



UNIVERSIDAD MARIANO GALVEZ DE GUATEMALA

Curso: Programación, Jornada Matutina

Sección: "A"

Catedrática: Ing. Carlos Alejandro Arias.



Carnet: __5090-23-1762

Nombre completo: Roberto Carlos de León Gramajo

Fecha entrega: 17/04/2024

Segundo Parcial del semestre

Introducción

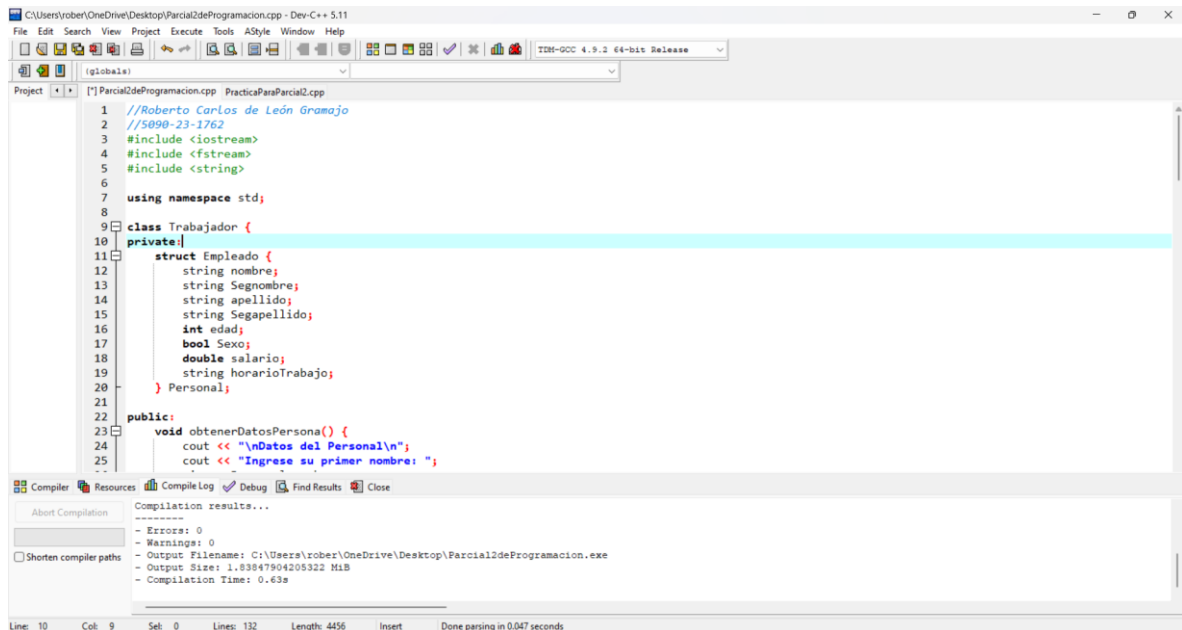
- Diseña una clase llamada Empleado con los siguientes atributos: nombre, apellido, edad y salario.
- Implementa métodos para establecer y obtener los valores de los atributos.
- Crea un programa principal que permita al usuario ingresar los datos de varios empleados y los almacene en un archivo de texto llamado empleados.txt;.
- Asegúrate de que el programa pueda manejar la escritura y la lectura de los datos del archivo de manera adecuada.

2. Consulta de Empleados

Instrucciones:

- Implementa un programa que lea los datos del archivo empleados.txt; creado en el Programa 1.
- Crea una función que reciba el nombre completo de un empleado como entrada y muestre su información completa (nombre, apellido, edad y salario) si existe en el archivo.
- Permite al usuario realizar consultas múltiples de empleados hasta que decida salir del programa.
- Asegúrate de manejar adecuadamente el caso en el que el empleado buscado no exista en el archivo.

