Thynk Unlimited

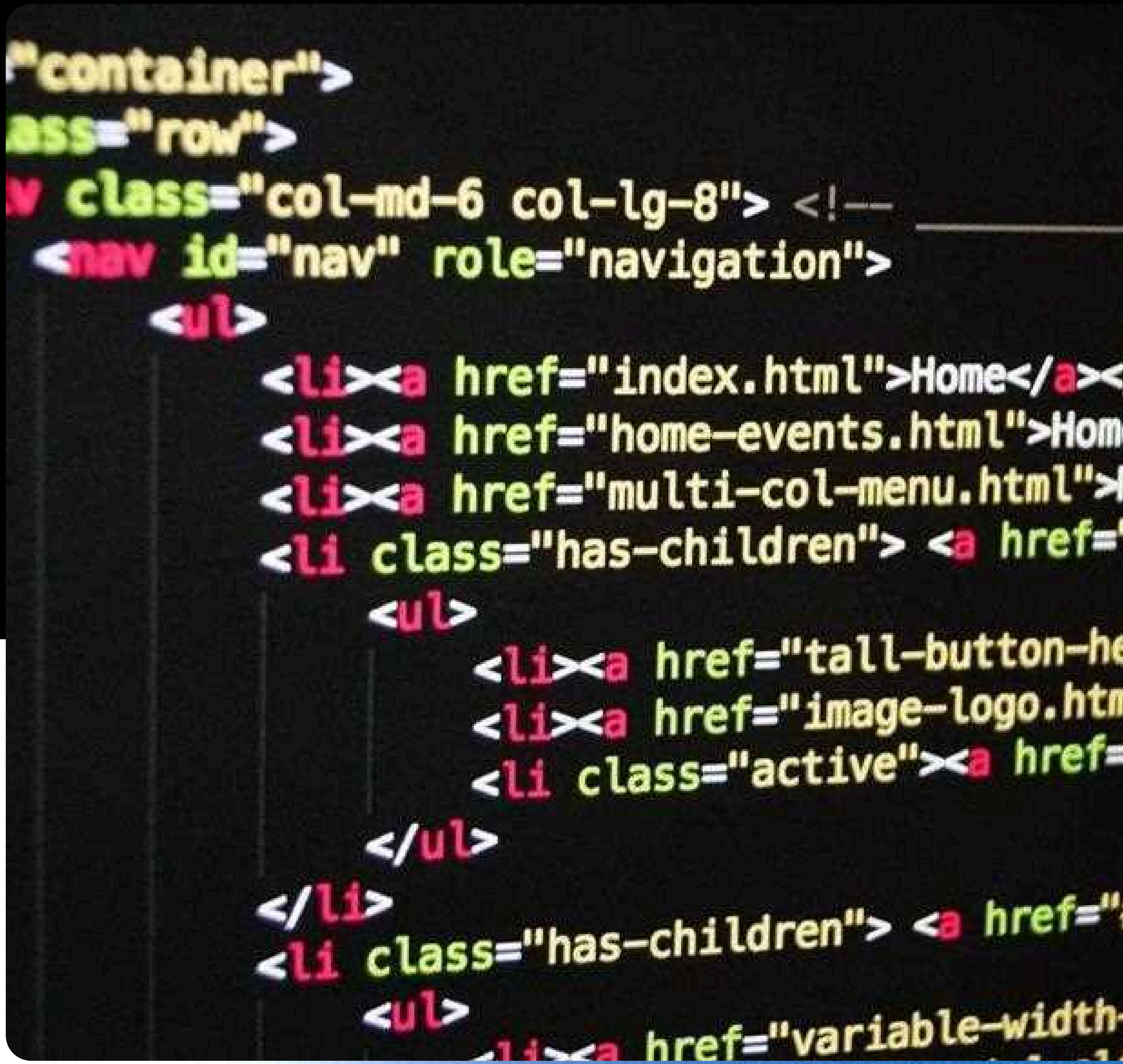
INTRODUCTION TO COMPUTER PROGRAMMING AND JAVA

BY:
OCHIENG BOSTONE

Thynk Unlimited

What is Computer Programming?

