Universidad Autónoma de Baja California

Facultad de Ingeniería, Arquitectura y Diseño

Programación Estructurada

Prof. Pedro Núñez Yepiz

Aarón Alejandro Parra Velarde

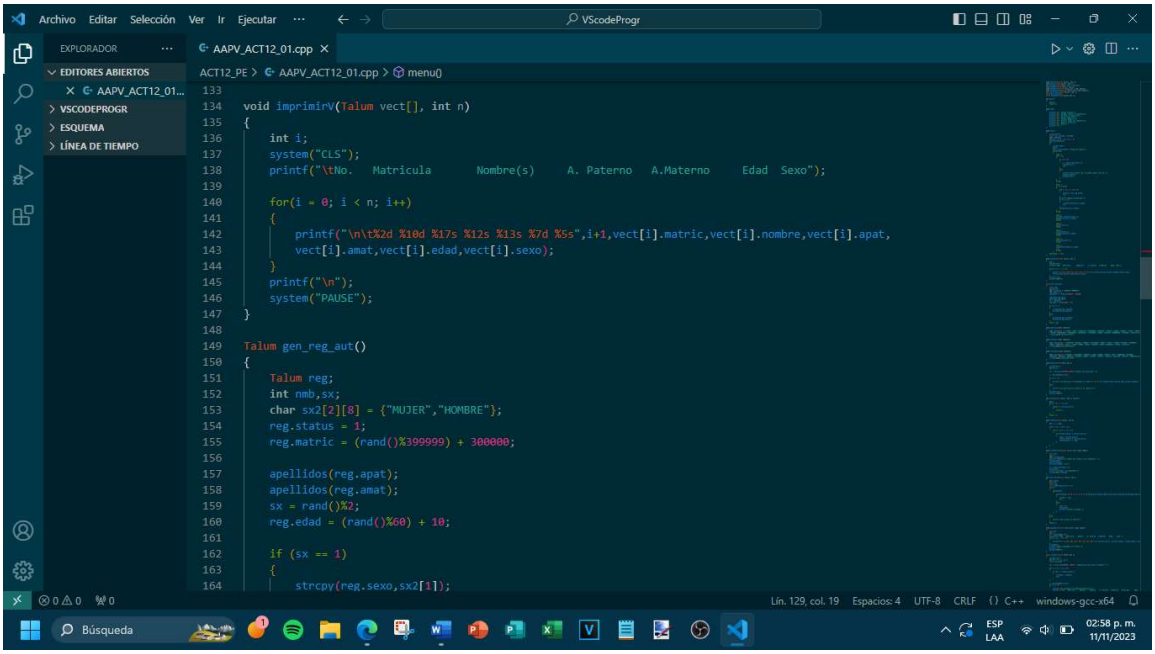
Actividad 12 - Anexos

Grupo: 432

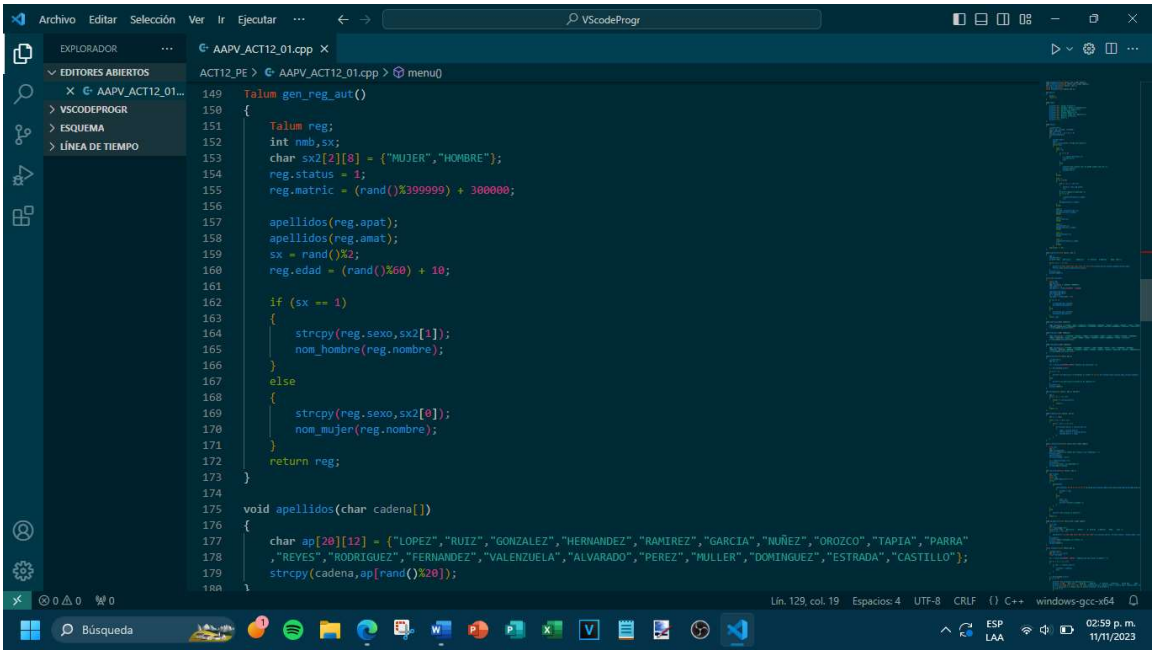
Matrícula: 372193

Ensenada, B.C., 11 de Noviembre de 2023

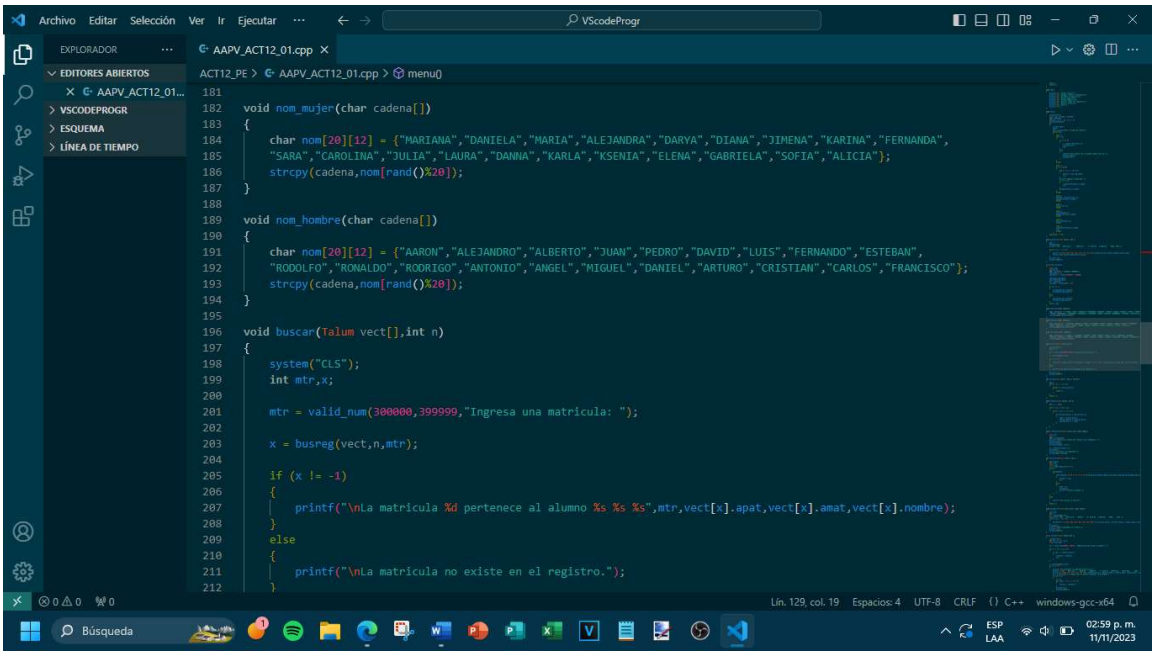
```c
/*****************************************************************************/
/* Parra Velarde Aaron Alejandro        372193                               */
/* 09/Nov/2023                                                               */
/* Programa que despliega un menu de estructuras y archivos.                 */
/* PVAA_A12_A32_01                                                           */
/*****************************************************************************/

#include <string.h>
#include "arzat.h"
#include <time.h>
#define N 1500

typedef struct _alum
{
    int status;
    int matric;
    char nombre[30];
    char apat[30];
    char amat[30];
    int edad;
    char sexo[10];
}Talum;

void menu();
void msg();
void apellidos(char cadena[]);
void nom_mujer(char cadena[]);
void nom_hombre(char cadena[]);
void imprimirV(Talum vect[], int n);
void buscar(Talum vect[], int n);
int busrug(Talum vect[], int n, int mCr);
void ordenar(Talum vect[], int m);
void agregartxt(Talum vect[],int n,char name[]);
void createtxtfile(Talum vect[],int n,char name[]);
int lee_archivo(Talum vect[], int i);
Talum gen_reg_aut();
Talum eliminar(Talum vekt[],int n);

int main()
{
    menu();
    return 0;
}

void msg()
{
    printf("\n1.- Cargar Archivo");
    printf("\n2.- Agregar registro automatico");
    printf("\n3.- Eliminar un Registro");
    printf("\n4.- Buscar Registro");
    printf("\n5.- Ordenar Registro");
