula, Imprimir el registro encontrado en forma de registro y preguntar si quiere eliminar si o no. (Eliminado Lógico x campo status)

3.- Buscar: La búsqueda se realizará por matrícula, el programa deberá ser capaz de realizar la búsqueda secuencial o Binaria según sea el caso. Imprimir el registro encontrado en forma de registro.

4.- Ordenar: La ordenación será por MATRICULA usar función de ordenación más adecuada según sea el caso usar 2 métodos de ordenación y el programa decidirá cuál es el que usará dependiendo del estado y tamaño de registros dentro del arreglo.
Nota: (validar si el arreglo ya está ordenado no volver ordenar por el mismo campo)

5.- Imprimir: El programa deberá imprimir los datos del arreglo (solo registros activos) en forma de tabla en pantallas de 40 registros y presionando la tecla de continuar en cada uno de los casos.

6.- Archivo de Texto: El programa deberá generar un archivo de texto con los datos del arreglo (solo registros activos) formatear salida.

REALICE EL SIGUIENTE PROGRAMA QUE CONTENGA UN MENÚ.

MENÚ

1.- AGREGAR (AUTOM 10 REGISTROS)

2.- AGREGAR MANUAL

3- ELIMINAR REGISTRO (lógico)

4.- BUSCAR

5- ORDENAR

6.- IMPRIMIR

0.- SALIR

UTILIZAR UN ARREGLO DE 500 REGISTROS

SE DEBERÁ UTILIZAR ESTRUCTURAS CON LOS DATOS BÁSICOS DE UN ALUMNO (status, Matricula, ApPat, ApMat, Nombre, Edad, Sexo)

Busqueda y Ordenacion por campo MATRICULA

PROGRAMA:

```
#define P 300
#define N 100
#include "maton.h"
int msg();
void menu();
int validaL(long ri, long rf, const char msg[], const char msgError[]);
int validaInt(int ri, int rf, const char msg[], const char msgError[]);
int busqSec(Tkey *vect, int tam, long matricula);
int ordenarR(Tkey registros[], int tam, int band);
int agregarR(Tkey registros[], int tam);
int agregarM(Tkey registros[], int tam);
int eliminarEsp(char cadena[]);
int soloLetras(char cadena[]);
int existeTkey(long mat, Tkey *reg, int tam);
Tkey datosR(Tkey registros[], int tam);
Tkey datosM(Tkey registros[], int tam);
void validaCad(char cadena[], const char *msg);
void burbuja(Tkey vect[], int tam);
void quickSort(Tkey registros[], int limIzq, int limDer);
void buscarR(Tkey registros[], int tam);
void imprimirR(Tkey vect[], int tam);
void eliminarR(Tkey registros[], int tam);
void archivo(Tkey vect[],int tam);

int main()
{
    menu();
    return 0;
}
int msg()
{
    int op;
    printf("\tMENU\n1-Cargar\n2-Eliminar\n3-Buscar\n4-Ordenar\n5-
Imprimir\n6-Archivo Texto\n0-Salir\n");
    op = validaInt(0,6,"Escoge una de la opcion\n","Error, no escogiste una
de las opciones");
    return op;
}
void menu()
{
```

```
    Tkey reg[P];
    int op, band, tam;
    tam = 0;
    band = 0;

    do{
        op = msg();
        switch(op)
        {
            case 1:
                tam = agregarR(reg, tam);
                tam = agregarM(reg, tam);
                break;
            case 2:
                eliminarR(reg, tam);
                break;
            case 3:
                buscarR(reg, tam);
                break;
            case 4:
                band = ordenarR(reg, tam, band);
                break;
            case 5:
                imprimirR(reg, tam);
                break;
            case 6:
                archivo(reg,tam);
                break;
            case 0:
                printf("Has escogido salir del menu");
                break;
        }

    }while (op != 0);
}
```

LIBRERÍA:

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
typedef struct Tkey
{
    long matri;
    char nombre[31];
    char apellidoP[31];
    char apellidoM[31];
    int status;
    int genero;
    int edad;

