



PROYECTO FINAL BASE DE DATOS DE UNIVERSIDAD

INTEGRANTES DEL GRUPO:

- ❑ Carpio Peña, Josue
- ❑ León Zarate, Ariana
- ❑ Rodriguez Pinto, Anthony

Docente: DSc. Manuel Eduardo Loaiza
Fernández

Departamento de de ciencia de la computación
Universidad católica San Pablo
Semestre 2021-II
Arequipa-Perú

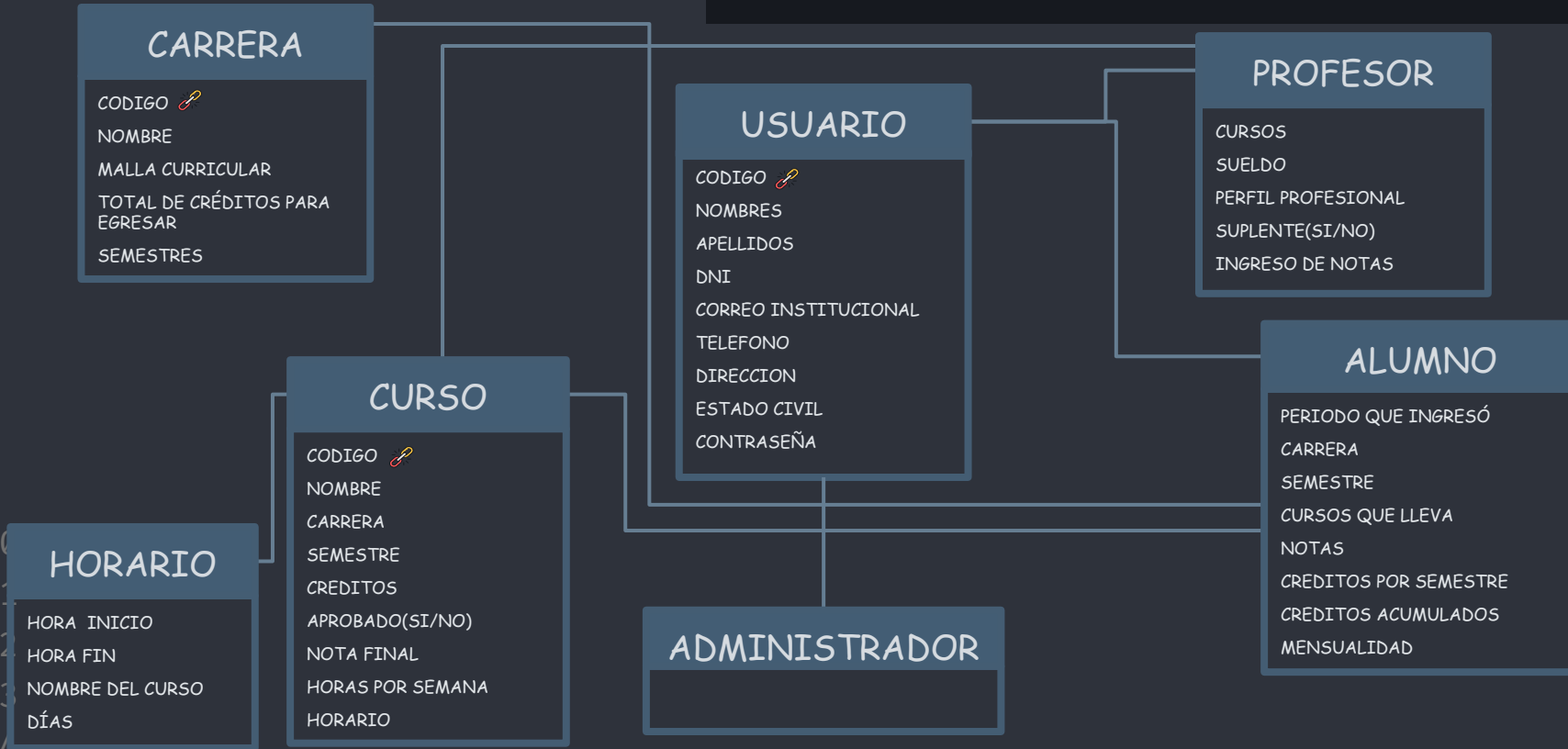
PASOS



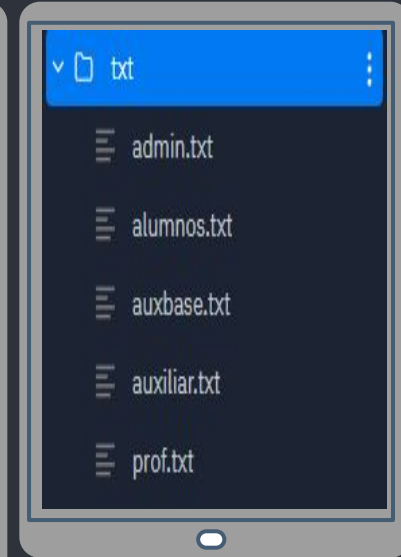
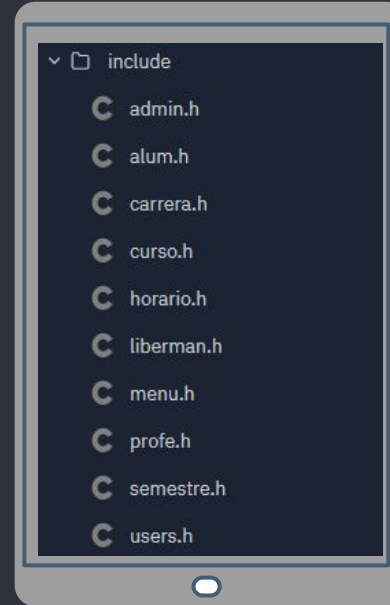
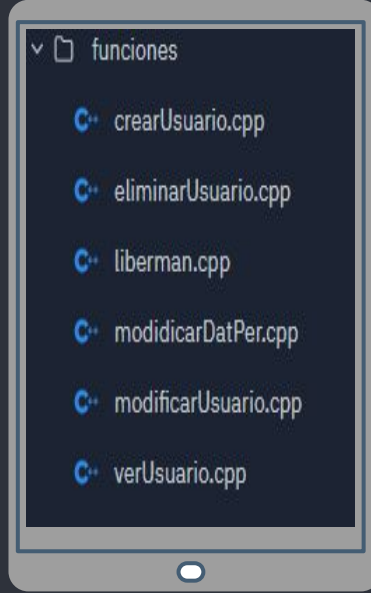
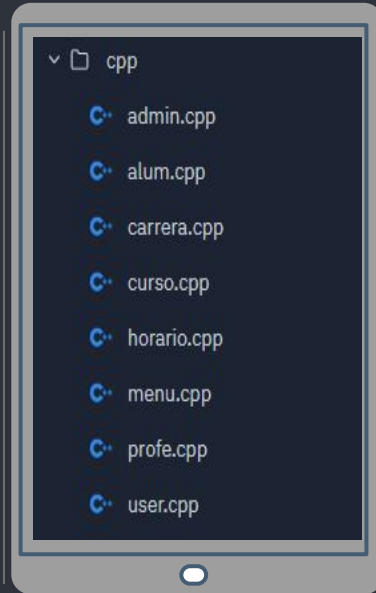
PROYECTO FINAL: BASE DE DATOS

Estructura

1
2
3
4
5
6
7
8
9
10
11
12
13
14



Archivos{



}

CALENDARIO	SEMANA 3		
	JOSUE	ANTHONY	ARIANA
CLASE CARRERA	X		
CLASE CURSO		X	
CLASE HORARIO			X
MÉTODO CONSTRUCTOR CARRERA	X		
MÉTODO CONSTRUCTOR HORARIO			X
MÉTODO CONSTRUCTOR CURSO		X	
ESTÉTICA	X	X	X

Main.cpp {

```
//Realizado en Replit
/*Grupo: Las chicas super poderosas          CIENCIA DE LA COMPUTACIÓN I  CCOMP2-1
INTEGRANTES:
- Josue Carpio
- Ariana Leon
- Anthony Rodriguez */

#include "liberman.h"

int main()
{
    Menu inicio;
    inicio.menuPrincipal();
    return 0;
}
```

