

Universidad Mariano Gálvez de Guatemala
Facultad de Ingeniería Matemática y Ciencias Físicas
Campus Villa Nueva Guatemala
Ingeniería en Sistemas de Información y Ciencias de la Computación
Curso: Programación I
Código de la carrera: 5090
Código de curso: 012
Sección: A
Fecha: 8/04/2024



Laboratorio VII

Fátima Lourdes Santos Guzmán. 23-5148

Introducción

En la presente documentación del laboratorio 7, se realizaron programas que contiene los temas de abstracción, instanciación, clases, objetos, métodos, funciones, bibliotecas string e iostream, y finalmente los algoritmos de ordenamiento. Estos amplios temas se fueron aplicando en los ejercicios como ordenamiento de nombres de estudiantes con el algoritmo de burbuja. En otros casos se fue realizando varias clases generales también objetos y métodos para obtener datos generales de estudiantes que se fueran ingresando. Estos temas que se desarrollaron en este laboratorio fueron de volver a recordar ya que el tema de clases fue uno de los primeros que vimos de esta unidad. El tema de archivos es más fácil poder manejarlo y es un tema muy complejo que se pueden desarrollar fácilmente y abrir sin ningún problema el archivo con información que se ingresa desde consola. A continuación, se mostrará a detalle los ejercicios que se realizaron y con explicación en el código en Dev C++.

Laboratorio 7

Archivos y POO

Manipulación de Archivos:

Crear un programa en C++ que permita al usuario ingresar datos (por ejemplo, nombres y edades de personas) y los almacene en un archivo de texto utilizando la biblioteca `fstream`. Implementar una función que lea los datos del archivo y los muestre en la pantalla.

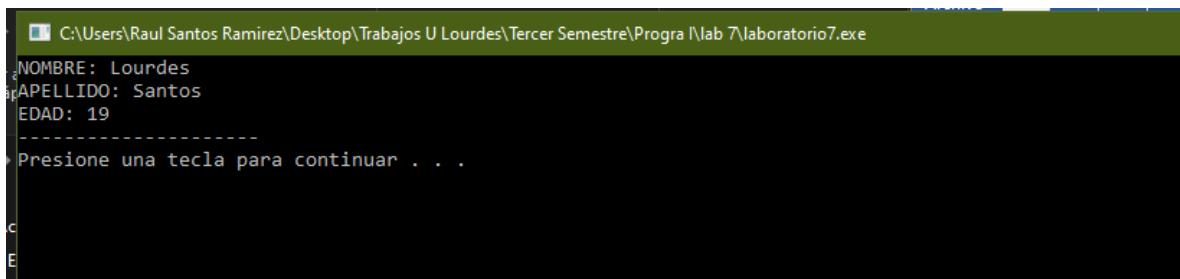
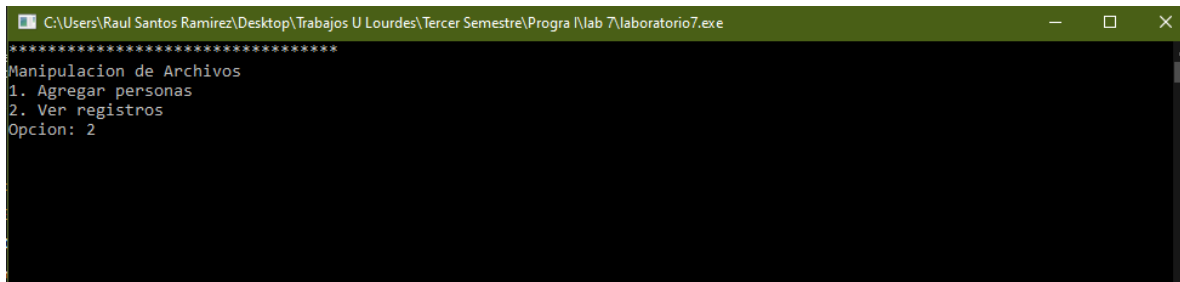
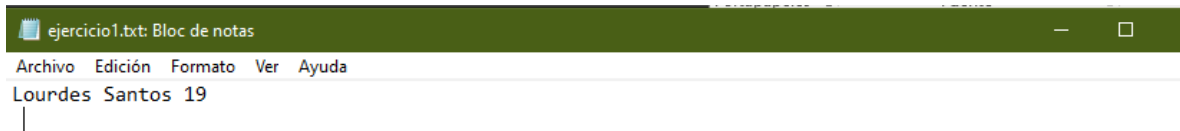
En este programa uno de los desafíos que tuve que enfrentar fue al poder agregar datos directamente a la consola, porque normalmente solo había hecho ejercicios cuando los datos están establecidos desde el código.