}Tkey;
int validaL(long ri, long rf,const char msg[],const char msgError[])
{
    long num;
    char xnum[30];

    do
    {
        puts(msg);
        fflush(stdin);
        gets(xnum);
        num = atoi(xnum);

        if (num < ri || num > rf)
        {
            printf("%s", msgError);
            printf("\n");
        }
    } while (num < ri || num > rf);

    return num;
}
int validaInt(int ri, int rf,const char msg[],const char msgError[])
{
    int num;
    char xnum[30];
```

```
    do
    {
        puts(msg);
        fflush(stdin);
        gets(xnum);
        num = atoi(xnum);

        if (num < ri || num > rf)
        {
            printf("%s", msgError);
            printf("\n");
        }
    } while (num < ri || num > rf);

    return num;
}
int busqSec(Tkey *registros, int tam, long matricula)
{
    int i;
    for (i = 0; i < tam; i++)
    {
        if(registros[i].matri == matricula)
        {
            return i;
        }
    }

    return -1;
}
void burbuja(Tkey vect[], int tam)
{
    system("CLS");
    int i, j;
    Tkey burbuja;
    for(i = 0; i < tam - 1; i++)
    {
```

```
        for(j = i + 1; j < tam; j++)
```

```
        {
            if (vect[j].matri < vect[i].matri)
            {
                burbuja = vect[j];
                vect[j] = vect[i];
                vect[i] = burbuja;
            }
        }
    }

    system("PAUSE");
}
void quickSort(Tkey registros[], int limIzq, int limDer)
{
    int izq, der, tem;
    Tkey central;

    izq = limIzq;
    der = limDer;
    central.matri = registros[(izq + der) / 2].matri;

    do{

        while(registros[izq].matri < central.matri && izq < limDer)
        {
            izq++;
        }

        while(central.matri < registros[der].matri && der > limIzq)
        {
            der--;
        }

        if ( izq <= der)
        {
            tem = registros[izq].matri;
            registros[izq].matri = registros[der].matri;
            registros[der].matri = tem;
            izq++;
            der--;
        }
```

```
    }while(izq <= der);

    if (limIzq < der)
    {
        quickSort(registros, limIzq, der);
    }

    if (limDer > izq)
    {
        quickSort(registros, izq, limDer);
    }
}
int ordenarR(Tkey registros[], int tam, int band)
{
    system("cls");
    if(tam > 0)
    {
        if(band == 0)
        {
            burbuja(registros, tam);
            band = 1;
        }
        else
        {
            quickSort(registros, 0, tam);
        }

        printf("HAS ORDENADO LOS DATOS\n");
        system("PAUSE");
        return band;
    }
    else
    {
        printf("PRIMERO DEBE HABER DATOS\n");
        system("PAUSE");
        return 0;
    }
```

```
}
```

```
int eliminarEsp(char cadena[])
{
    int tam;
    int i;

    tam = strlen(cadena);

    if(cadena[0] == '\0')
    {
        return 0;
    }

    for(i = 0; i <= tam; i++)
    {
        if(cadena[i] == ' ' && cadena[i + 1] == ' ')
        {
            return -1;
        }
    }
    return 0;
}
int soloLetras(char cadena[])
{
    int tam;
    int i;

    tam = strlen(cadena);

    if(eliminarEsp(cadena) == 0)
    {
        return 0;
    }

    for(i =  0; i <= tam; i++)
    {
        if(cadena[i] < 'A' || cadena[i] > 'z')
        {
            return -1;
        }
        if(cadena[i] > 'Z' && cadena[i] < 'a')
        {
```

```
            return -1;
        }

        return 0;
    }
    return 0;
}
void validaCad(char cadena[], const char *msg)
{
    int tam, band;

    band = 1;
    do{

        printf("%s", msg);
        fflush(stdin);
        gets(cadena);

        if (soloLetras(cadena) == 0)
        {
            if(eliminarEsp(cadena) == -1)
            {
                printf("NO SE PERMITEN ESPACIOS\n");
                continue;
            }
        }
        else
        {
            if(cadena[0] == ' ')
            {
                printf("NO SE PERMITEN ESPACIOS\n");
                continue;
            }
            else
            {
                printf("INGRESA SOLO LETRAS\n");
                continue;
```