    printf("\n6.- Mostrar todos los registros");
    printf("\n7.- Generar Archivo");
    printf("\n0.- Salir");
    printf("\n");
}

void menu()
{
    system("CLS");
    Talum reg, vect[N], vect2[N];
    char name[40];
    int opc, i = 0, j,k = 0,z = 0;
    srand(time(NULL));
    do
    {
        system("CLS");
        msg();
        opc = valid_num(0,7,"Elige una opcion");
        switch(opc)
        {
            case 1:
            if (k == 0)
            {
                i = lee_archivo(vect,i);
                if (z == 0)
                {
                    createtxtfile(vect,i,name);
                    z++;
                }
                system("cls");
                k++;
            }
            else
            {
                printf("\nEl archivo solo se puede cargar una vez.");
                system("pause");
                system("cls");
            }
            break;

            case 2:
            if (i < N-10)
            {
                for(j = 0; j < 10; j++)
                {
                    vect[i] = gen_reg_aut();
                    i++;
                }
                printf("\nRegistro generado.");
                if (z == 0)
                {
                    createtxtfile(vect,i,name);
                    z++;
                }
                agregartxt(vect,i,name);
            }
            break;

            case 3:
            vect[N] = eliminar(vect,i);
            agregartxt(vect,i,name);
            break;

            case 4:
            buscar(vect,i);
            break;

            case 5:
            ordenar(vect,i);
            agregartxt(vect,i,name);
            break;

            case 6:
            imprimirV(vect,i);
            break;

            case 7:
            createtxtfile(vect,i,name);
            z++;
            break;
        }
    }while(opc != 0);
}
```



