



instituto politécnico nacional

Escuela superior de ingeniería textil

Fundamentos de programación

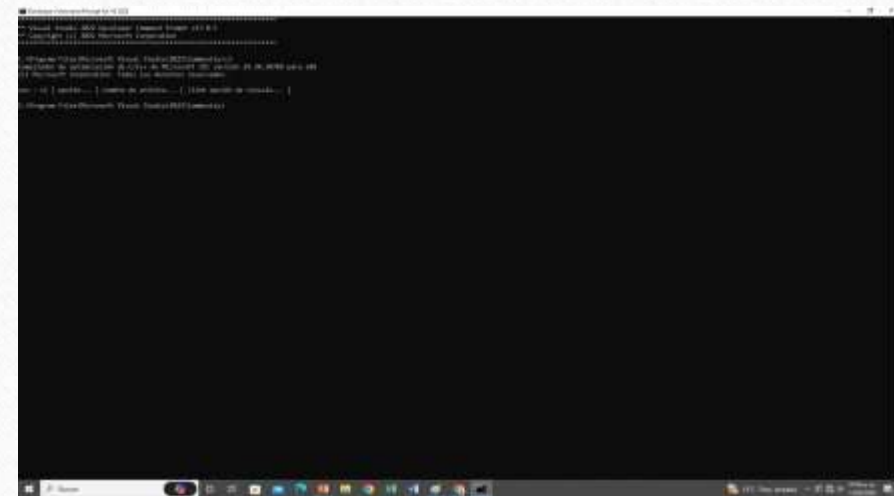
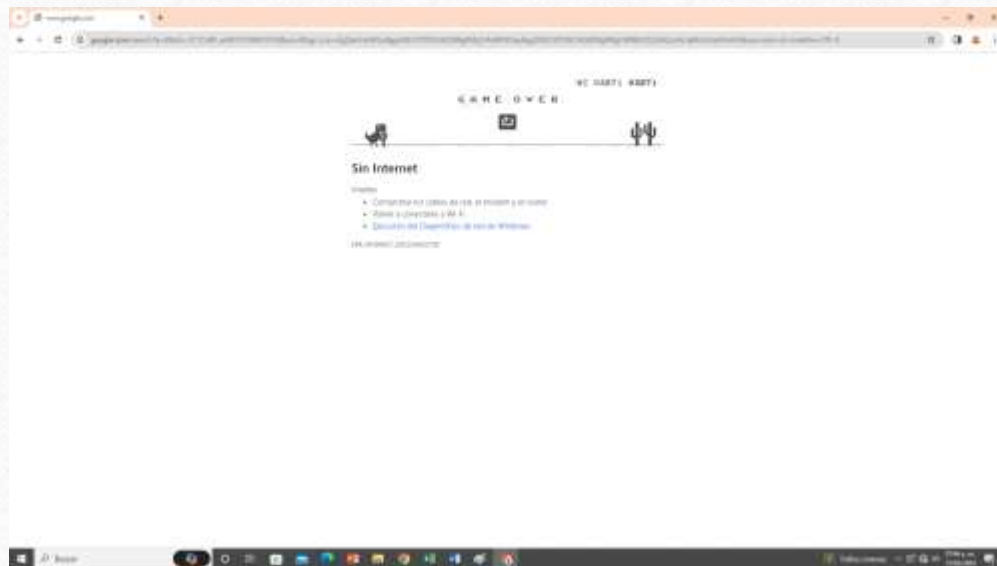
Maestro: Villalobos Martínez Rodolfo

Alumno: González López Alonso

Reporte de actividades de clase

1TM14

Clase 1: Partes de la computadora, encendido, forma de tomar captura, desconectado de cable internet y concurso de juego de dinosaurio.

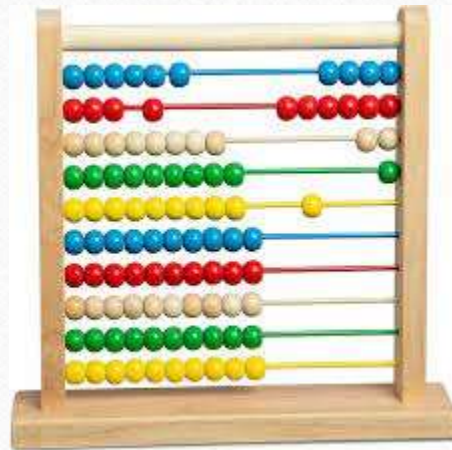


Clase 2

Escribe los siguientes números del sistema decimal 0, 4, 16, 32, 64, 128 al:

- Vigesimal: 0, 4, 16, 1C, 34
- Hexadecimal: 0, 4, 10, 20, 40, 80
- Binario: 0, 100, 1000, 10000, 100000, 1000000
- Algún otro:
- Senario: 0
- Octal: 4
- Terciario: 121

Abaco



¿Para que usan la computadora los ingenieros?

- Los ingenieros civiles trabaja en la realización de proyectos y obras de construcción por medio del computador. El computador es un gran aliado hoy en día para el ingeniero civil ya que ayuda a reducir el tiempo de su trabajo dependiendo el proyecto y de la obra a realizar.

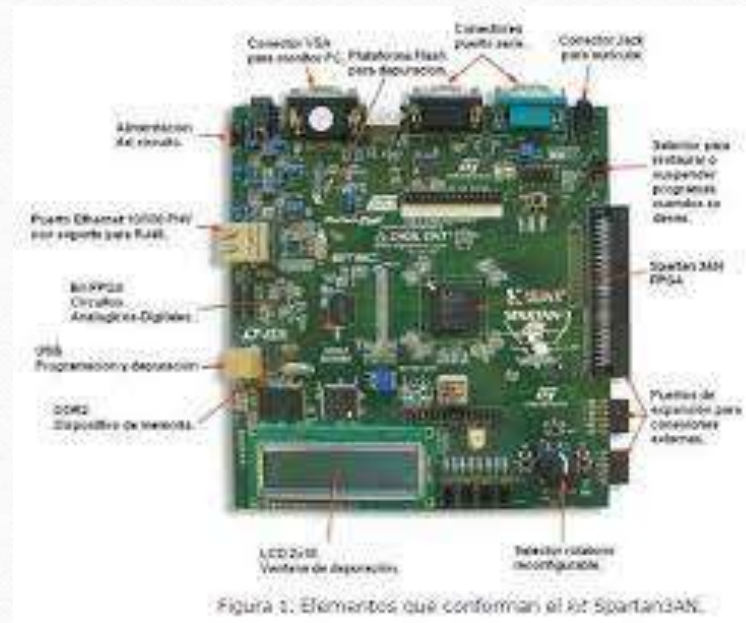
Lenguajes de programación

- Los lenguajes de programación permiten a todos los dispositivos electrónicos, como ordenadores y Smartphones, interpretar instrucciones y códigos que hacen posible crear sitios web, apps, apps web, programas y plataformas que actualmente utilizamos todos a diario.