Programa

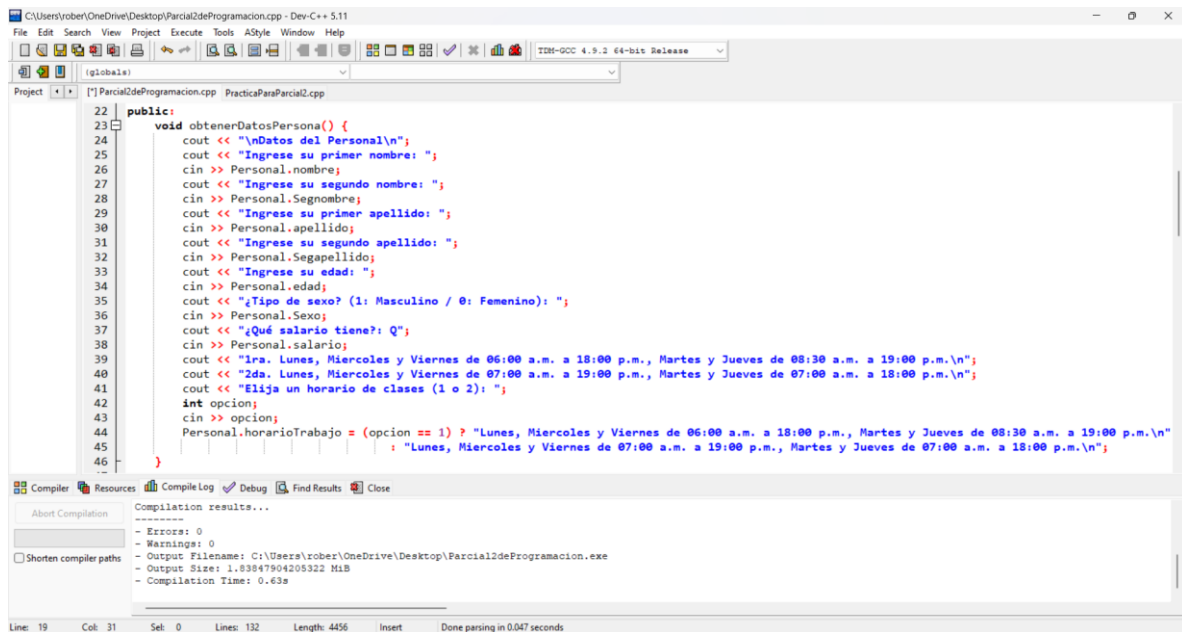


The screenshot shows a C++ IDE with the following code in `PracticaParaParcial2.cpp`:

```
1 //Roberto Carlos de León Gramajo
2 //5090-23-1762
3 #include <iostream>
4 #include <fstream>
5 #include <string>
6
7 using namespace std;
8
9 class Trabajador {
10 private:
11     struct Empleado {
12         string nombre;
13         string Segnombre;
14         string apellido;
15         string Segapellido;
16         int edad;
17         bool Sexo;
18         double salario;
19         string horarioTrabajo;
20     } Personal;
21
22 public:
23     void obtenerDatosPersona() {
24         cout << "\nDatos del Personal\n";
25         cout << "Ingrese su primer nombre: ";
```

The compiler output at the bottom shows 0 errors and 0 warnings, with an output file named `C:\Users\rober\OneDrive\Desktop\Parcial2deProgramacion.exe`.

Definimos nuestra clase con atributos por medio de una estructura, eso nos ayudara a tener más orden de donde proviene nuestro atributo, y para facilitar el uso de string en nuestro programa.