```
            }
```

```
        }

        strupr(cadena);
        band = 0;

    }while(band == 1);
}
int existeTkey(long mat, Tkey *reg, int tam)
{
    int i;

    for(i = 0; i <= tam; i++)
    {
        if (mat == reg[i].matri)
        {
            return 1;
        }
    }
    return 0;
}
Tkey datosR(Tkey registros[], int tam)
{
    Tkey reg;
    long matri;
    char *nombre;
    char *apellido;
    int status;
    int sexo;
    int edad;

    matri = (rand() % 99999) + 300000;
    while(existeTkey(matri, registros, tam) == 1)
    {
        matri = (rand() % 99999) + 300000;
    }

    edad = (rand() % 60) + 17;
    status = rand() % 2;
    sexo = rand() % 2;

    reg.status = status;
```

```
    reg.matri = matri;
    reg.edad = edad;
    reg.genero = sexo;
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////
    char nombresM[][31] = {"Juan", "Luis", "Carlos", "Pedro"};
    char nombresF[][31] = {"Maria", "Ana", "Laura", "Sofia"};
    char apellidosP[][31] = {"Garcia", "Rodriguez", "Martínez", "Lopez",
"Perez", "Fernandez", "Gonzalez", "Hernandez"};
    char apellidosM[][31] = {"Sanchez", "Ramirez", "Torres", "Diaz",
"Vargas", "Jimenez", "Ruiz", "Silva"};
    int apellidoPIndex = rand() % 8;
    int apellidoMIndex = rand() % 8;
    if (sexo == 1)
    {
        // Genera un índice aleatorio para nombres masculinos
        int nombreIndex = rand() % 4;
        strcpy(reg.nombre, nombresM[nombreIndex]);
    }
    else
    {
        // Genera un índice aleatorio para nombres femeninos
        int nombreIndex = rand() % 4;
        strcpy(reg.nombre, nombresF[nombreIndex]);
    }

    /*strcpy(reg.nombre, nombres[nombreIndex]);*/
    strcpy(reg.apellidoP, apellidosP[apellidoPIndex]);
    strcpy(reg.apellidoM, apellidosM[apellidoMIndex]);

    return reg;
}
int agregarR(Tkey registros[], int tam)
{
    system("CLS");
    int i;
    int band = 0;
```

```
    for(i = 0; i < 100; i++)
```

```
    {
        system("CLS");
        registros[tam] = datosR(registros, tam);
        system("CLS");
        tam++;
    }
    printf("HAS GENERADO 500 REGISTROS\n");
    system("PAUSE");

    return tam;
}
Tkey datosM(Tkey registros[], int tam)
{

    Tkey reg;
    long mat;
    char apPat[21], apMat[21], nombre[41];
    do{
        reg.status = validaInt(0, 1,"0) INACTIVO 1) ACTIVO","FUERA DE
RANGO");
        mat = validaL(300000, 399999,"INGRESA TU MATRICULA: ","MATRICULA
INVALIDA");
        if(existeTkey(mat, registros, tam) == 1)
        {
            printf("ESTA MATRICULA YA EXISTE \n INGRESE UNA MATRICULA
VALIDA\n");
        }
    }while(existeTkey(mat, registros, tam) == 1);

    reg.matri = mat;

    validaCad(apPat, "INGRESA TU APELLIDO PATERNO: \n");
    strcpy(reg.apellidoP, apPat);

    validaCad(apMat, "INGRESA TU APELLIDO MATERNO: \n");
    strcpy(reg.apellidoM, apMat);

    validaCad(nombre, "INGRESA TU NOMBRE: \n");
    strcpy(reg.nombre, nombre);

    reg.edad = validaInt(17, 80,"INGRESA LA EDAD: ","EDAD FUERA DE RANGO");
```

```
    reg.genero = validaInt(1, 2,"INGRESE EL GENERO\n 1) HOMBRE\n 2)
MUJER","OPCION FUERA DE RANGO");