Clase 3: sistemas embebidos

- Los **sistemas embebidos** son tecnologías invisibles, pero esenciales que impulsan muchos de nuestros dispositivos cotidianos, como consolas de videojuegos, cajas registradoras, decodificadores, teléfonos móviles y lavadoras. En su esencia, estos sistemas son microprocesadores integrados que operan dentro de estos dispositivos, encargándose de procesar información, supervisar y controlar funciones específicas

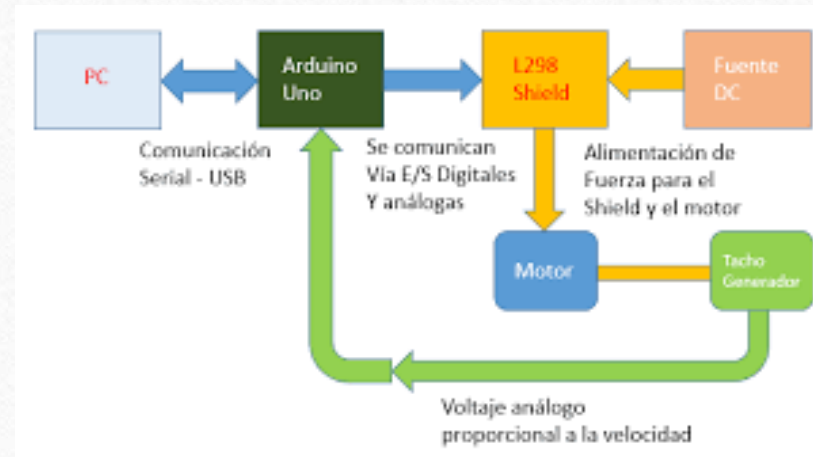
Componentes de los sistemas embebidos



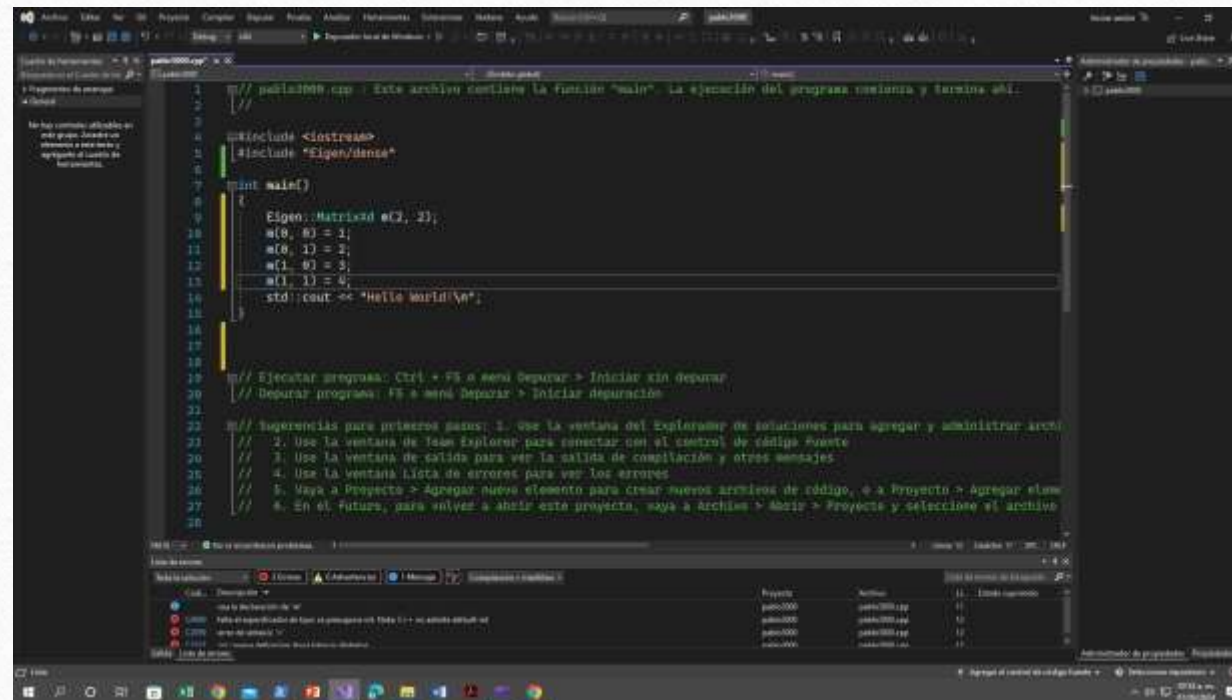
Características

- Tienen un procesador central
- Son sistemas confiables
- Requieren poco o nulo mantenimiento
- Son sistemas seguros
- Son eficientes en cuanto al sistema de energía
- Son de propósito específico
- Tienen interfaz de usuario simple o carecen de ella

Diagrama de bloques



Clase 4



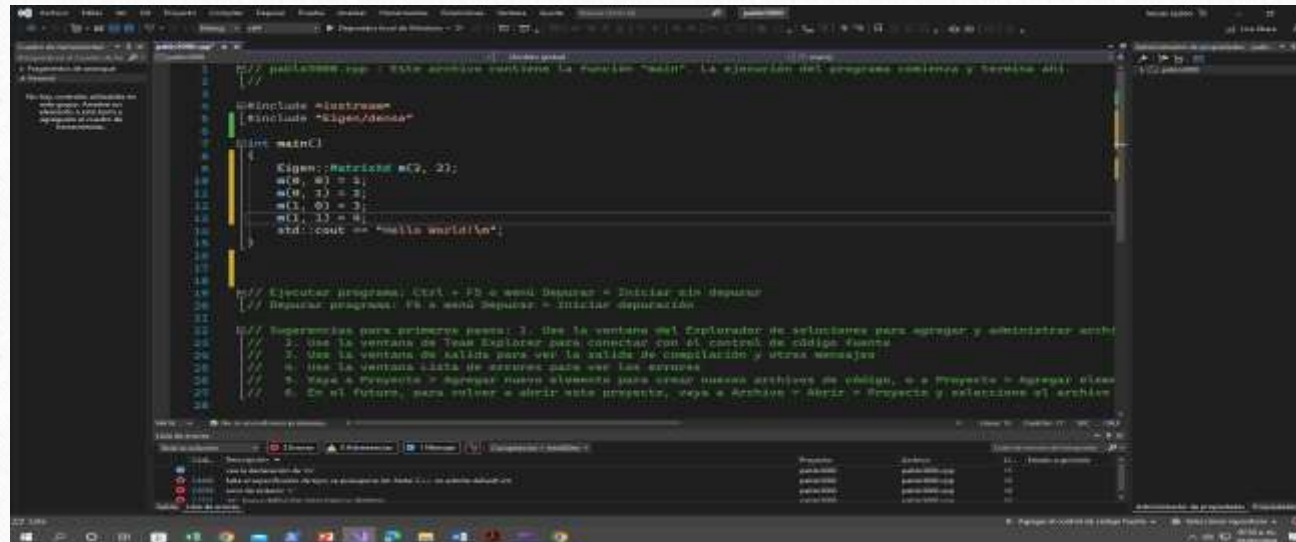
The screenshot shows a C++ IDE with a dark theme. The main editor displays a C++ program named `pdila3000.cpp`. The code includes a comment in Spanish explaining the `main` function, followed by standard C++ headers and a `main` function that initializes a `Matrixed` object and prints "Hello World". Below the code, there are instructions in Spanish for running and debugging the program, along with a list of suggestions for first steps. The bottom of the IDE shows a console window with the output "Hello World" and a file explorer on the right side.

```
1 // pdila3000.cpp : Este archivo contiene la función "main". La ejecución del programa comienza y termina ahí.
2
3
4 #include <iostream>
5 #include "Eigen/dense"
6
7 int main()
8 {
9     Eigen::Matrixd m(2, 2);
10    m(0, 0) = 1;
11    m(0, 1) = 2;
12    m(1, 0) = 3;
13    m(1, 1) = 4;
14    std::cout << "Hello World!\n";
15 }
16
17
18
19 // Ejecutar programa: Ctrl + F5 o menú Depurar > Iniciar sin depurar
20 // Depurar programa: F5 o menú Depurar > Iniciar depuración
21
22 // sugerencias para primeros pasos: 1. Use la ventana del Explorador de soluciones para agregar y administrar archivos.
23 // 2. Use la ventana de Team Explorer para conectar con el control de código fuente.
24 // 3. Use la ventana de Salida para ver la salida de compilación y otros mensajes.
25 // 4. Use la ventana Lista de errores para ver los errores.
26 // 5. Vaya a Proyecto > Agregar nuevo elemento para crear nuevos archivos de código, o a Proyecto > Agregar elemento existente para agregar archivos existentes.
27 // 6. En el futuro, para volver a abrir este proyecto, vaya a Archivo > Abrir > Proyecto y seleccione el archivo.
```

Output: Hello World!