The image shows a C++ IDE with two windows. The top window displays the source code for `laboratorio7.cpp`, which includes `<iostream>` and `<stdlib.h>`. The code defines a menu for file manipulation and a `while` loop for user input. The bottom window shows the program's execution, where the user has entered 'Lourdes' for the name, 'Santos' for the last name, and '19' for the age.

```
1 #include <iostream>
2 #include <stdlib.h>
3
4
5 *****
6 Manipulacion de Archivos
7 1. Agregar personas
8 2. Ver registros
9 Opcion: 1
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43 while(!flec.eof()){ //se inicia un ciclo while para poder ir mostrando los datos que se han agregado anteriormente
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Execution output:

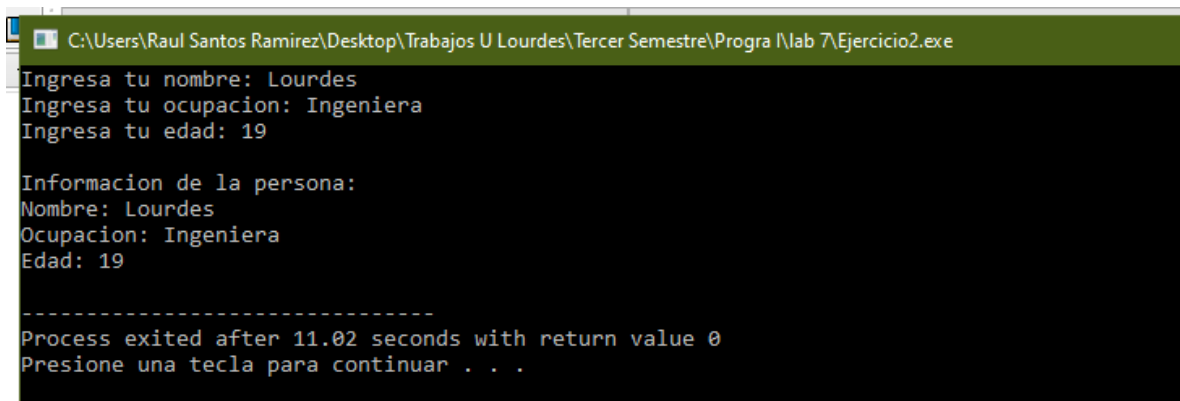
```
C:\Users\Raul Santos Ramirez\Desktop\Trabajos U Lourdes\Tercer Semestre\Progra I\lab 7\laboratorio7.exe
Ingresa tu nombre: Lourdes
Ingresa tu apellido: Santos
Ingresa tu edad: 19
```



Programación Orientada a Objetos:

Definir una clase llamada "Persona" con atributos como nombre, edad y ocupación. Implementar métodos para establecer y obtener los valores de los atributos. Crear objetos de la clase Persona y utilizar los métodos para modificar y mostrar la información de cada persona.

En este programa tuve como desafío implementar datos desde la consola y que pudieran compilarse correctamente. Uno de los errores que me provocaban era que estaba confundida porque estaba pidiendo datos desde la clase y no desde el menú principal.



3. Abstracción e Instanciación: Explicar el concepto de abstracción en la programación orientada a objetos y cómo se relaciona con la definición de clases. Demostrar la instanciación de objetos utilizando la clase Persona creada en la actividad anterior

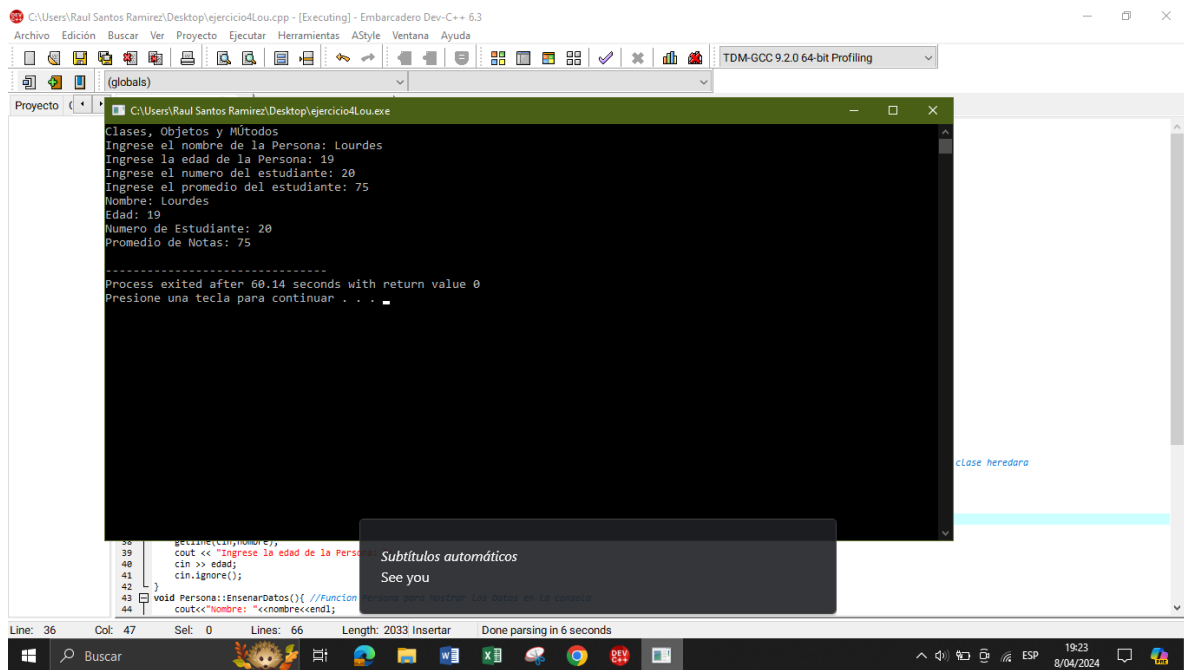
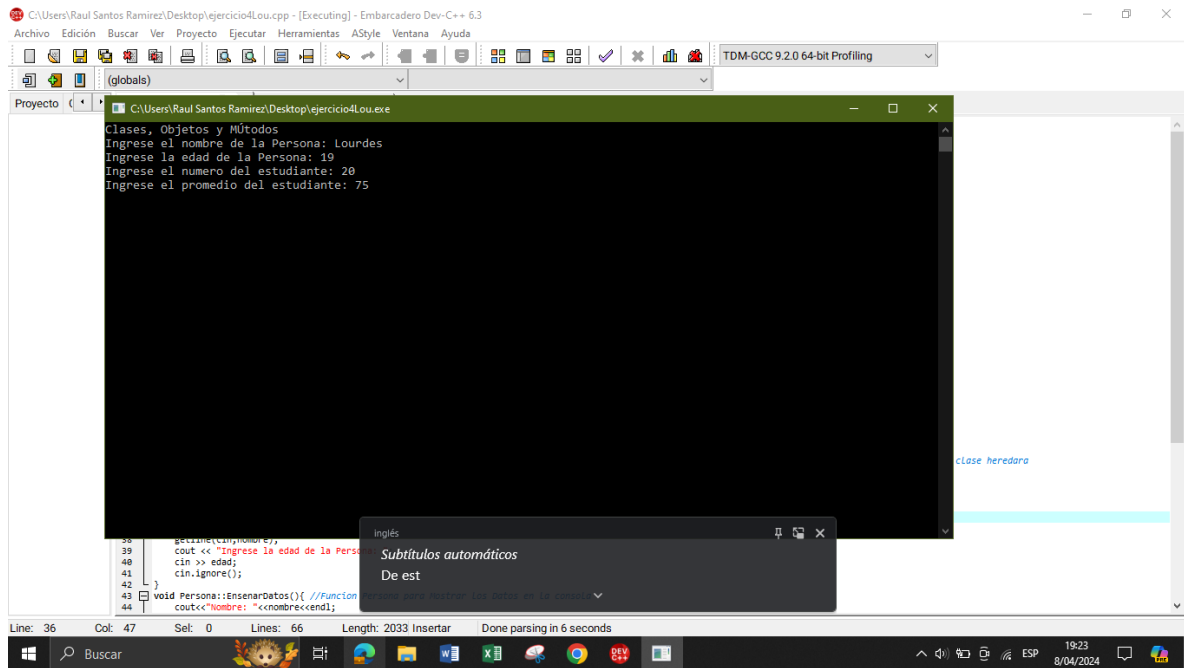
La abstracción orientada a objetos se refiere a características específicas de algún objeto que se haya declarado en la clase, lo van distinguiendo de los demás por los tipos de objetos que logran definir límites conceptuales respecto a quien se hace la referencia de abstracción. Se enfoca más en una visión externa del objeto. Esta se relaciona con la definición de clases es que la abstracción nos permite ir creando clases que van capturando funciones relevantes y van ocultando detalles internos.