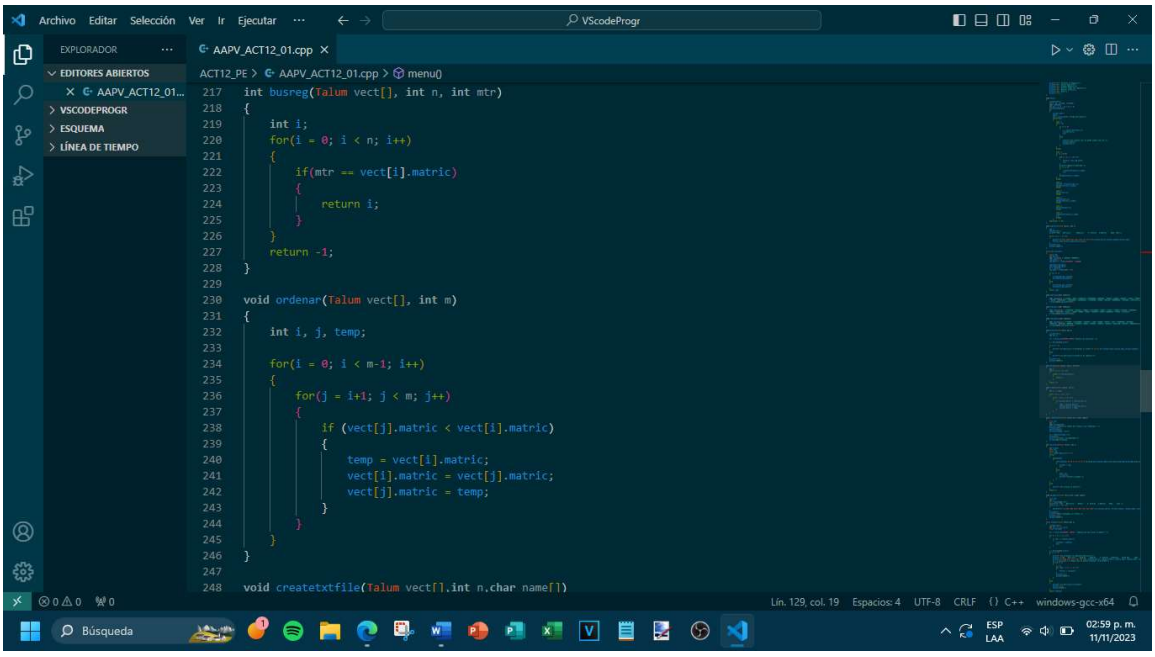
```
void imprimirV(Talum vect[], int n)
{
    int i;
    system("CLS");
    printf("\tNo.   Matricula       Nombre(s)     A. Paterno   A.Materno     Edad  Sexo");

    for(i = 0; i < n; i++)
    {
        printf("\n\t%2d %10d %17s %12s %13s %7d %5s",i+1,vect[i].matric,vect[i].nombre,vect[i].apat,
        vect[i].amat,vect[i].edad,vect[i].sexo);
    }
    printf("\n");
    system("PAUSE");
}

Talum gen_reg_aut()
{
    Talum reg;
    int nmb,sx;
    char sx2[2][8] = {"MUJER","HOMBRE"};
    reg.status = 1;
    reg.matric = (rand()%399999) + 300000;

    apellidos(reg.apat);
    apellidos(reg.amat);
    sx = rand()%2;
    reg.edad = (rand()%60) + 10;

    if (sx == 1)
    {
        strcpy(reg.sexo,sx2[1]);
```