}

Class Usuario {

```
#ifndef USUARIO_H
#define USUARIO_H

#define NUMBER_USUARIO_PROPERTIES 9

class Usuario
{
protected:
    string codigo;
    string contrasena;
    string nombres;
    string apellidos;
    long dni;
    string correoInsti;
    long celular;
    string direccion;
    string estdCiv;

    vector<Usuario *> users_;

public:
    Usuario();
    ~Usuario();

    Usuario *getByCodigo();
    vector<Usuario *> getAll();
    void findByName(std::string name_);
}
```

```
30 //getter
31 string getCodigo(){return this->codigo;}
32 string getContrasena(){return this->contrasena;}
33 string getNombres(){return this->nombres;}
34 string getApellidos(){return this->apellidos;}
35 string getCorreoInsti(){return this->correoInsti;}
36 long getDNI(){return this->dni;}
37 long getCelular(){return this->celular;}
38 string getDireccion(){return this->direccion;}
39 string getEstadoCiv(){return this->estdCiv;}
```

```
41 //setter
42 void setCodigo(string value_){this->codigo = value_;}
43 void setContrasena(string value_){this->contrasena = value_;}
44 void setNombres(string value_){this->nombres = value_;}
45 void setApellidos(string value_){this->apellidos = value_;}
46 void setDNI(long value_){this->dni = value_;}
47 void setCorreoInsti(string value_){this->correoInsti = value_;}
48 void setCelular(long value_){this->celular = value_;}
49 void setDireccion(string value_){this->direccion = value_;}
50 void setEstadoCiv(string value_){this->estdCiv = value_;}
```

Class Administrador {

```
#ifndef ADMIN_H_
#define ADMIN_H_
#define NUMBER_ADMIN_PROPERTIES 9

class Admin : public Usuario
{
protected:
    vector<Admin *> admin_;
public:
    Admin();
    ~Admin();

    //void menuAdmin();
    vector<Admin *> getAll();

    //MEMORIA DINAMICA

    void SaveAll();
    void DeleteObjeto(int );
    void AddObjeto(Admin * admin_);
    Admin* findByCodigo(std::string );
    Admin* findByContraseña(std::string );
    Admin* findByNombre(std::string );
    Admin* findByApellido(std::string );
    Admin* findByDNI(long );
    Admin* findByCelular(long );
    Admin* findByCorreoInsti(std::string );
    Admin* findByDireccion(std::string );
    Admin* findByEstadCiv(std::string );
    void readDatabase();
};
#endif
```

```
Admin :: Admin() :Usuario()
{
    this->readDatabase();//Usuario();
}
```


Class Profesor {

```
#ifndef PROFESOR_H_
#define PROFESOR_H_
#define NUMBER_PROFESOR_PROPERTIES 11

class Profesor : public Usuario
{
private:
    string  cursosEnsenar;
    string  sueldo;
    string  perfilProf;
    string  suplente;
    vector<Profesor *> profe__;

public:
    Profesor();
    void menuProf();
    void modificarNotas();
    void mostrarRegistroPerP(string codigo);
    void modificarDatosPerP();
    void mostrarDatosPerP();
    void verUsuario();
    vector<Profesor *> getAll();
}
```