```
19      cout << "Ingresa tu nombre: ";
20      cin >> nombre;
21
22      void ocupacionP(){ //metodos
23          cout << "Ingresa tu ocupacion: ";
24          cin >> ocupacion;
25      }
26      void edadP(){ //metodos
27          cout << "Ingresa tu edad: ";
28          cin >> edad;
29      }
30
31      void mostrarInfo() { //metodos
32          cout << "Nombre: " << nombre << endl;
33          cout << "Ocupacion: " << ocupacion << endl;
34          cout << "Edad: " << edad << endl;
35      }
36
37  };
38
39  int main() {
40      string nombre;
41      string ocupacion;
42      int edad;
43
44      cout << "Ingresa tu nombre: ";
45      cin >> nombre;
46      cout << "Ingresa tu ocupacion: ";
47      cin >> ocupacion;
48      cout << "Ingresa tu edad: ";
49      cin >> edad;
50      datos.nombreP(nombre); //instanciacion de objetos de la clase
51      datos.ocupacionP(ocupacion); //instanciacion de objetos de la clase
52      datos.edadP(edad); //instanciacion de objetos de la clase
53      Persona datos(nombre, ocupacion, edad);
54
55      cout << "\nInformacion de la persona: " << endl;
56      datos.mostrarInfo(); //En esta parte se demuestra la instanciacion del objeto en la clase
57      return 0;
58  }
59
60
61
62 }
```

4. Clases, Objetos y Métodos: Desarrollar una clase llamada "Estudiante" que herede de la clase "Persona" y que tenga atributos adicionales como número de estudiante y promedio de calificaciones. Implementar métodos para establecer y obtener los valores de los atributos específicos de los estudiantes. Crear objetos de la clase Estudiante y utilizar los métodos heredados y propios.

En este ejercicio el desafío que tuve que lograr fue como implementar varias clases dentro de un solo archivo. Y también cómo se maneja correctamente los métodos heredados y propios. Solicitando datos desde consola y compilarlo correctamente.



5. Bibliotecas Estándar: Utilizar la biblioteca string para manipular cadenas de caracteres en los programas anteriores. Reemplazar los arreglos de caracteres por objetos de tipo string en las entradas de datos.

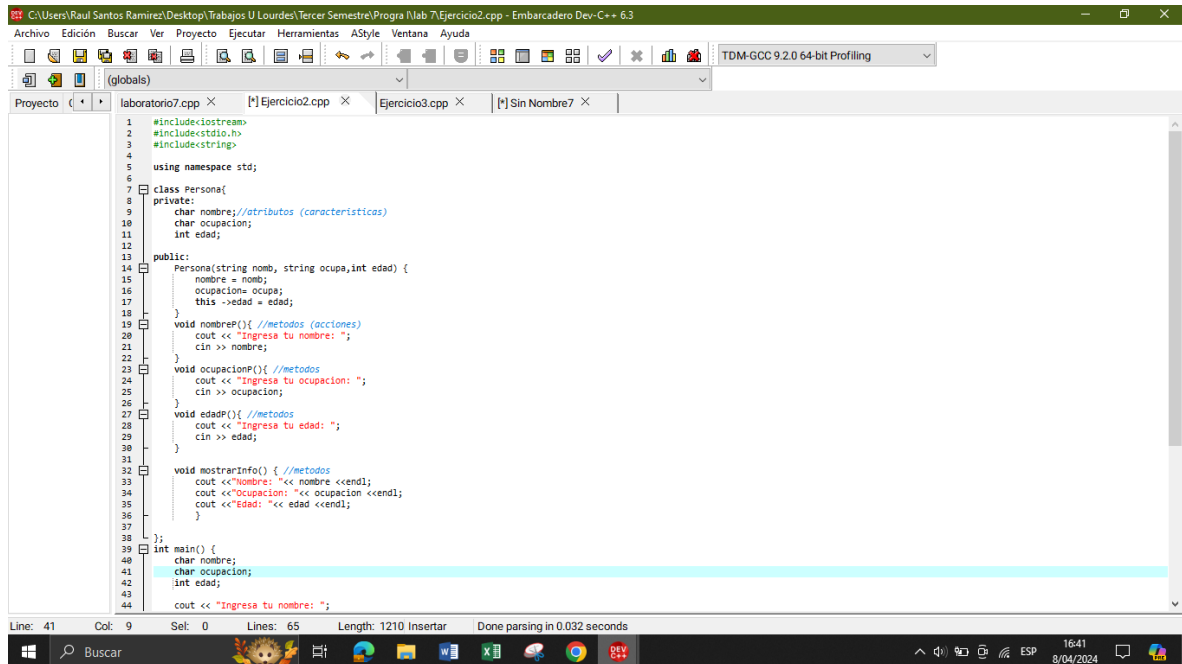
Declarada la biblioteca string y el tipo de dato char