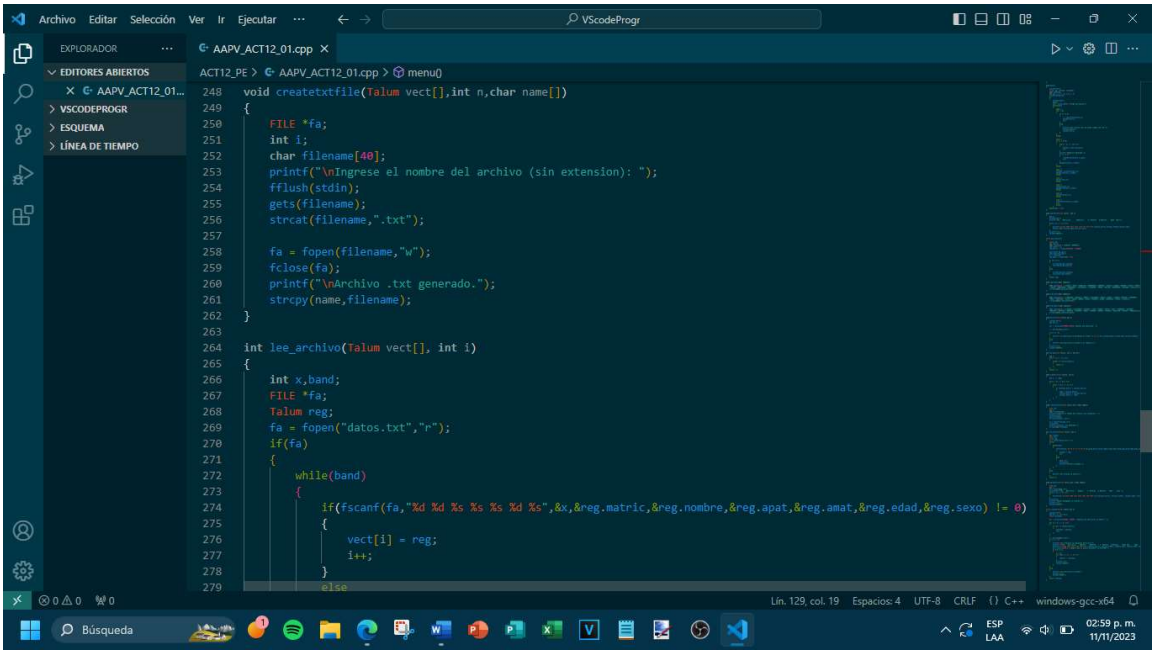
```
Talum gen_reg_aut()
{
    Talum reg;
    int nmb,sx;
    char sx2[2][8] = {"MUJER","HOMBRE"};
    reg.status = 1;
    reg.matric = (rand()%399999) + 300000;

    apellidos(reg.apat);
    apellidos(reg.amat);
    sx = rand()%2;
    reg.edad = (rand()%60) + 10;

    if (sx == 1)
    {
        strcpy(reg.sexo,sx2[1]);
        nom_hombre(reg.nombre);
    }
    else
    {
        strcpy(reg.sexo,sx2[0]);
        nom_mujer(reg.nombre);
    }
    return reg;
}

void apellidos(char cadena[])
{
    char ap[20][12] = {"LOPEZ","RUIZ","GONZALEZ","HERNANDEZ","RAMIREZ","GARCIA","NUÑEZ","OROZCO","TAPIA","PARRA"
    ,"REYES","RODRIGUEZ","FERNANDEZ","VALENZUELA","ALVARADO","PEREZ","MULLER","DOMINGUEZ","ESTRADA","CASTILLO"};
    strcpy(cadena,ap[rand()%20]);
}
```