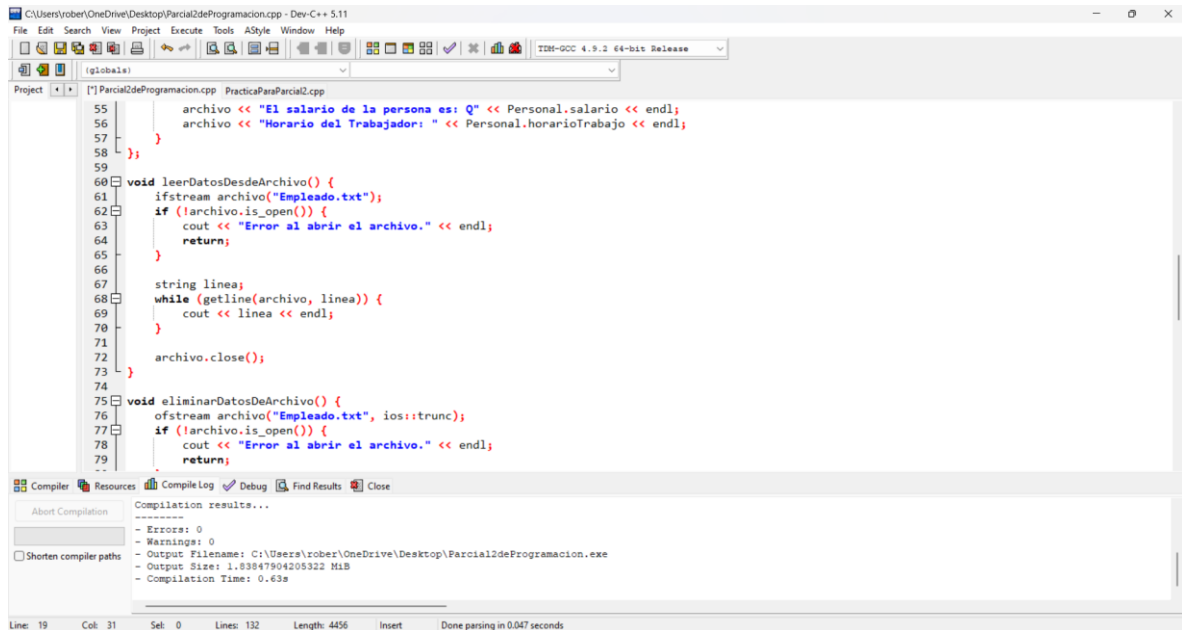


The screenshot shows the same C++ IDE with the code updated to include input for sex, salary, and work schedule:

```
22 public:
23     void obtenerDatosPersona() {
24         cout << "\nDatos del Personal\n";
25         cout << "Ingrese su primer nombre: ";
26         cin >> Personal.nombre;
27         cout << "Ingrese su segundo nombre: ";
28         cin >> Personal.Segnombre;
29         cout << "Ingrese su primer apellido: ";
30         cin >> Personal.apellido;
31         cout << "Ingrese su segundo apellido: ";
32         cin >> Personal.Segapellido;
33         cout << "Ingrese su edad: ";
34         cin >> Personal.edad;
35         cout << "¿Tipo de sexo? (1: Masculino / 0: Femenino): ";
36         cin >> Personal.Sexo;
37         cout << "¿Qué salario tiene?: Q: ";
38         cin >> Personal.salario;
39         cout << "Lra. Lunes, Miércoles y Viernes de 06:00 a.m. a 18:00 p.m., Martes y Jueves de 08:30 a.m. a 19:00 p.m.\n";
40         cout << "Zda. Lunes, Miércoles y Viernes de 07:00 a.m. a 19:00 p.m., Martes y Jueves de 07:00 a.m. a 18:00 p.m.\n";
41         cout << "Elija un horario de clases (1 o 2): ";
42         int opcion;
43         cin >> opcion;
44         Personal.horarioTrabajo = (opcion == 1) ? "Lunes, Miércoles y Viernes de 06:00 a.m. a 18:00 p.m., Martes y Jueves de 08:30 a.m. a 19:00 p.m.\n"
45                                     : "Lunes, Miércoles y Viernes de 07:00 a.m. a 19:00 p.m., Martes y Jueves de 07:00 a.m. a 18:00 p.m.\n";
46     }
```

The compiler output remains the same, showing 0 errors and 0 warnings.

Agregue horario de trabajo y sexo para nuestro trabajador, porque yo ya había practicado días anteriores al examen, sobre el registro de un trabajador, y considero que son importantes, en este caso las dos opciones se presentan como bool, y la segunda tiene como procedimiento un opción que sería comparado con el dato que el usuario ingresa para horario de trabajo, así dependiendo de la respuesta este nos diera uno de los dos horarios, porque así facilitamos al usuario rellenar los datos, con opciones que uno ya proporciona.