```
1 #include <iostream>
2 #include <stdlib.h>
3 #include <fstream>
4 #include <string>
5 #include <windows.h>
6
7
8 using namespace std;
9
10 int menu(){ //funcion menu donde se pone las opciones que se necesita para el programa
11     int x; //variable
12     system("cls");
13     cout << "*****" << endl;
14     cout << "Manipulacion de Archivos" << endl;
15     cout << "1. Agregar personas" << endl;
16     cout << "2. Ver registros" << endl; //opciones
17     cout << "Opcion: ";
18     cin >> x;
19     return x; //se retorna la variable x
20 }
21
22 void agregar(ofstream &es){ //funcion agregar para atrapar los datos
23     char Nom;
24     char Ape;
25     char edad; //se indican las variables a usar
26     es.open("ejercicio1.txt", ios::out | ios::app); //se abre el archivo de texto
27     cout << "Ingresa tu nombre: ";
28     cin >> Nom;
29     cout << "Ingresa tu apellido: ";
30     cin >> Ape;
31     cout << "Ingresa tu edad: ";
32     cin >> edad;
33     es << Nom << " " << Ape << " " << edad << "\n "; //concatenamos los datos que se han ingresado a las variables
34     es.close(); //se cierra el programa
35 }
36
37 void ver(ifstream &lec){ //funcion de ver los datos para la consola
38     system("cls");
39     char Nom;
40     char Ape;
41     char edad; //se indican las variables
42     lec.open("ejercicio1.txt", ios::in); //se abre el archivo de texto donde se han agregado los datos
43     Lec << Nom;
44     while(!lec.eof()){ //se inicia un ciclo while para poder ir mostrando los datos que se han agregado anteriormente
```

Datos cambiados de tipo string.