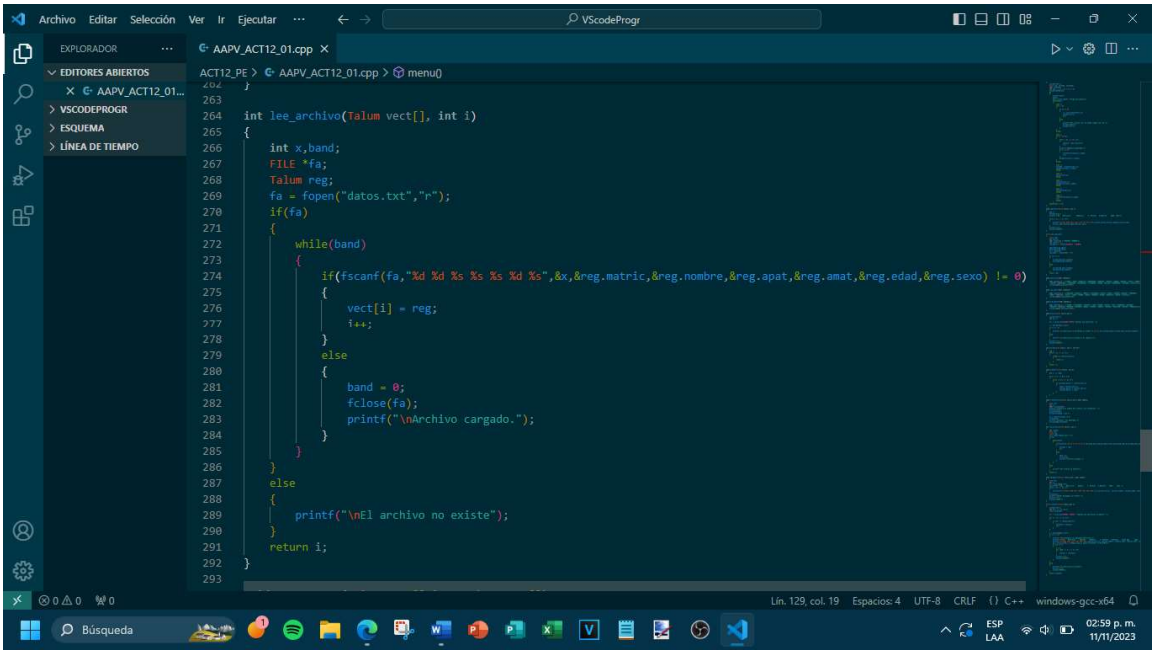
```
181
182 void nom_mujer(char cadena[])
183 {
184     char nom[20][12] = {"MARIANA","DANIELA","MARIA","ALEJANDRA","DARYA","DIANA","JIMENA","KARINA","FERNANDA",
185     "SARA","CAROLINA","JULIA","LAURA","DANNA","KARLA","KSENIA","ELENA","GABRIELA","SOFIA","ALICIA"};
186     strcpy(cadena,nom[rand()%20]);
187 }
188
189 void nom_hombre(char cadena[])
190 {
191     char nom[20][12] = {"AARON","ALEJANDRO","ALBERTO","JUAN","PEDRO","DAVID","LUIS","FERNANDO","ESTEBAN",
192     "RODOLFO","RONALDO","RODRIGO","ANTONIO","ANGEL","MIGUEL","DANIEL","ARTURO","CRISTIAN","CARLOS","FRANCISCO"};
193     strcpy(cadena,nom[rand()%20]);
194 }
195
196 void buscar(Talum vect[],int n)
197 {
198     system("CLS");
199     int mtr,x;
200
201     mtr = valid_num(300000,399999,"Ingresa una matricula: ");
202
203     x = busreg(vect,n,mtr);
204
205     if (x != -1)
206     {
207         printf("\nLa matricula %d pertenece al alumno %s %s %s",mtr,vect[x].apat,vect[x].amat,vect[x].nombre);
208     }
209     else
210     {
211         printf("\nLa matricula no existe en el registro.");
212     }
```