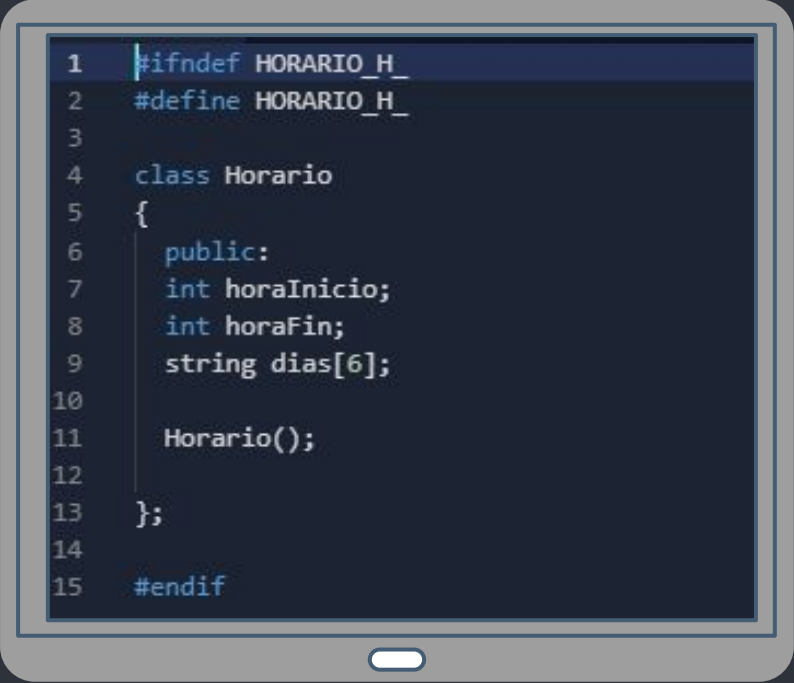
```
//MEMORIA DINAMICA
void SaveAll();
void DeleteObjeto(int value_);
void AddObjeto(Profesor * profe_);
Profesor *Profesor::findByContraseña(std::string
contraseña_);
Profesor *Profesor::findByCodigo(std::string codigo_);
Profesor *Profesor::findByNombre(std::string nombre_);
Profesor * Profesor::findBysueldo(std::string sueldo_);
Profesor *Profesor::findByperfilProf(std::string
perfilProf_);
Profesor *Profesor::findBysuplente(std::string suplente_);
Profesor *Profesor::findByApellido(std::string apellidos_);
Profesor *Profesor::findByDNI(long dni_);
Profesor *Profesor::findByDireccion(std::string direccion_);
Profesor *Profesor::findByCelular(long celular_);
Profesor *Profesor::findByEstadCiv(std::string estadCiv_);
//void getAll();
void readDatabase();
```

Class Profesor {

```
43 //getter
44 string getCursoEnsenas(){return this->cursosEnsenas;}
45 string getSueldo(){return this->sueldo;}
46 string getPerfilProf(){return this->perfilProf;}
47 string getSuplente(){return this->suplente;}
48
49 //setter
50 void setCursoEnsenas(string value_){this->cursosEnsenas =
51 value_;}
51 void setSueldo(string value_){this->sueldo = value_; }
52 void setPerfilProf(string value_){this->perfilProf = value_;
53 }
53 void setSuplente(string value_){this->suplente = value_;}
54 };
55
56 #endif
```

}

1 **Class** Horario{

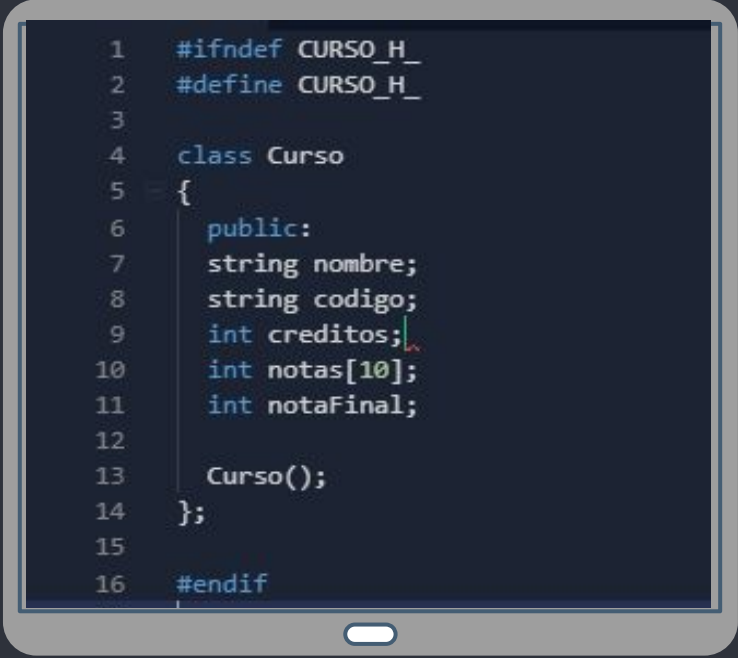


```
1 #ifndef HORARIO_H_
2 #define HORARIO_H_
3
4 class Horario
5 {
6     public:
7     int horaInicio;
8     int horaFin;
9     string dias[6];
10
11     Horario();
12
13 };
14
15 #endif
```