Clase 5

- C++

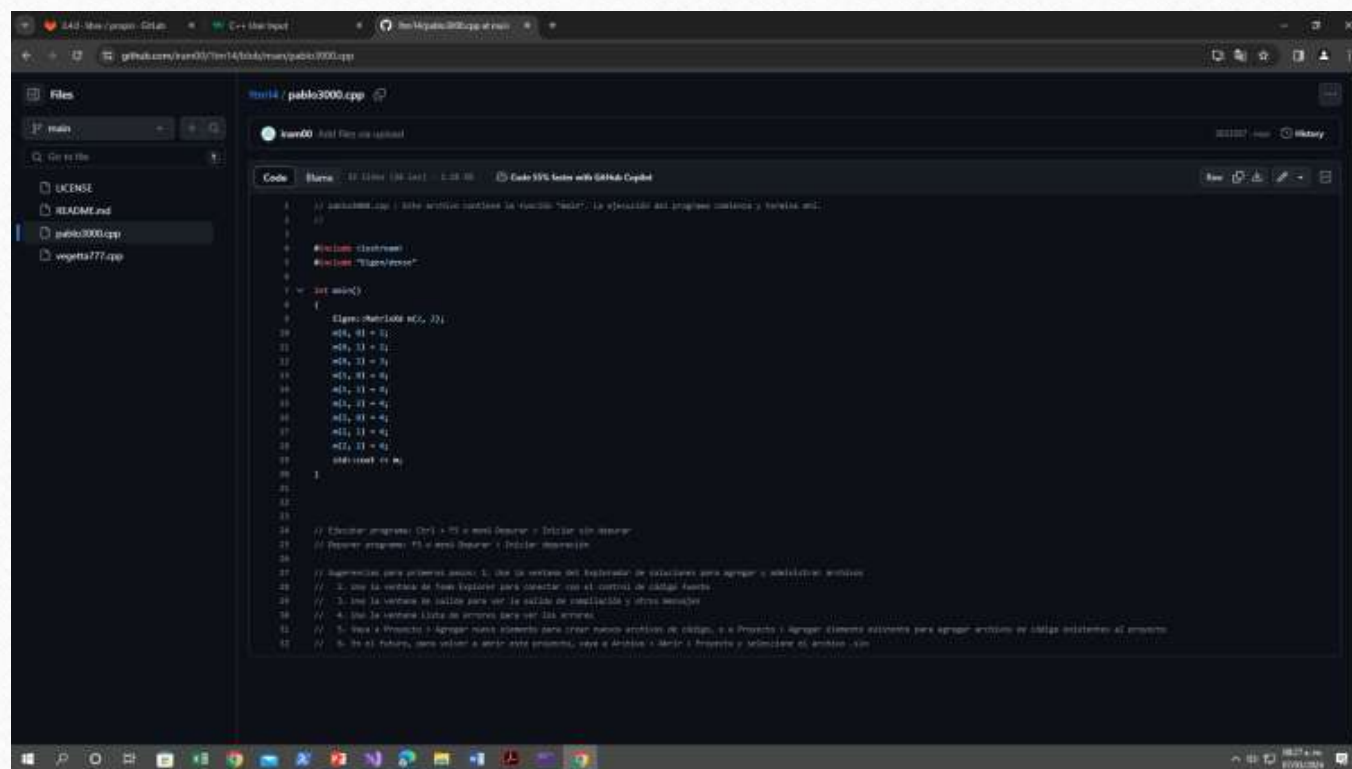


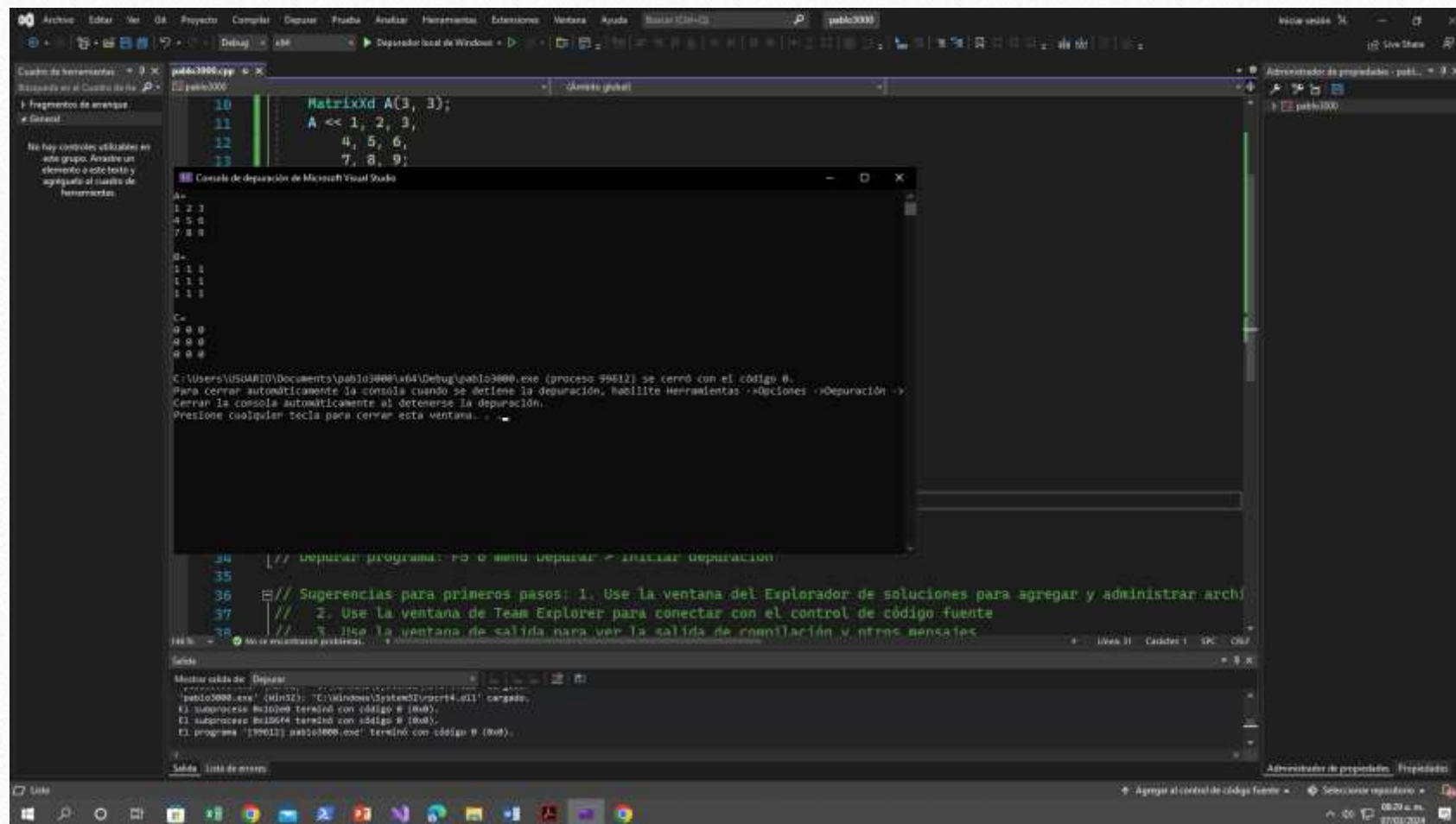
The screenshot shows a Visual Studio IDE with a C++ project. The main code file, `main.cpp`, contains the following code:

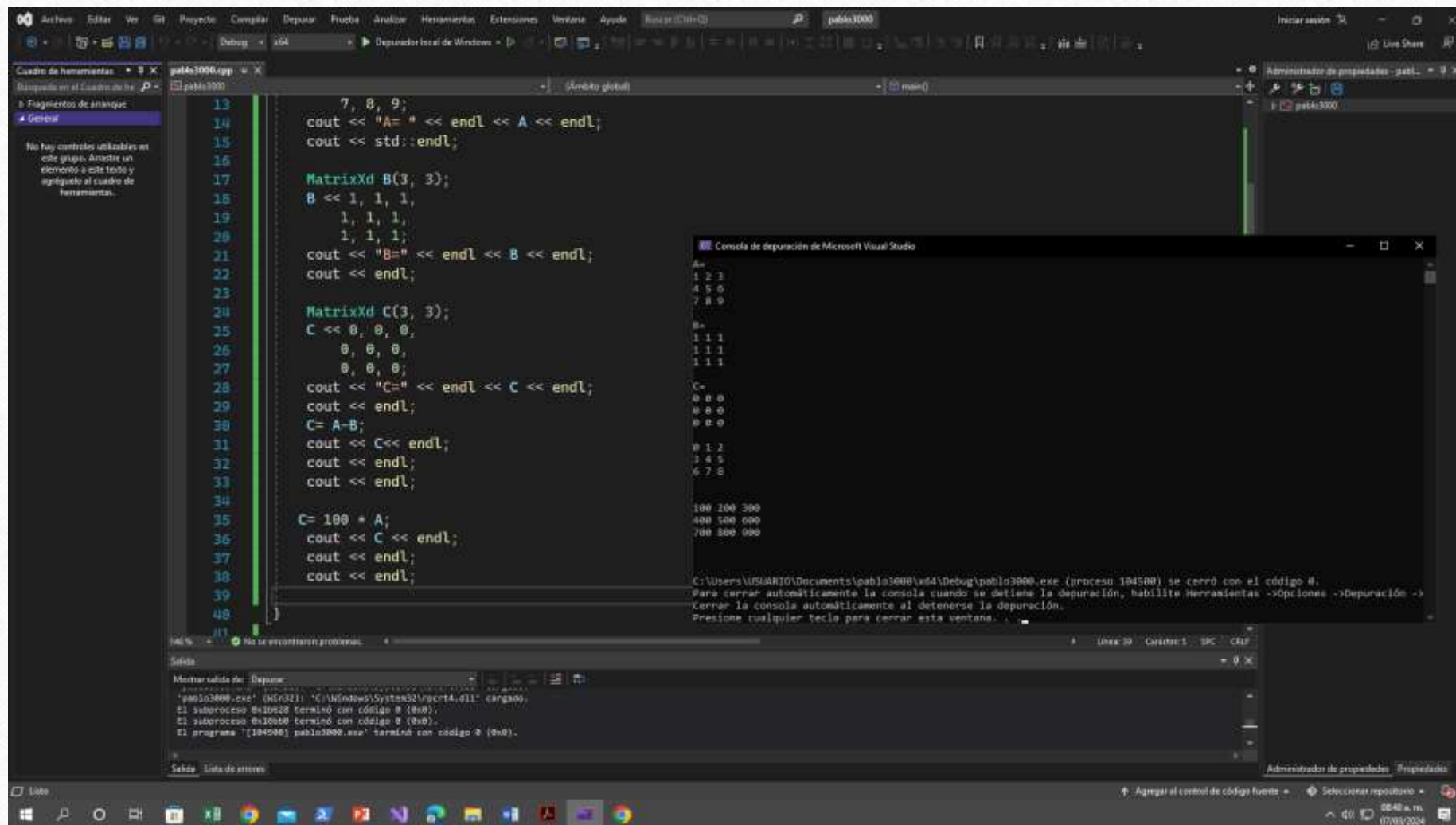
```
1 // main.cpp : Este archivo contiene la función "main". La ejecución del programa comienza y termina allí.
2
3 #include <iostream>
4 #include "Eigen/dense"
5
6 using namespace Eigen;
7
8 int main()
9 {
10     Matrix<double, 2, 2> m1, m2;
11     m1(0, 0) = 1;
12     m1(0, 1) = 2;
13     m1(1, 0) = 3;
14     m1(1, 1) = 4;
15     std::cout << "m1\n";
16 }
17
18 // Ejecutar programa: Ctrl + F5 o menú Depurar > Iniciar sin depurar
19 // Depurar programa: F5 o menú Depurar > Iniciar depuración
20
21 // Superconstrucciones para primeros pasos: 1. Use la ventana del Explorador de soluciones para agregar y administrar archivos.
22 // 2. Use la ventana de Team Explorer para conectar con el control de código fuente.
23 // 3. Use la ventana de Salida para ver la salida de compilación y otras mensajes.
24 // 4. Use la ventana Lista de errores para ver los errores.
25 // 5. Vaya a Proyecto > Agregar nuevo elemento para crear nuevos archivos de código, o a Proyecto > Agregar elemento existente para agregar archivos existentes.
26 // 6. En el futuro, para volver a abrir este proyecto, vaya a Archivo > Abrir > Proyecto y seleccione el archivo.
```

The bottom of the image shows the Visual Studio interface with the Solution Explorer on the left, the Code window in the center, and the Output window at the bottom. The Output window shows the compilation output for the program.