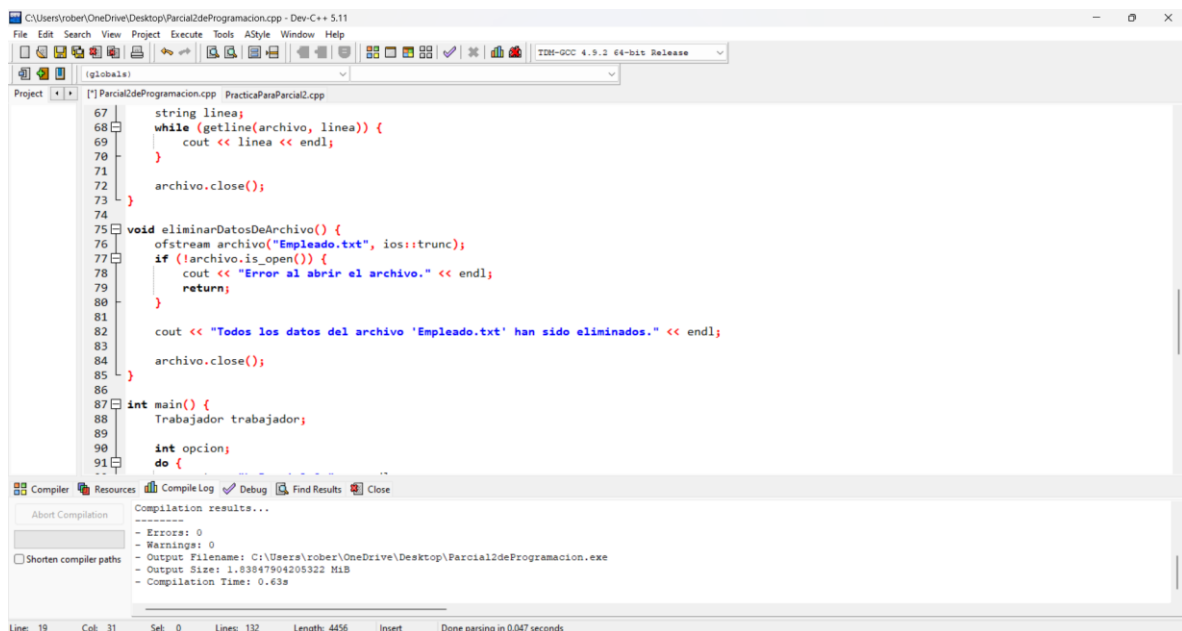
```
55     archivo << "El salario de la persona es: Q" << Personal.salario << endl;
56     archivo << "Horario del Trabajador: " << Personal.horarioTrabajo << endl;
57 }
58 }
59
60 void leerDatosDesdeArchivo() {
61     ifstream archivo("Empleado.txt");
62     if (!archivo.is_open()) {
63         cout << "Error al abrir el archivo." << endl;
64         return;
65     }
66
67     string linea;
68     while (getline(archivo, linea)) {
69         cout << linea << endl;
70     }
71
72     archivo.close();
73 }
74
75 void eliminarDatosDeArchivo() {
76     ofstream archivo("Empleado.txt", ios::trunc);
77     if (!archivo.is_open()) {
78         cout << "Error al abrir el archivo." << endl;
79         return;
80     }
81 }
```

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\rober\OneDrive\Desktop\Parcial2deProgramacion.exe
- Output Size: 1.83847904205322 MiB
- Compilation Time: 0.63s

Line: 19 Col: 31 Sel: 0 Lines: 132 Length: 4456 Insert Done parsing in 0.047 seconds