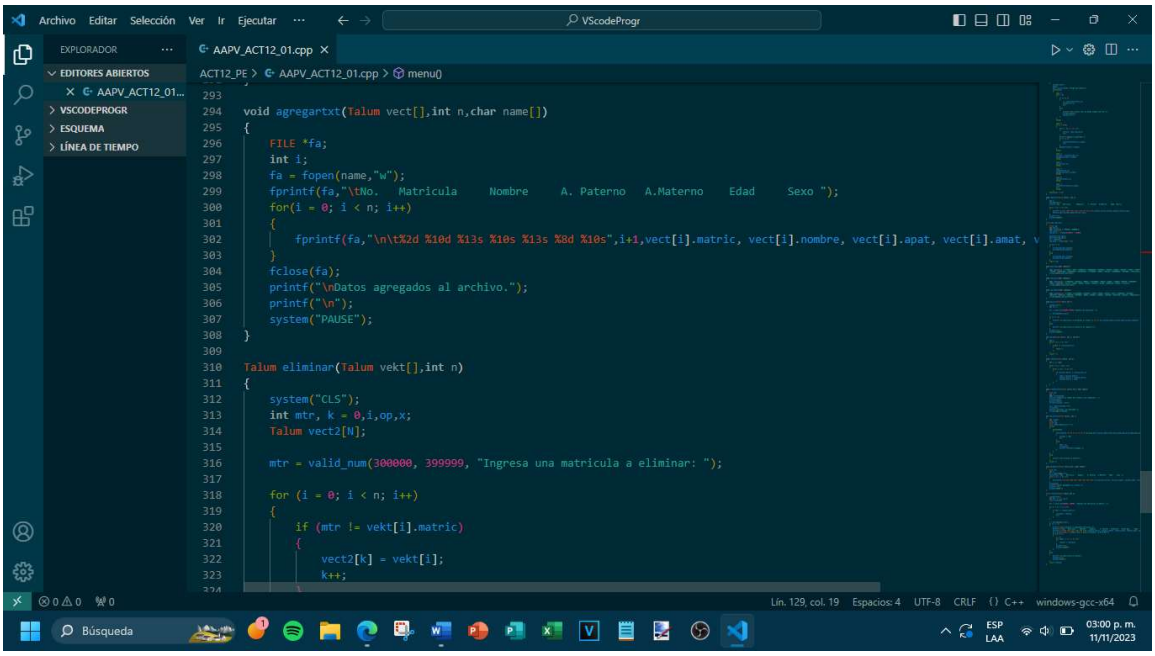
```
217 int busreg(Talum vect[], int n, int mtr)
218 {
219     int i;
220     for(i = 0; i < n; i++)
221     {
222         if(mtr == vect[i].matric)
223         {
224             return i;
225         }
226     }
227     return -1;
228 }
229
230 void ordenar(Talum vect[], int m)
231 {
232     int i, j, temp;
233
234     for(i = 0; i < m-1; i++)
235     {
236         for(j = i+1; j < m; j++)
237         {
238             if (vect[j].matric < vect[i].matric)
239             {
240                 temp = vect[i].matric;
241                 vect[i].matric = vect[j].matric;
242                 vect[j].matric = temp;
243             }
244         }
245     }
246 }
247
248 void createtxtfile(Talum vect[],int n,char name[])
```