14 }

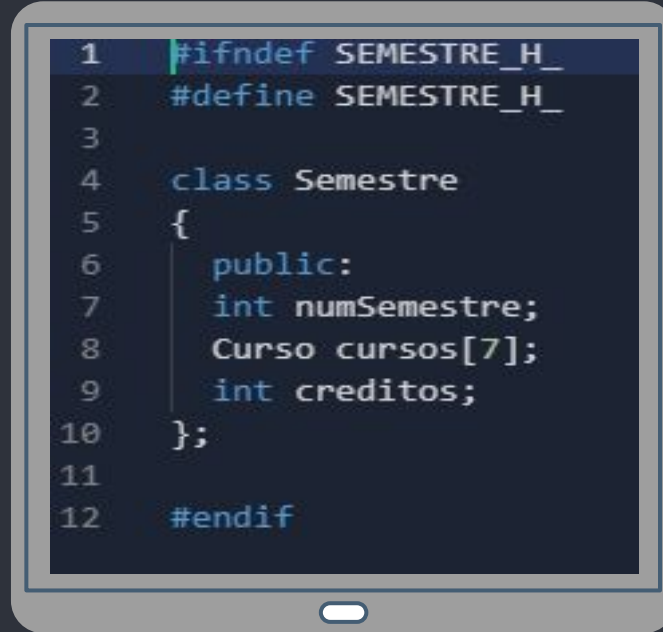
1 **Class** Curso{

14 }



```
1  #ifndef CURSO_H_
2  #define CURSO_H_
3
4  class Curso
5  {
6  public:
7      string nombre;
8      string codigo;
9      int creditos;
10     int notas[10];
11     int notaFinal;
12
13     Curso();
14 };
15
16 #endif
```

```
1  Class Semestre{  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14 }
```



Class Carrera{

```
1  #ifndef CARRERA_H_
2  #define CARRERA_H_
3
4  class Carrera
5  {
6      public:
7          string nombre;
8          string codigo;
9          int creditosEgresar;
10         //Semestre semestres[12];
11         int numSemestres;
12
13         Carrera(string, string, int,int);
14
15     };
16
17 #endif
```

```
3  Carrera :: Carrera(string nombre, string codigo, int creditosEgresar, int
numSemestres)
4  {
5      this->nombre = nombre;
6      this->codigo = codigo;
7      this->creditosEgresar = creditosEgresar;
8      //numSemestres[12]=" ";
9      this->numSemestres = numSemestres;
10 }
```

Class Alumno {

```
1  #ifndef ALUM_H_
2  #define ALUM_H_
3  #define NUMBER_ALUMNO_PROPERTIES 16
4
5  #include "curso.h"
6
7  class Alumno : public Usuario
8  {
9      private:
10         string carrera;
11         int numCursos;
12         //Curso *cursos = new Curso[7];
13         string periodoIngreso;
14         int semestre;
15         int creditosSemestre;
16         int creditosAcumulados;
17         float mensualidad;
18         vector<Alumno *> alumno__;
```

```
14 }
```

```
20 public:
21     Alumno();
22     vector<Alumno *> getAll();
23     void menuAlumno();
24     void crearAlumno();
25     //MEMORIA DINAMICA
26     void SaveAll();
27     void DeleteObjeto(int value_);
28     void AddObjeto(Alumno * alumno_);
29     //BUSCADOR EN LA LISTA
30     Alumno* findByCodigo(std::string );
31     Alumno* findByContrasena(std::string );
32     Alumno* findByNombre(std::string );
33     Alumno* findByApellido(std::string );
34     Alumno* findByDNI(long dni_);
35     Alumno* findByCelular(long );
```