```
1 #include <iostream>
2 #include <stdlib.h>
3 #include <fstream>
4 #include <string>
5 #include <windows.h>
6
7
8 using namespace std;
9
10 int menu(){ //funcion menu donde se pone las opciones que se necesita para el programa
11     int x; //variable
12     system("cls");
13     cout << "*****" << endl;
14     cout << "Manipulacion de Archivos" << endl;
15     cout << "1. Agregar personas" << endl;
16     cout << "2. Ver registros" << endl; //opciones
17     cout << "Opcion: ";
18     cin >> x;
19     return x; //se retorna la variable x
20 }
21
22 void agregar(ofstream &es){ //funcion agregar para atrapar los datos
23     string Nom;
24     string Ape;
25     string edad; //se indican las variables a usar
26     es.open("ejercicio1.txt", ios::out | ios::app); //se abre el archivo de texto
27     cout << "Ingresa tu nombre: ";
28     cin >> Nom;
29     cout << "Ingresa tu apellido: ";
30     cin >> Ape;
31     cout << "Ingresa tu edad: ";
32     cin >> edad;
33     es << Nom << " " << Ape << " " << edad << "\n "; //concatenamos los datos que se han ingresado a las variables
34     es.close(); //se cierra el programa
35 }
36
37 void ver(ifstream &lec){ //funcion de ver los datos para la consola
38     system("cls");
39     string Nom;
40     string Ape;
41     string edad; //se indican las variables
42     lec.open("ejercicio1.txt", ios::in); //se abre el archivo de texto donde se han agregado los datos
43     Lec << Nom;
44     while(!lec.eof()){ //se inicia un ciclo while para poder ir mostrando los datos que se han agregado anteriormente
```

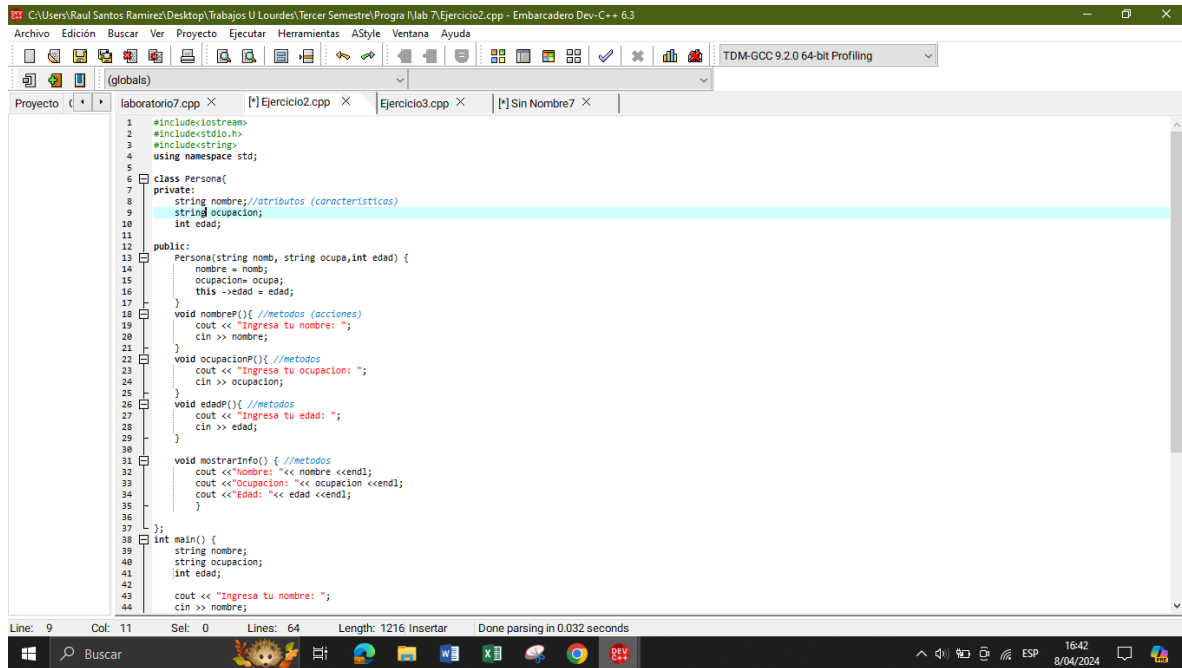
Tipo de dato char



The screenshot shows a C++ IDE with a file named 'Ejercicio2.cpp'. The code defines a 'Persona' class with private attributes 'nombre', 'ocupacion', and 'edad' of type 'char'. The 'main' function prompts the user to enter their name, occupation, and age, storing them in 'char' variables. The status bar at the bottom indicates 'Line: 41 Col: 9 Sel: 0 Lines: 65 Length: 1210 Insertar Done parsing in 0.032 seconds'.