Parte importante de la programación orientada a objetos es el leer un documento ya creado, porque pude que el proceso lo realice uno, pero que este se guarde en un archivo es impórtate, por ello realizo un procedimiento sin retorno, para verificar si el archivo tiene caracteres los cuales tienen que ser leídos por el código.



```
67     string linea;
68     while (getline(archivo, linea)) {
69         cout << linea << endl;
70     }
71
72     archivo.close();
73 }
74
75 void eliminarDatosDeArchivo() {
76     ofstream archivo("Empleado.txt", ios::trunc);
77     if (!archivo.is_open()) {
78         cout << "Error al abrir el archivo." << endl;
79         return;
80     }
81
82     cout << "Todos los datos del archivo 'Empleado.txt' han sido eliminados." << endl;
83
84     archivo.close();
85 }
86
87 int main() {
88     Trabajador trabajador;
89
90     int opcion;
91     do {
```

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\rober\OneDrive\Desktop\Parcial2deProgramacion.exe
- Output Size: 1.83847904205322 MiB
- Compilation Time: 0.63s

Line: 19 Col: 31 Sel: 0 Lines: 132 Length: 4456 Insert Done parsing in 0.047 seconds

La opción principal de nuestro código es crear, leer, y eliminar, por ello la opción trunc, que nos ayudaría a poder leer por medio de ios, y trunc que es eliminar datos de nuestro programa.

```
84     archivo.close();
85 }
86
87 int main() {
88     Trabajador trabajador;
89
90     int opcion;
91     do {
92         cout << "\nParcial 2:" << endl;
93         cout << "1. Leer datos desde el archivo" << endl;
94         cout << "2. Agregar trabajador" << endl;
95         cout << "3. Eliminar todos los trabajadores" << endl;
96         cout << "4. Salir" << endl;
97         cout << "Seleccione una opcion: ";
98         cin >> opcion;
99
100        switch (opcion) {
101            case 1:
102                system("cls");
103                leerDatosDesdeArchivo();
104                break;
105            case 2: {
106                system("cls");
107                ofstream archivo("Empleado.txt", ios::app);
108                if (!archivo.is_open()) {
```

Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\rober\OneDrive\Desktop\Parcial2deProgramacion.exe
- Output Size: 1.83847904205322 MiB
- Compilation Time: 0.63s

Line: 19 Col: 31 Sel: 0 Lines: 132 Length: 4456 Insert Done parsing in 0.047 seconds

Establecemos por medio de un case varias opciones, que mi programa puede realizar, como un tipo de menú que ayude a la persona a escoger la opción que necesita.

```
Primer Nombre: Roberto
Segundo Nombre: Carlos
Primer Apellido: Leñ
Segundo Apellido: Gramajo
Edad: 20
Tipo de sexo: Masculino
El salario de la persona es: Q3600
Horario de Trabajo: Lunes, Miercoles y Viernes de 07:00 a.m. a 19:00 p.m., Martes y Jueves de 07:00 a.m. a 18:00 p.m.
```

Ln 1, Col 1 | 281 caracteres. | 100% | Windows (CRLF) | ANSI

El archivo creado, como puede ver, el archivo Empleado.txt fue creado exitosamente, por medio de ofstream, que nuestro código nos permite realizar.

Conclusiones:

Quiero iniciar diciendo que si utilice ChatGPT, pero no con los propósitos de copiar, sino con el fin de resolver una duda sobre un error al crear un nuevo objeto en mi programa, primero no sabía que no se podía utilizar, hasta que aviso, ya cuando lo utilice, y segundo fue con los fines de saber cuál era mi error, y no de copiar un código que no fuera mío, más porque me he pasado la semana estudiando y siendo responsable en mi parcial, por ello utilizo códigos que no pidió pero que si tenía por prácticas que había realizado anteriormente, considero que si estuvo mal de mi parte utilizar una herramienta que no permitía, y quiero que sepa que fue utilizado solo para resolver duda sobre error en el código y nada más.

Además de ese error que me dio, estuve practicando con diferentes códigos toda la semana, por ello utilicé procedimiento de más que quería compartir con mi segundo parcial, así demostrando mi habilidades para la programación.

Programa en GitHub

<https://github.com/RobertodeLeonUmg/ParcialDos>.