    return reg;
}
int agregarM(Tkey registros[], int tam)
{
    int op = 1;
    do{
        system("CLS");
        registros[tam] = datosM(registros, tam);
        system("CLS");
        tam++;
        op = validaInt(0, 1,"1.- AGREGAR DATOS \n0.- SALIR \n SELECCIONE
UNA OPCION: ","OPCION FUERA DE RANGO");
    }while(op == 1);

    return tam;
}
void imprimirR(Tkey registros[], int tam)
{
    Tkey reg;
    int i;
    char *sexo;
    system("CLS");

    if (tam > 0)
    {
        printf("%-10s %-14s %-15s %-20s %-20s %-10s %-10s\n", "ESTATUS",
"MATRICULA", "NOMBRE", "APELLIDO PATERNO", "APELLIDO MATERNO", "EDAD",
"GENERO\n");

        for (i = 0; i < tam; i++)
        {
            if(registros[i].genero == 1)
            {
                sexo ="HOMBRE";
```

```
            }
```

```
            else
            {
                sexo ="MUJER";
            }

            if (registros[i].matri != -1 && registros[i].matri >= 300000 &&
registros[i].matri <= 399999)
            {
                printf("%-10d %-14ld %-15s %-20s %-20s %-10d %-10s\n",
registros[i].status, registros[i].matri, registros[i].nombre,
registros[i].apellidoP, registros[i].apellidoM, registros[i].edad, sexo);
            }
        }
    }
    else
    {
        printf("INGRESE DATOS PARA PODER IMPRIMIR\n");
    }

    printf("\n");
    system("PAUSE");
}
void buscarR(Tkey registros[], int tam)
{
    system("CLS");
    long mat;
    int pos, id, i;
    char *sexo;
    int band = 0;

    id = validaL(300000, 399999,"INGRESA LA MATRICULA QUE QUIERES ELIMINAR:
","MATRICULA INVALIDA");
    pos = busqSec(registros, tam, id);

    if (pos != -1)
    {
        system("CLS");
        printf("\nRESULTADOS ENCONTRADO.\n%-10s %-14s %-15s %-20s %-20s %-
10s %-10s\n", "ESTATUS", "MATRICULA", "NOMBRE", "APELLIDO PATERNO",
"APELLIDO MATERNO", "EDAD", "GENERO\n");
        if(registros[pos].genero == 1)
```

```
        {
            sexo ="HOMBRE";
        }
        else
        {
            sexo ="MUJER";
        }

        if (registros[pos].matri != -1 && registros[pos].matri >= 300000 &&
registros[pos].matri <= 399999)
        {
            printf("%-10d %-14ld %-15s %-20s %-20s %-10d %-10s\n",
registros[pos].status, registros[pos].matri, registros[pos].nombre,
registros[pos].apellidoP, registros[pos].apellidoM, registros[pos].edad,
sexo);
        }
    }
    else
    {
        printf("NO SE ENCONTRO LA MATRICULA\n");
    }

}
void eliminarR(Tkey registros[], int tam)
{
    system("CLS");

    long id;
    int pos, op;

    id = validaL(300000, 399999,"INGRESA LA MATRICULA QUE QUIERES ELIMINAR:
","MATRICULA INVALIDA");
    pos = busqSec(registros, tam, id);

    if (pos != -1)
    {
        imprimirR(registros, tam);
```

```
        op = validaInt(1, 2,"1) ELIMINAR\n 2) NO ELIMINAR\n","OPCION FUERA
DE RANGO");

        if (op == 1)
        {
            registros[pos].status = 0;
        }
    }
    else
    {
        printf("LA MATRICULA NO EXISTE\n");
    }
}
void archivo(Tkey registros[],int tam)
{
    system("CLS");
    Tkey reg;
    int i;
    char *sexo;
    FILE *arch;
    arch=fopen("archivo.txt","w");
    if (tam > 0)
    {
        fprintf(arch,"%-10s %-14s %-15s %-20s %-20s %-10s %-10s\n",
"ESTATUS", "MATRICULA", "NOMBRE", "APELLIDO PATERNO", "APELLIDO MATERNO",
"EDAD", "GENERO\n");

        for (i = 0; i < tam; i++)
        {
            if(registros[i].genero == 1)
            {
                sexo ="HOMBRE";
            }
            else
            {
                sexo ="MUJER";
            }

            if (registros[i].matri != -1 && registros[i].matri >= 300000 &&
registros[i].matri <= 399999)
            {
```