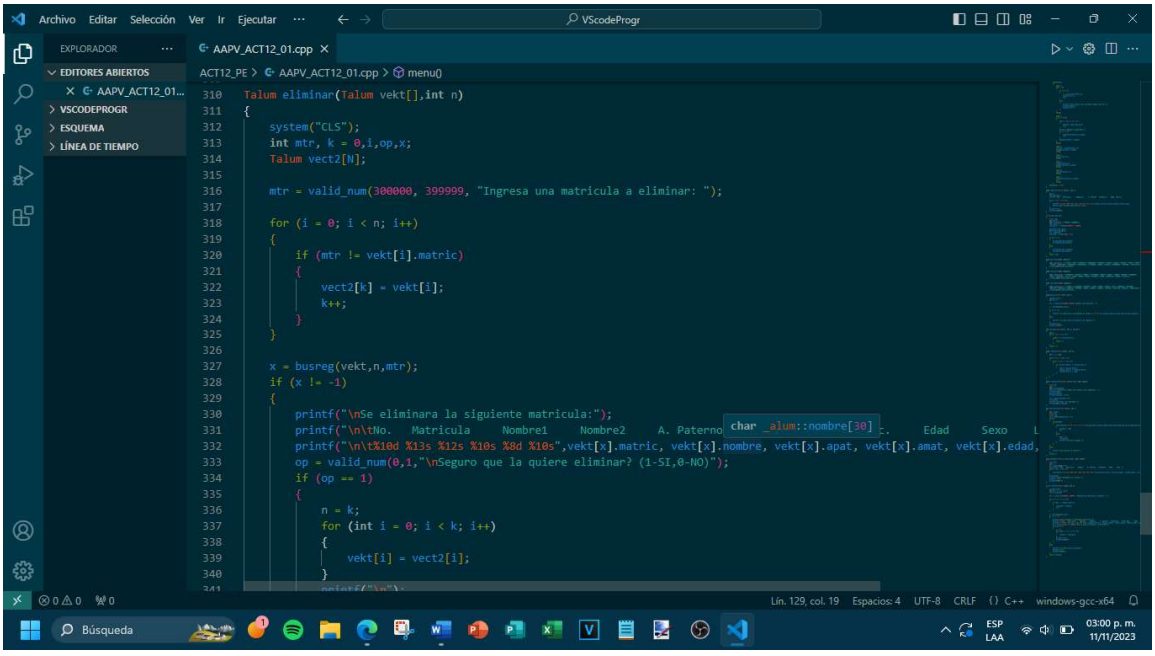
```
void createtxtfile(Talum vect[],int n,char name[])
{
    FILE *fa;
    int i;
    char filename[40];
    printf("\nIngrese el nombre del archivo (sin extension): ");
    fflush(stdin);
    gets(filename);
    strcat(filename,".txt");

    fa = fopen(filename,"w");
    fclose(fa);
    printf("\nArchivo .txt generado.");
    strcpy(name,filename);
}

int lee_archivo(Talum vect[], int i)
{
    int x,band;
    FILE *fa;
    Talum reg;
    fa = fopen("datos.txt","r");
    if(fa)
    {
        while(band)
        {
            if(fscanf(fa,"%d %d %s %s %s %d %s",&x,&reg.matric,&reg.nombre,&reg.apat,&reg.amat,&reg.edad,&reg.sexo) != 0)
            {
                vect[i] = reg;
                i++;
            }
            else
```