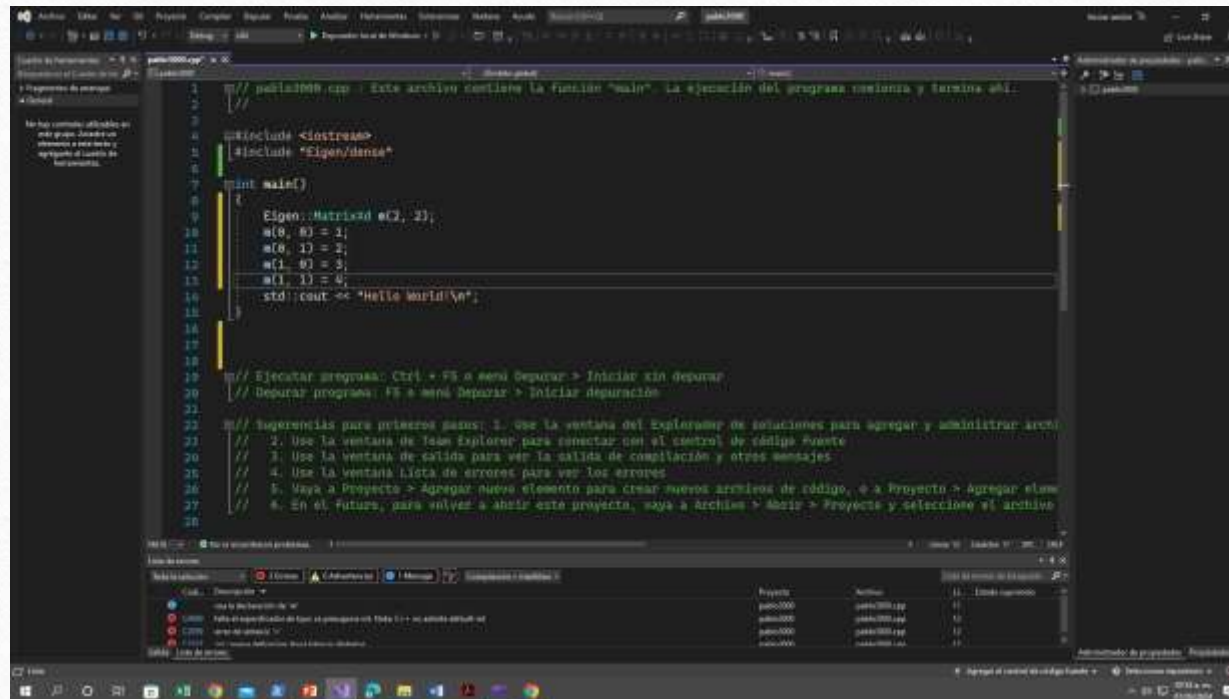
github







Clase 6

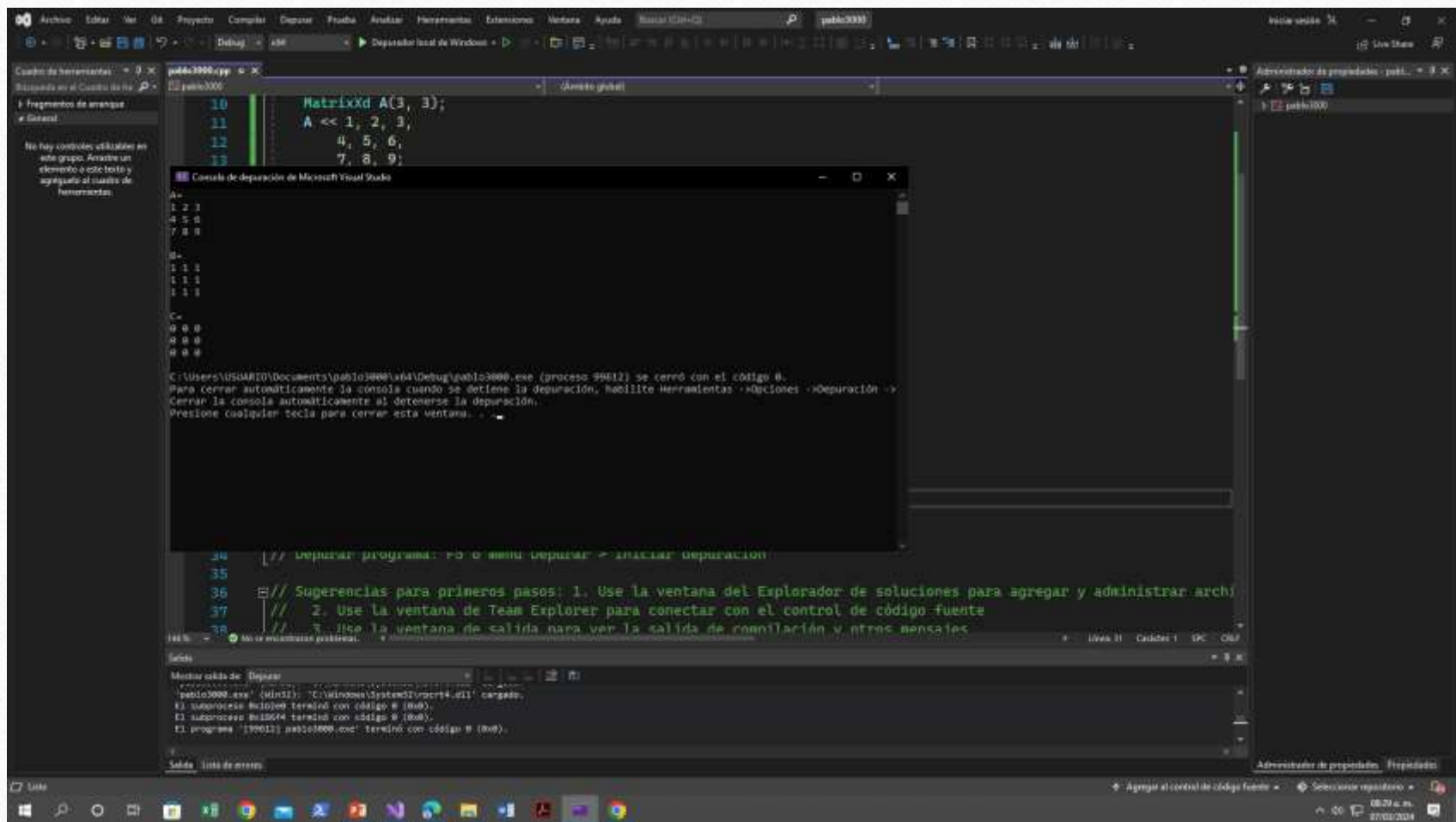


The screenshot shows a C++ IDE with a dark theme. The main editor displays the following code:

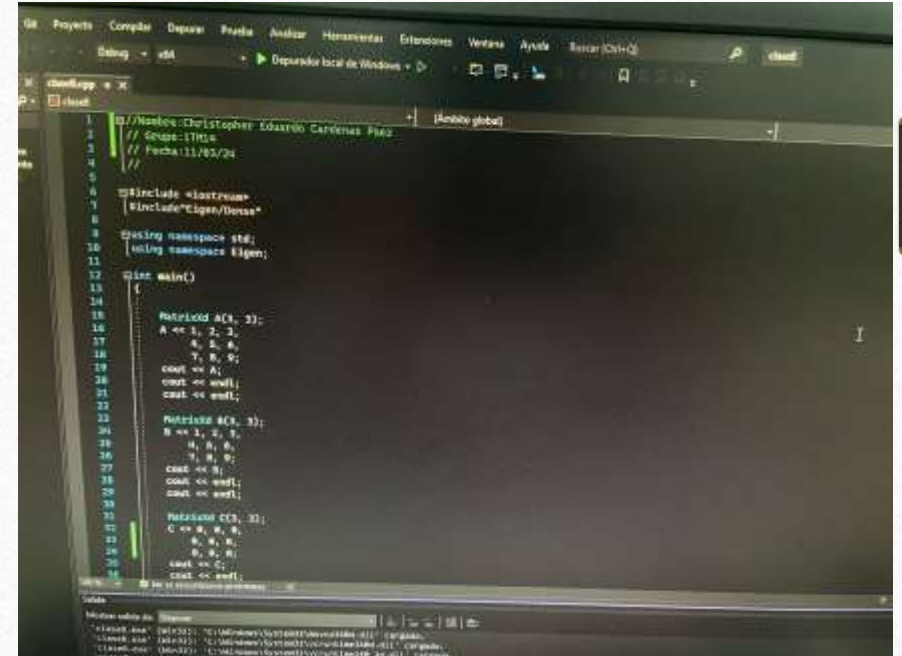
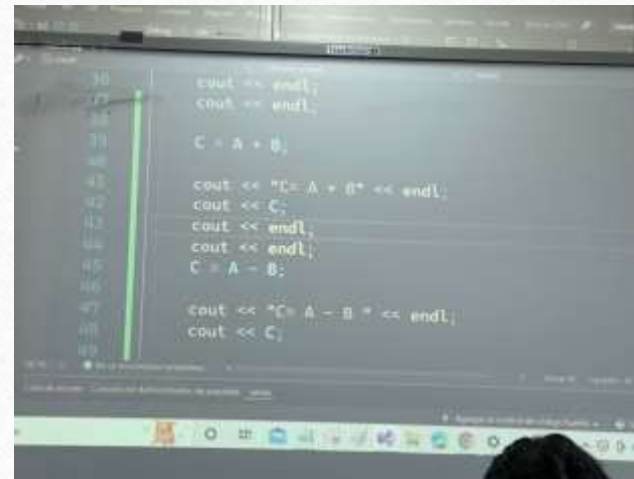
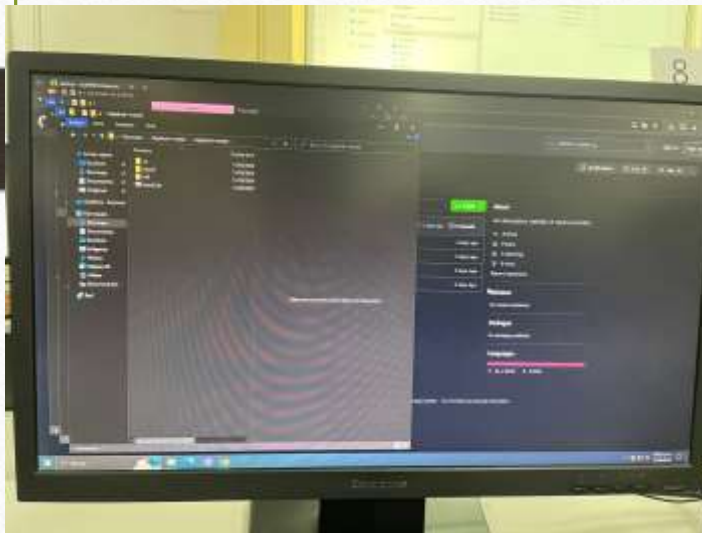
```
1 // paula3000.cpp : Este archivo contiene la función "main". La ejecución del programa comienza y termina ahí.
2
3
4 #include <iostream>
5 #include "Eigen/dense"
6
7 int main()
8 {
9     Eigen::MatrixXd m(2, 2);
10    m(0, 0) = 1;
11    m(0, 1) = 2;
12    m(1, 0) = 3;
13    m(1, 1) = 4;
14    std::cout << "Hello World!\n";
15 }
16
17
18
19 // Ejecutar programa: Ctrl + F5 o menú Depurar > Iniciar sin depurar
20 // Depurar programa: F5 o menú Depurar > Iniciar depuración
21
22 // sugerencias para primeros pasos: 1. Use la ventana del Explorador de soluciones para agregar y administrar archivos
23 // 2. Use la ventana de Team Explorer para conectar con el control de código fuente
24 // 3. Use la ventana de salida para ver la salida de compilación y otros mensajes
25 // 4. Use la ventana Lista de errores para ver los errores
26 // 5. Vaya a Proyecto > Agregar nuevo elemento para crear nuevos archivos de código, o a Proyecto > Agregar elemento
27 // 6. En el futuro, para volver a abrir este proyecto, vaya a Archivo > Abrir > Proyecto y seleccione el archivo
```