```
1 #include<iostream>
2 #include<stdio.h>
3 #include<string>
4 using namespace std;
5
6 class Persona{
7 private:
8     char nombre;//atributos (caracteristicas)
9     char ocupacion;
10    int edad;
11
12 public:
13     Persona(string nomb, string ocupa,int edad) {
14         nombre = nomb;
15         ocupacion= ocupa;
16         this->edad = edad;
17     }
18     void nombreP(){ //metodos (acciones)
19         cout << "Ingresa tu nombre: ";
20         cin >> nombre;
21     }
22     void ocupacionP(){ //metodos
23         cout << "Ingresa tu ocupacion: ";
24         cin >> ocupacion;
25     }
26     void edadP(){ //metodos
27         cout << "Ingresa tu edad: ";
28         cin >> edad;
29     }
30
31     void mostrarInfo() { //metodos
32         cout <<"Nombre: "<< nombre <<endl;
33         cout <<"Ocupacion: "<< ocupacion <<endl;
34         cout <<"Edad: "<< edad <<endl;
35     }
36 };
37
38 int main() {
39     char nombre;
40     char ocupacion;
41     int edad;
42     cout << "Ingresa tu nombre: ";
```

Cambiado a valor de dato string



The screenshot shows the same C++ IDE with the 'Ejercicio2.cpp' file. The code has been modified to use 'string' for 'nombre' and 'ocupacion', and 'int' for 'edad'. The 'main' function now prompts for a name, occupation, and age, storing them in 'string' and 'int' variables. The status bar at the bottom indicates 'Line: 9 Col: 11 Sel: 0 Lines: 64 Length: 1216 Insertar Done parsing in 0.032 seconds'.

```
1 #include<iostream>
2 #include<stdio.h>
3 #include<string>
4 using namespace std;
5
6 class Persona{
7 private:
8     string nombre;//atributos (caracteristicas)
9     string ocupacion;
10    int edad;
11
12 public:
13     Persona(string nomb, string ocupa,int edad) {
14         nombre = nomb;
15         ocupacion= ocupa;
16         this->edad = edad;
17     }
18     void nombreP(){ //metodos (acciones)
19         cout << "Ingresa tu nombre: ";
20         cin >> nombre;
21     }
22     void ocupacionP(){ //metodos
23         cout << "Ingresa tu ocupacion: ";
24         cin >> ocupacion;
25     }
26     void edadP(){ //metodos
27         cout << "Ingresa tu edad: ";
28         cin >> edad;
29     }
30
31     void mostrarInfo() { //metodos
32         cout <<"Nombre: "<< nombre <<endl;
33         cout <<"Ocupacion: "<< ocupacion <<endl;
34         cout <<"Edad: "<< edad <<endl;
35     }
36 };
37
38 int main() {
39     string nombre;
40     string ocupacion;
41     int edad;
42     cout << "Ingresa tu nombre: ";
43     cin >> nombre;
```