Class Alumno {

```
Alumno* findByDireccion(std::string );
Alumno* findByCorreoInsti(std::string );
Alumno* findByEstadCiv(std::string );
Alumno* findByCarrera(std::string );
Alumno* findByNumCursos(int );
Alumno* findByPeriodoIngreso(std::string);
Alumno* findBySemestre(int );
Alumno* findByCreditosSemestre(int );
Alumno* findByCreditosAcumulados(int );
Alumno* findByMensualidad(float );

//void getAll();
void readDatabase();
```

```
//getter
string getCarrera(){return this->carrera;}
int getCursos(){return this->numCursos;}
string getPeriodoIngreso(){return this->periodoIngreso;}
int getSemestre(){return this->semestre; }
int getCreditosSemestre(){return this->creditosSemestre;}
int getCreditosAcumulados(){return this->creditosAcumulados;
}
float getMensualidad(){ return this->mensualidad;}

//Setters
void setCarrera(string value_){this->carrera = value_;}
void setCursos(int value_){this->numCursos = value_;}
void setPeriodoIngreso(int value_){this->periodoIngreso =
value_;}
void setSemestre(int value_){this->semestre = value_;}
void setCreditosSemestre(int value_){this->creditosSemestre
= value_;}
void setCreditosAcumulados(int value_)
{this->creditosAcumulados = value_;}
void setMensualidad(float value_){this->mensualidad =
value_;}
};
```


Menu Principal {

```
817 void Menu :: menuPrincipal()
818 {
819     int opc;
820     do
821     {
822         cout << "\t\t***Universidad las chicas superpoderosas***\t\t" << endl;
823         cout << "[1] Administrador" << endl;
824         cout << "[2] Profesor" << endl;
825         cout << "[3] Alumno" << endl;
826         cout << "[4] Salir" << endl;
827         cout << "Elija una opción..." << endl;
828         cin >> opc;
829         std::cout << "\033[H\033[2J\033[3J";
830         switch (opc)
831         {
832             case 1:
833             {
834                 menuAdmin();
835                 break;
836             }
```

```
        case 2:
        {
            menuProf();
            break;
        }
        case 3:
        {
            menuAlumno();
            break;
        }
        case 4:
        {
            break;
        }
        default:
        {
            cout << "Opción no válida" << endl;
            break;
        }
    }
}
while(opc != 4 );
}
```

Menú Profesor {

```
722 void Menu :: menuProf()
723 {
724     int opc;
725     do
726     {
727         cout << "\t\t***PROFESOR***\t\t" << endl;
728         cout << "[1] Modificar datos personales" << endl;
729         cout << "[2] Modificar notas" << endl;
730         cout << "[3] Mostrar Usuario" << endl;
731         cout << "[4] Mostrar datos Propios" << endl;
732         cout << "[5] Salir" << endl;
733         cout << "Elija una opción..." << endl;
734         cin >> opc;
735         std::cout << "\033[H\033[2J\033[3J";
736         switch (opc)
737         {
738             case 1:
739             {
740                 //modificarDatosPer();
741                 break;
742             }
743         }
```

```
743     case 2:
744     {
745         //modificarNotas();
746         break;
747     }
748     case 3:
749     {
750         //mostrarUsuario();
751         break;
752     }
753     case 4:
754     {
755         //mostrarProfesor();
756     }
757     case 5:
758     {
759         break;
760     }
761     default:
762     {
763         cout << "Opción no válida" << endl;
764         break;
765     }
766 }
767 }
768 while(opc != 5 );
769 }
770 }
```

Menú Alumno {

```
772 void Menu :: menuAlumno()
773 {
774     int opc;
775     do
776     {
777         cout << "\t\t***ALUMNO***\t\t" << endl;
778         cout << "[1] Modificar datos personales" << endl;
779         cout << "[2] Mostrar Datos" << endl;
780         cout << "[3] Mostrar Usuario" << endl;
781         cout << "[4] Salir" << endl;
782         cout << "Elija una opción..." << endl;
783         cin >> opc;
784         std::cout << "\033[H\033[2J\033[3J";
785         switch (opc)
786         {
787             case 1:
788             {
789                 //modificarDatosPer();
790                 break;
791             }
```

```
792         case 2:
793         {
794             //mostrarAlumno();
795             break;
796         }
797         case 3:
798         {
799             //mostrarUsuario();
800             break;
801         }
802         case 4:
803         {
804             break;
805         }
806         default:
807         {
808             cout << "Opción no válida" << endl;
809             break;
810         }
811     }
812 }
813 while(opc != 4 );
814 }
815 }
```