The IDE interface includes a menu bar at the top, a toolbar, and a bottom status bar. The bottom panel shows a file explorer with a project named 'paula3000' and a list of files including 'paula3000.cpp'.

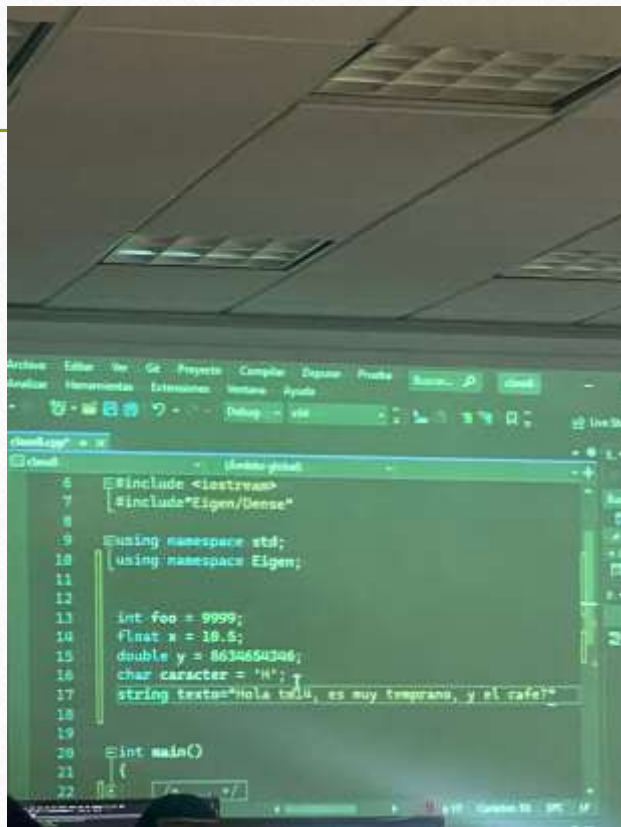
```
1 // pablo3000.cpp : Este archivo contiene la función "main", la ejecución del programa comienza y termina ahí.
2 //
3
4 #include <iostream>
5 #include "Eigen/Eigen"
6
7 int main()
8 {
9     Eigen::MatrixXd m(5, 2);
10     m(0, 0) = 1;
11     m(0, 1) = 2;
12     m(1, 0) = 3;
13     m(1, 1) = 4;
14     m(2, 0) = 5;
15     m(2, 1) = 6;
16     m(3, 0) = 7;
17     m(3, 1) = 8;
18     m(4, 0) = 9;
19     m(4, 1) = 0;
20     std::cout << m;
21 }
22
23
24 // Ejecutar programa: Ctrl + F5 o menú Depurar > Solucionar sin depurar
25 // Depurar programa: F5 o menú Depurar > Iniciar depuración
26
27 // Superficies para primeros pasos: 1. Use la ventana del Explorador de soluciones para agregar o eliminar archivos
28 // 2. Use la ventana de Team Explorer para conectar con el control de código fuente
29 // 3. Use la ventana de Salida para ver la salida de compilación y otros mensajes
30 // 4. Use la ventana Lista de errores para ver los errores
31 // 5. Vaya a Proyecto > Agregar nuevo elemento para crear nuevos archivos de código, o a Proyecto > Agregar elemento existente para agregar archivos de código existentes al proyecto
32 // 6. Ir al futuro, para volver a crear este proyecto, vaya a Archivo > Nuevo > Proyecto y seleccionar el idioma con
```