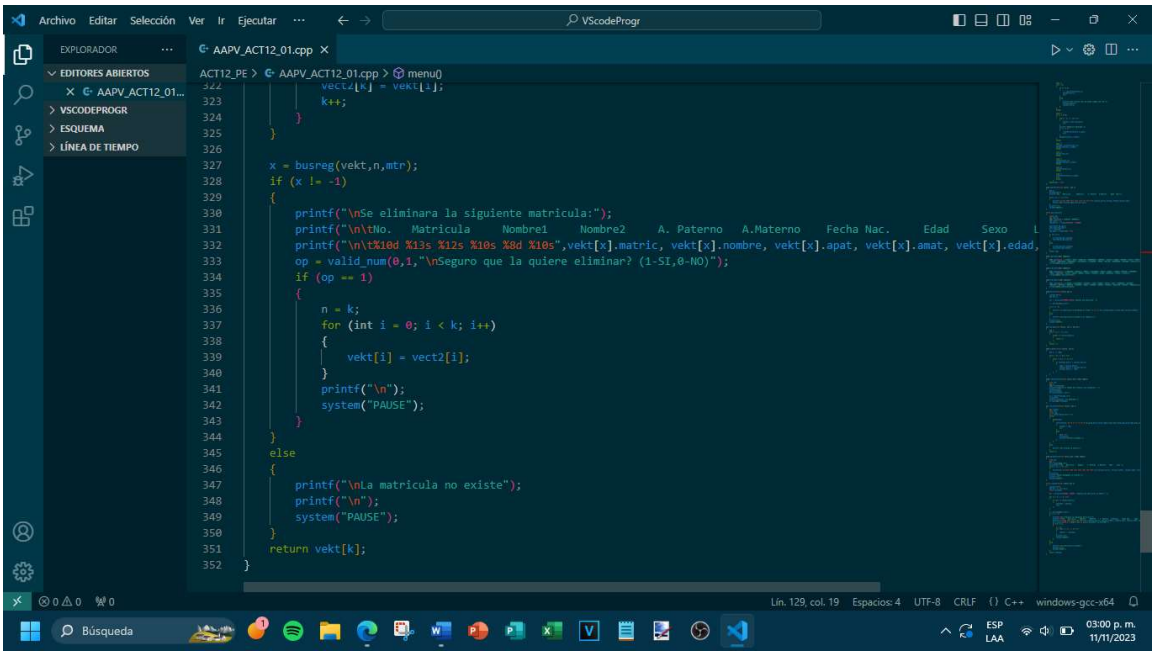
```
}

int lee_archivo(Talum vect[], int i)
{
    int x,band;
    FILE *fa;
    Talum reg;
    fa = fopen("datos.txt","r");
    if(fa)
    {
        while(band)
        {
            if(fscanf(fa,"%d %d %s %s %s %d %s",&x,&reg.matric,&reg.nombre,&reg.apat,&reg.amat,&reg.edad,&reg.sexo) != 0)
            {
                vect[i] = reg;
                i++;
            }
            else
            {
                band = 0;
                fclose(fa);
                printf("\nArchivo cargado.");
            }
        }
    }
    else
    {
        printf("\nEl archivo no existe");
    }
    return i;
}
```



```
293
294 void agregartxt(Talum vect[],int n,char name[])
295 {
296     FILE *fa;
297     int i;
298     fa = fopen(name,"w");
299     fprintf(fa,"\tNo.   Matricula     Nombre     A. Paterno   A.Materno    Edad     Sexo ");
300     for(i = 0; i < n; i++)
301     {
302         fprintf(fa,"\n\t%2d %10d %13s %10s %13s %8d %10s",i+1,vect[i].matric, vect[i].nombre, vect[i].apat, vect[i].amat, v
303     }
304     fclose(fa);
305     printf("\nDatos agregados al archivo.");
306     printf("\n");
307     system("PAUSE");
308 }
309
310 Talum eliminar(Talum vekt[],int n)
311 {
312     system("CLS");
313     int mtr, k = 0,i,op,x;
314     Talum vect2[N];
315
316     mtr = valid_num(300000, 399999, "Ingresa una matricula a eliminar: ");
317
318     for (i = 0; i < n; i++)
319     {
320         if (mtr != vekt[i].matric)
321         {
322             vect2[k] = vekt[i];
323             k++;
```



```
310 Talum eliminar(Talum vekt[],int n)
311 {
312     system("CLS");
313     int mtr, k = 0,i,op,x;
314     Talum vect2[N];
315
316     mtr = valid_num(300000, 399999, "Ingresa una matricula a eliminar: ");
317
318     for (i = 0; i < n; i++)
319     {
320         if (mtr != vekt[i].matric)
321         {
322             vect2[k] = vekt[i];
323             k++;
324         }
325     }
326
327     x = busreg(vekt,n,mtr);
328     if (x != -1)
329     {
330         printf("\nSe eliminara la siguiente matricula:");
331         printf("\n\tNo.   Matricula     Nombre1     Nombre2     A. Paterno   char _alum::nombre[30]     Edad     Sexo
332         printf("\n\t%10d %13s %12s %10s %8d %10s",vekt[x].matric, vekt[x].nombre, vekt[x].apat, vekt[x].amat, vekt[x].edad,
333         op = valid_num(0,1,"\nSeguro que la quiere eliminar? (1-SI,0-NO)");
334         if (op == 1)
335         {
336             n = k;
337             for (int i = 0; i < k; i++)
338             {
339                 vekt[i] = vect2[i];
340             }
```


Archivo Editar Selección Ver Ir Ejecutar
VScodeProgr
EXPLORADOR
EDITORES ABIERTOS
AAPV_ACT12_01...
VSCODEPROGR
ESQUEMA
LÍNEA DE TIEMPO
AAPV_ACT12_01.cpp
ACT12_PE > AAPV_ACT12_01.cpp > menu()
vect2[k] = vekt[i];
k++;
}
}
x = busreg(vekt,n,mtr);
if (x != -1)
{
printf("\nSe eliminara la siguiente matricula:");
printf("\n\tNo. Matricula Nombre1 Nombre2 A. Paterno A.Materno Fecha Nac. Edad Sexo L
printf("\n\t%10d %13s %12s %10s %8d %10s",vekt[x].matric, vekt[x].nombre, vekt[x].apat, vekt[x].amat, vekt[x].edad,
op = valid_num(0,1,"\nSeguro que la quiere eliminar? (1-SI,0-NO)");
if (op == 1)
{
n = k;
for (int i = 0; i < k; i++)
{
vekt[i] = vect2[i];
}
printf("\n");
system("PAUSE");
}
}
else
{
printf("\nLa matricula no existe");
printf("\n");
system("PAUSE");
}
return vekt[k];
}
Lín. 129, col. 19 Espacios: 4 UTF-8 CRLF C++ windows-gcc-x64
Búsqueda
ESP
LAA
03:00 p. m.
11/11/2023