Menú Administrador {

```
672 void Menu :: menuAdmin()
673 {
674     int op;
675     do
676     {
677         cout << "\t\t***ADMINISTRADOR***\t\t" << endl;
678         cout << "[1] Crear Usuario" << endl;
679         cout << "[2] Modificar Usuario" << endl;
680         cout << "[3] Mostrar Datos de Usuario" << endl;
681         cout << "[4] Eliminar Usuario" << endl;
682         cout << "[5] Salir" << endl;
683         cout << "Elija una opción..." << endl;
684         cin >> op;
685         std::cout << "\033[H\033[2J\033[3J";
686         switch (op)
687         {
688             case 1:
689             {
690                 crearUsuario();
691                 break;
692             }
```

```
693         case 2:
694         {
695             modificarUsuario();
696             break;
697         }
698         case 3:
699         {
700             verUsuarioAux();
701             break;
702         }
703         case 4:
704         {
705             //eliminarUsuario();
706             break;
707         }
708         case 5:
709         {
710             break;
711         }
712         default:
713         {
714             cout << "Opción no válida" << endl;
715             break;
716         }
717     }
718 }
719 while(op != 5 );
720 }
```

Función Error{

```
14 void error()  
15 {  
16     cout<<"No se pudo abrir el archivo de registros, asegurese que el archivo se  
17     encuentre en\n";  
18     cout<<"la misma ubicación que el programa o que el archivo de texto se  
19     llame: \n";  
20     cout<<"usuarios.txt, si el archivo tiene otro nombre renómbrelo al ya  
21     mencionado\n\n";  
22 }
```

}

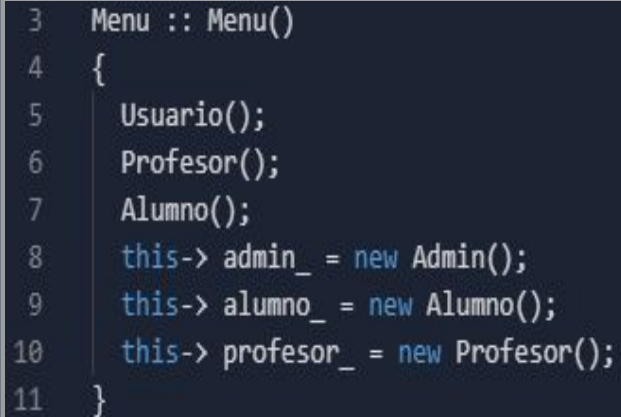
Class Menu{

```
1  #ifndef MENU_H_
2  #define MENU_H_
3
4  class Menu
5  {
6      public:
7          Menu();
8          Usuario *usuario_;
9          Admin *admin_;
10         //vector<Admin> admins();
11         Profesor *profesor_;
12         Alumno *alumno_;
13
14         void menuPrincipal();
15         void crearAdmin();
16         void crearProf();
17         void crearAlumno();
18         void crearUsuario();
19         void fin();
```

```
}
```

```
181 //ADMIN
182 void menuAdmin();
183 void crearUsuario();
184 void eliminarUsuario();
185 void verUsuario();
186 void verUsuarioAd();
187 void modificarUsuario();
188
189 //PROF
190 void menuProf();
191 void modificarNotas();
192 void modificarDatosP();
193 void mostrarDatosP();
194 void verUsuarioP();
195
196 //ALUMNO
197 void menuAlumno();
198 void modificarDatosA();
199 void mostrarDatosA();
200 void verUsuarioA();
201 };
```

Class Menu{



```
3  Menu :: Menu()  
4  {  
5      Usuario();  
6      Profesor();  
7      Alumno();  
8      this-> admin_ = new Admin();  
9      this-> alumno_ = new Alumno();  
10     this-> profesor_ = new Profesor();  
11 }
```

}

1
2
3
4
5
6
7
8
9
10
11
12
13
14



GIT HUB {

https://github.com/Anthony-Rodriguez18/PROYECTO_BASE_U_SEM3



}



1
2
3
4
5
6
7
8
9
10
11
12
13
14



```
int main {  
    COUT<<"GRACIAS";  
}
```



}