```
                fprintf(arch,"%-10d %-14ld %-15s %-20s %-20s %-10d %-
10s\n", registros[i].status, registros[i].matri, registros[i].nombre,
registros[i].apellidoP, registros[i].apellidoM, registros[i].edad, sexo);
            }
        }
        printf("Se han cargado los datos en el archivo");
        fclose(arch);
    }
    else
    {
        printf("INGRESE DATOS PARA PODER IMPRIMIR\n");
    }

    printf("\n");
    system("PAUSE");
}
int cargartxt(Tkey vect[],int *tam)
{
    int i;
    Tkey reg;
    char basura[30];
    FILE *fa;
    i=(*tam);
    fa= fopen("datos.txt","r");
    if (fa)
    {
        while (!feof(fa))
        {
            fscanf(fa,"%s%ld%s%s%s%d%s\n",basura,reg.matri,reg.nombre,reg.a
pellidoP,reg.apellidoM,reg.edad,reg.genero);
            vect[i++]=reg;
        }
        printf("Se cargo correctamente el archivo");
        fclose(fa);
    }
    (*tam)=i;
    return 0;
```

```
}
```

```
PS C:\Users\marti\OneDrive\Documentos\ESTRUCTURADA 2023 1> .\NActividad110
        MENU
  1-Cargar
  2-Eliminar
  3-Buscar
  4-Ordenar
  5-Imprimir
  6-Archivo Texto
  0-Salir
  Escoge una de la opcion
```

```
HAS GENERADO 500 REGISTROS
Press any key to continue . . .
```

```
 0) INACTIVO 1) ACTIVO
 1
 INGRESA TU MATRICULA:
 369705
 INGRESA TU APELLIDO PATERNO:
 Santos
 INGRESA TU APELLIDO MATERNO:
 Tirado
 INGRESA TU NOMBRE:
 Martin
 INGRESA LA EDAD:
 21
 INGRESE EL GENERO
  1) HOMBRE
  2) MUJER
```

```
Press any key to continue . . .
HAS ORDENADO LOS DATOS
Press any key to continue . . .
```

```
1       326129      Sofia       Rodriguez       Jimenez         57      MUJER
0       326302      Maria       Rodriguez       Diaz            48      MUJER
0       326740      Luis        Hernandez       Sanchez         41      HOMBRE
0       326924      Luis        Rodriguez       Jimenez         69      HOMBRE
0       327446      Carlos      Mart├¡nez       Ruiz            62      HOMBRE
0       328321      Maria       Rodriguez       Sanchez         75      MUJER
0       328433      Laura       Garcia          Ramirez         66      MUJER
1       329337      Luis        Garcia          Ruiz            68      HOMBRE
0       329358      Pedro       Rodriguez       Ramirez         39      HOMBRE
1       329510      Laura       Rodriguez       Torres          22      MUJER
1       330695      Laura       Garcia          Jimenez         46      MUJER
1       331003      Juan        Fernandez       Silva           29      HOMBRE
0       331101      Carlos      Hernandez       Vargas          50      HOMBRE
1       331111      Sofia       Garcia          Ruiz            41      MUJER
0       331316      Luis        Rodriguez       Diaz            48      HOMBRE
1       331322      Ana         Fernandez       Silva           50      MUJER
0       331329      Luis        Lopez           Torres          45      HOMBRE
1       331934      Ana         Gonzalez        Silva           42      MUJER
1       332439      Carlos      Mart├¡nez       Ruiz            43      HOMBRE
1       332678      Ana         Rodriguez       Sanchez         29      MUJER
1       369705      MARTIN      SANTOS          TIRADO          21      HOMBRE
```

```
Se han cargado los datos en el archivo
Press any key to continue . . .
```