```
1 #include<iostream>
2 #include<stdlib.h>
3 #include <string>
4 using namespace std;
5
6 class Persona{
7     private:
8         char nombre;
9         int edad;
10    public:
11        Persona(string, int); //En esta parte se declara la parte del constructor
12        void SolicitarDatos();
13        void EnseñarDatos();
14};
15
16 class Estudiante: public Persona{
17     private:
18         int NumeroEstudiante;
19         char PromedioNotas;
20    public:
21        //Este constructor pertenece a la clase Estudiante
22        Estudiante(string, int, float);
23        void SolicitarDatos();
24        void EnseñarDatos();
25};
26
27 Persona::Persona(string nom,int edad){ //Esta es la parte del constructor de la clase Persona, es los métodos propios.
28     nombre=nom;
29     edad=edad;
30 }
31
32 Estudiante::Estudiante(string nom,int edad, int num, float prom):Persona(nom, edad){ //Esta es la parte del constructor de la clase Estudiante que son los métodos heredados
33     NumeroEstudiante = num;
34     PromedioNotas= prom;
35 }
36
37 void Persona::SolicitarDatos(){ //En esta función Persona para solicitar los datos de la consola
38     cout << "Clases, Objetos y Métodos" << endl;
39     cout << "Ingrese el nombre de la Persona: ";
40     getline(cin,nombre);
41     cout << "Ingrese la edad de la Persona: ";
42     cin >> edad;
43     cin.ignore();
44 }
45
46 void Persona::EnseñarDatos(){ //Función Persona para mostrar los datos en la consola
47     cout<<"Nombre: "<<nombre<<endl;
```

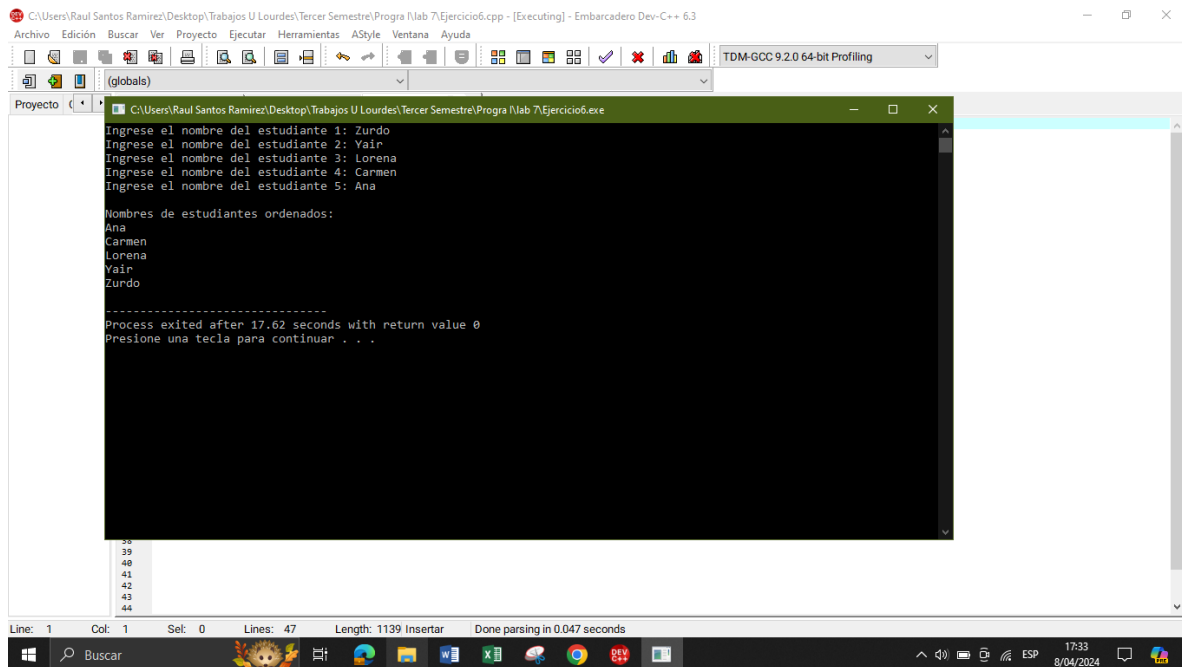
En esta foto está establecida las variables de datos char