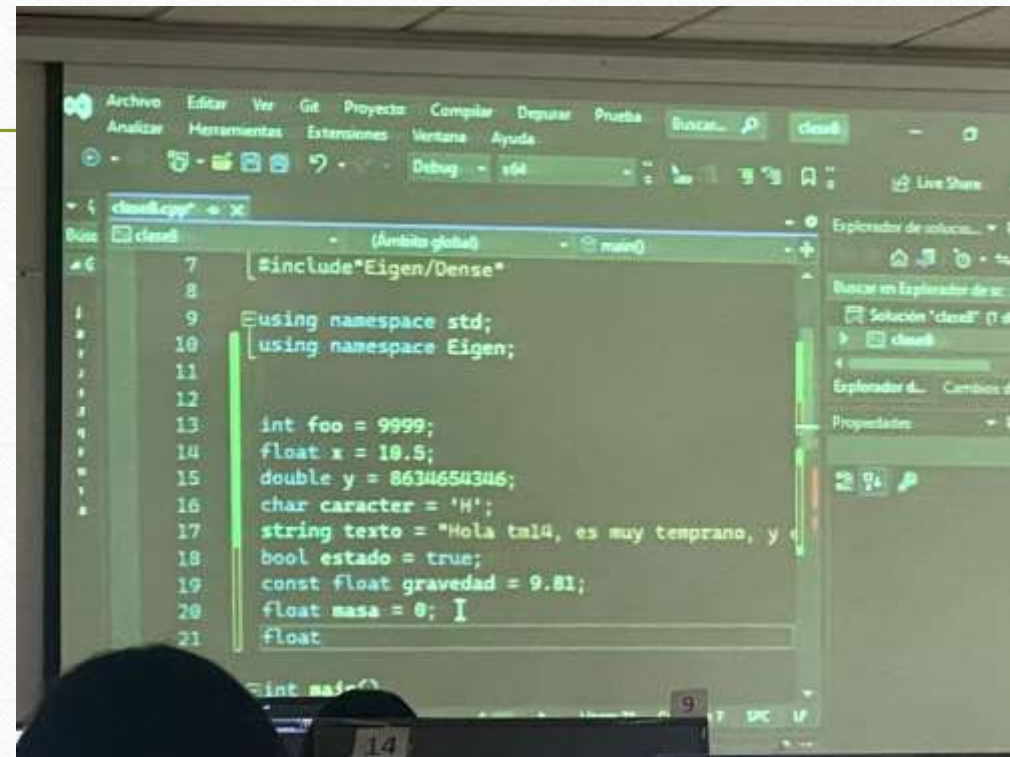
Clase 8



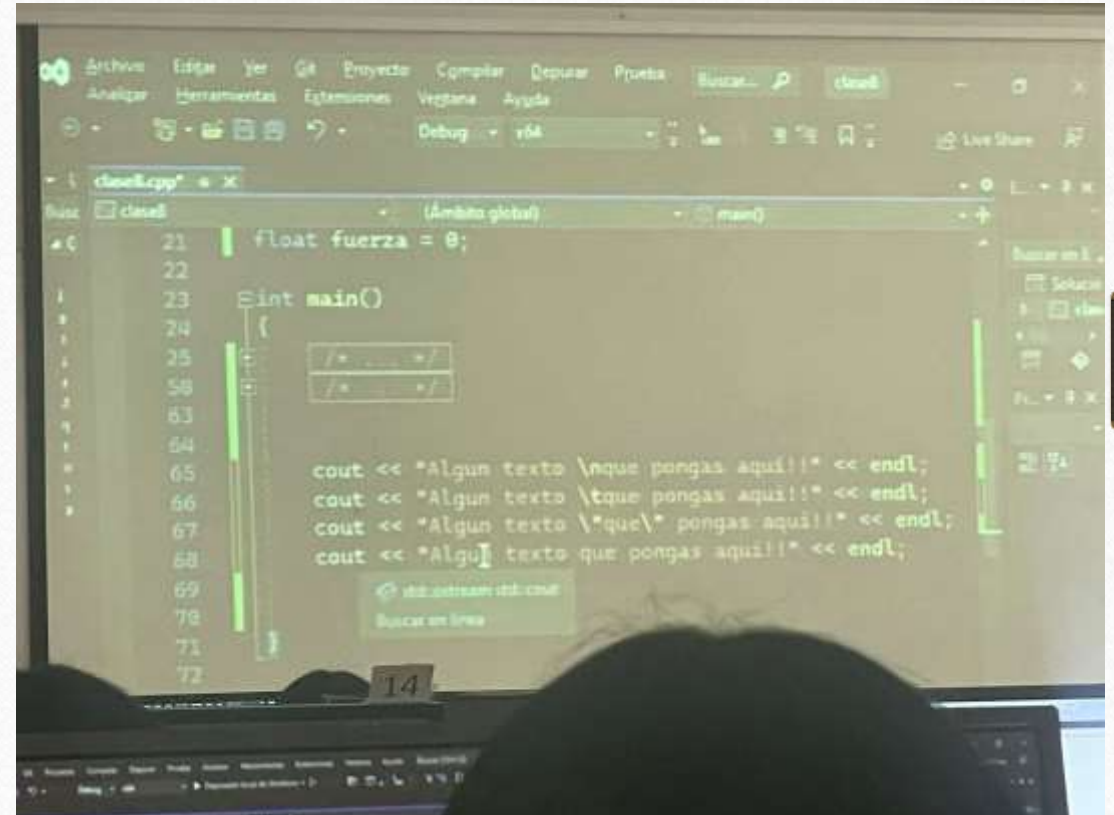
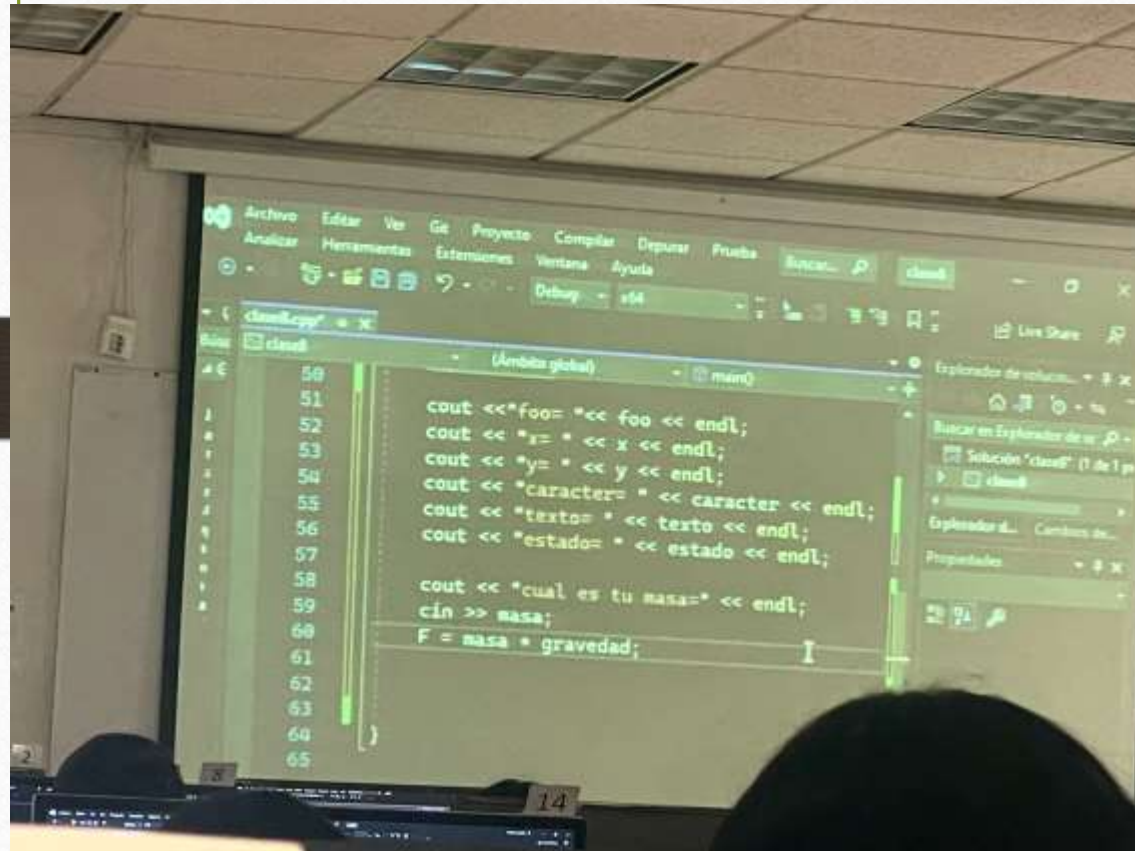
Clase 9