```
1 #include<iostream>
2 #include<stdlib.h>
3 #include <string>
4 using namespace std;
5
6 class Persona{
7     private:
8         string nombre;
9         int edad;
10    public:
11        Persona(string, int); //En esta parte se declara la parte del constructor
12        void SolicitarDatos();
13        void EnseñarDatos();
14};
15
16 class Estudiante: public Persona{
17     private:
18         int NumeroEstudiante;
19         float PromedioNotas;
20    public:
21        //Este constructor pertenece a la clase Estudiante
22        Estudiante(string, int, float);
23        void SolicitarDatos();
24        void EnseñarDatos();
25};
26
27 Persona::Persona(string nom,int edad){ //Esta es la parte del constructor de la clase Persona, es los métodos propios.
28     nombre=nom;
29     edad=edad;
30 }
31
32 Estudiante::Estudiante(string nom,int edad, int num, float prom):Persona(nom, edad){ //Esta es la parte del constructor de la clase Estudiante que son los métodos heredados
33     NumeroEstudiante = num;
34     PromedioNotas= prom;
35 }
36
37 void Persona::SolicitarDatos(){ //En esta función Persona para solicitar los datos de la consola
38     cout << "Clases, Objetos y Métodos" << endl;
39     cout << "Ingrese el nombre de la Persona: ";
40     getline(cin,nombre);
41     cout << "Ingrese la edad de la Persona: ";
42     cin >> edad;
43     cin.ignore();
44 }
45
46 void Persona::EnseñarDatos(){ //Función Persona para mostrar los datos en la consola
47     cout<<"Nombre: "<<nombre<<endl;
```

Y en esta captura fue ya cambiado las variables a string, implementando también la función string.

6. Algoritmo de Ordenamiento: Implementar un algoritmo de ordenamiento (por ejemplo, el algoritmo de burbuja) para ordenar una lista de nombres de estudiantes. Mostrar los nombres ordenados en pantalla.

El desafío que tuve en este ejercicio fue que no había trabajado con algoritmos de ordenamiento sin tener que usar vectores y como analizar que los nombres se puedan ir ordenando correctamente.



The screenshot shows a C++ program running in a terminal window. The program prompts the user to enter five student names. The names entered are Zurdo, Vair, Lorena, Carmen, and Ana. The program then displays the names in sorted order: Ana, Carmen, Lorena, Vair, and Zurdo. The terminal window is titled "C:\Users\Raul Santos Ramirez\Desktop\Trabajos U Lourdes\Tercer Semestre\Progra I\lab 7\Ejercicio6.exe". The status bar at the bottom indicates the program exited after 17.62 seconds with a return value of 0.

```
C:\Users\Raul Santos Ramirez\Desktop\Trabajos U Lourdes\Tercer Semestre\Progra I\lab 7\Ejercicio6.exe
Archivo Edición Buscar Ver Proyecto Ejecutar Herramientas AStyle Ventana Ayuda
(globals)
Proyecto C:\Users\Raul Santos Ramirez\Desktop\Trabajos U Lourdes\Tercer Semestre\Progra I\lab 7\Ejercicio6.exe
Ingrese el nombre del estudiante 1: Zurdo
Ingrese el nombre del estudiante 2: Vair
Ingrese el nombre del estudiante 3: Lorena
Ingrese el nombre del estudiante 4: Carmen
Ingrese el nombre del estudiante 5: Ana

Nombres de estudiantes ordenados:
Ana
Carmen
Lorena
Vair
Zurdo

-----
Process exited after 17.62 seconds with return value 0
Presione una tecla para continuar . . .
```

Conclusión

En conclusión, en este laboratorio se puso en práctica el tema de archivos y POO. Fue un laboratorio muy complejo con un poco de dificultad, para implementar distintas clases en un solo archivo. El poder obtener datos desde consola es un tema que se me hizo difícil, pero con ayuda de tutoriales pude resolver correctamente lo solicitado. Fue una buena práctica poder recordar cómo se manejar las clases tanto públicas como privadas y los distintos métodos y atributos que se tiene que crear. En el tema de archivos es un tema que puedo manejar bien y el ejercicio que pusimos en práctica es un ejercicio básico de fácil de crear.

Fue muy buen aprendizaje que tuve al practicar en cada uno de estos ejercicios en este laboratorio.

Link del repositorio de Git

<https://github.com/lurdeees/Laboratorio7.-Lourdes-Santos.git>