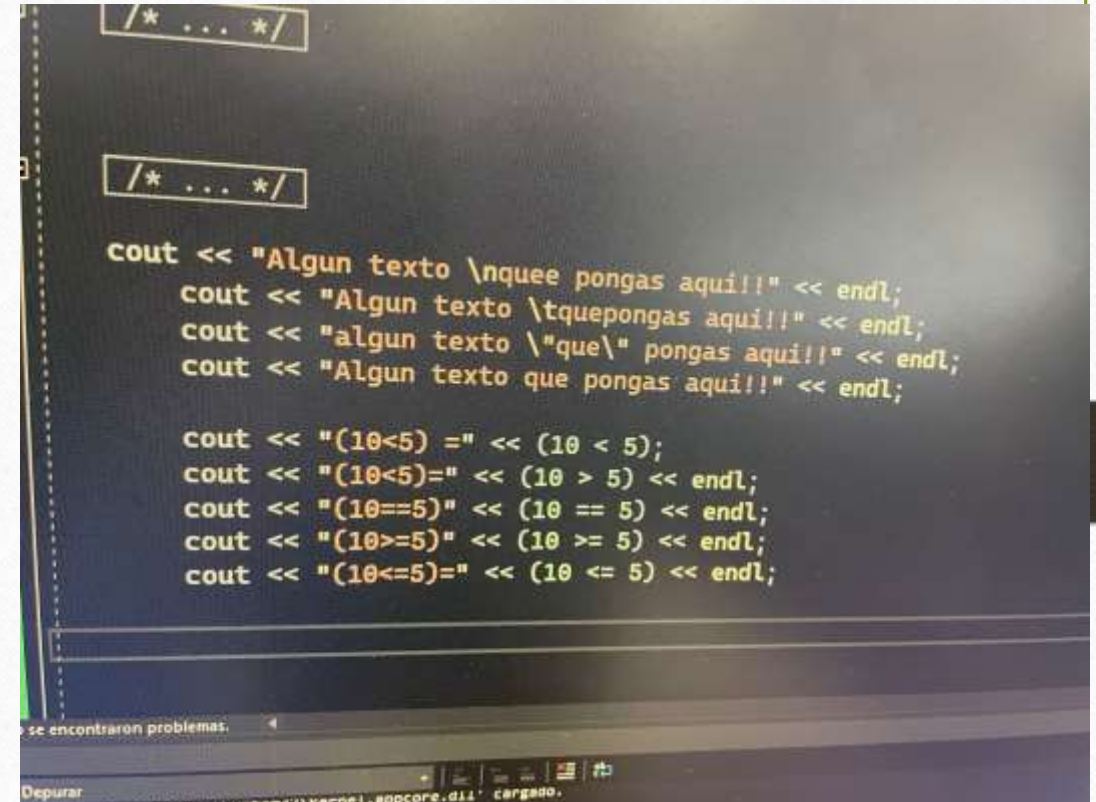
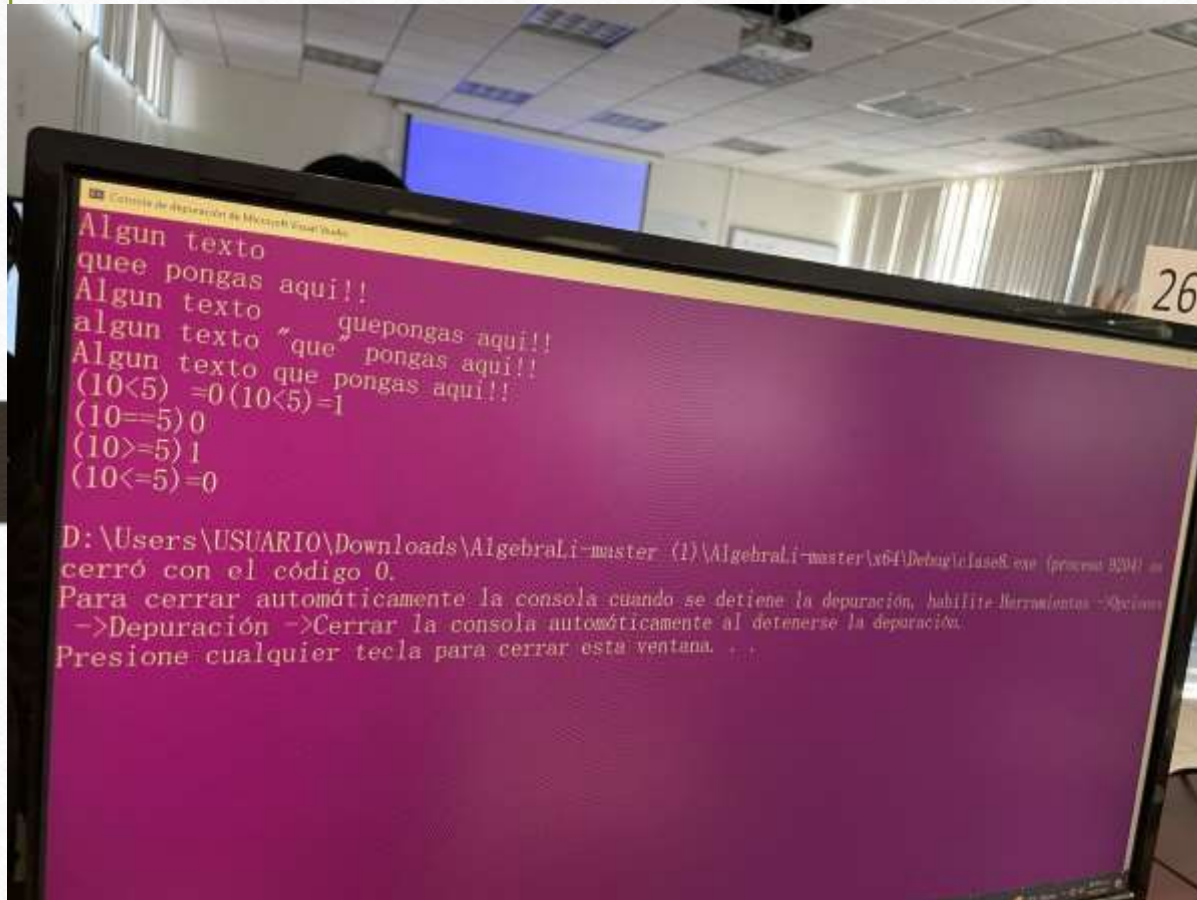


```
1 #include <iostream>
2 #include "Eigen/Dense"
3
4 using namespace std;
5 using namespace Eigen;
6
7 int foo = 9999;
8 float x = 10.5;
9 double y = 8634654346;
10 char caracter = 'H';
11 string texto = "Hola tm14, es muy temprano, y el cafe?";
12
13 int main()
14 {
15 }
```



```
1 #include "Eigen/Dense"
2
3 using namespace std;
4 using namespace Eigen;
5
6 int foo = 9999;
7 float x = 10.5;
8 double y = 8634654346;
9 char caracter = 'H';
10 string texto = "Hola tm14, es muy temprano, y el cafe?";
11 bool estado = true;
12 const float gravedad = 9.81;
13 float masa = 0;
14 float
15
16 int main()
17 {
18 }
```





```

7  #include <iostream>
8  #include "Eigen/Dense"
9
10 using namespace std;
11 using namespace Eigen;
12
13 int foo = 9999;
14 float x = 10.5;
15 double y = 8634654346;
16 char character = 'H';
17 string texto = "Hola tm14";
18 bool estado = true;
19
20 /*
21  const float gravedad = 9.81;
22  double masa = 0;
23  bool respuesta = true;
24  */
25
26 int main()
27 {
28
29  /*

```

```

6  #include <iostream>
7  #include "Eigen/Dense"
8
9
10 using namespace std;
11 using namespace Eigen;
12
13 int foo = 9999;
14 float x = 10.5;
15 double y = 8634654346;
16 char character = 'H';
17 string texto = "Hola tm14";
18 bool estado = true;
19
20 /*
21  const float gravedad = 9.81;
22  double masa = 0;
23  bool respuesta = true;
24  */
25
26 int main()
27 {

```