- Process of giving machines instructions
- Describes what the computer should do
- Translates human intentions into machine actions

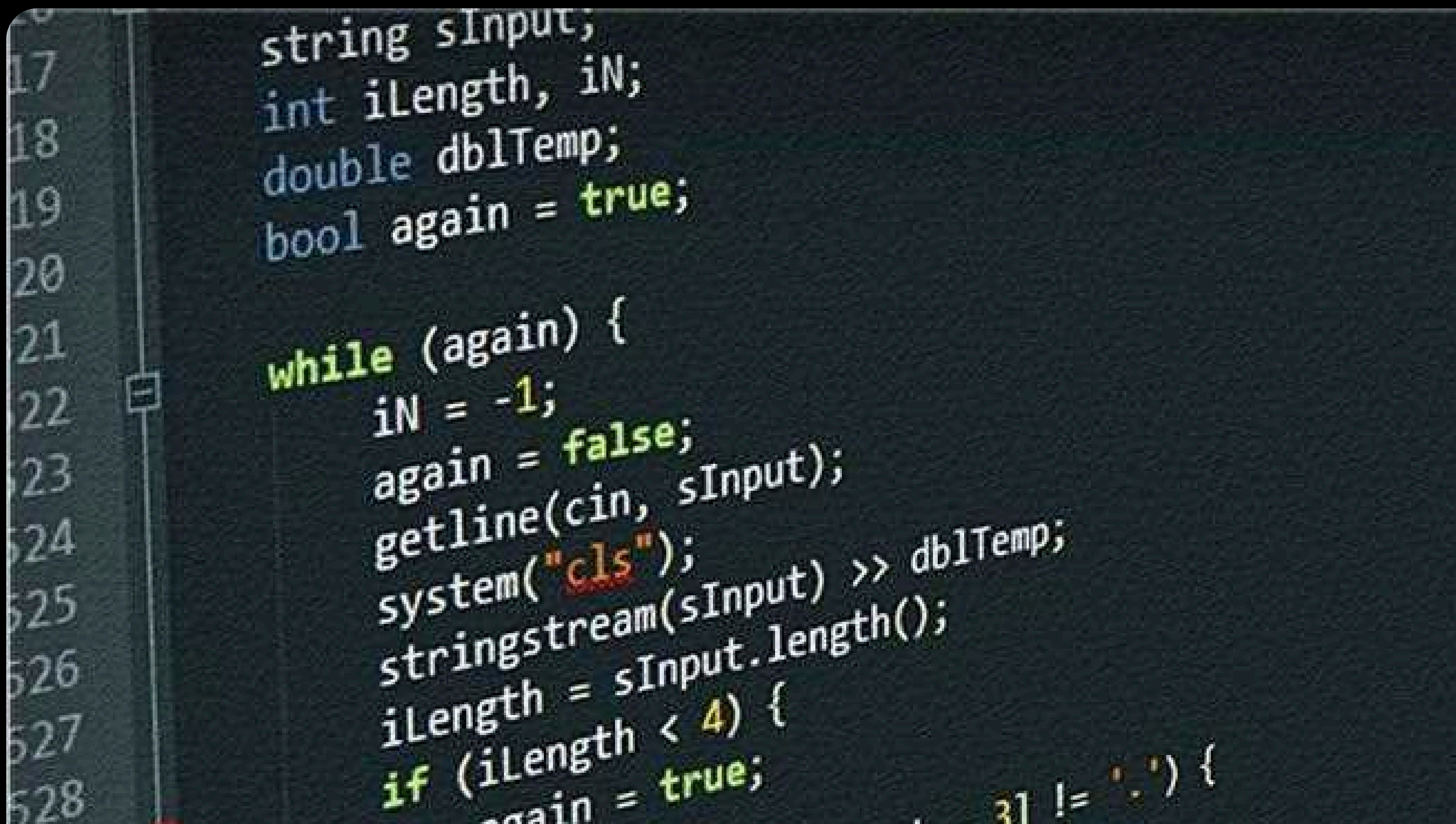


```
"container">
ass="row">
v class="col-md-6 col-lg-8"> <!--
 <nav id="nav" role="navigation">
     <ul>
         <li><a href="index.html">Home</a><
         <li><a href="home-events.html">Hom
         <li><a href="multi-col-menu.html">
         <li class="has-children"> <a href=
             <ul>
                 <li><a href="tall-button-he
                 <li><a href="image-logo.htm
                 <li class="active"><a href=
             </ul>
         </li>
         <li class="has-children"> <a href="
             <ul>
                 <li><a href="variable-width-
```

History of Computer Programming

BUILDING IN SPRINTS

- Early computing for calculations and automation
- Evolution from binary to high-level languages
- Growth in complexity and user accessibility



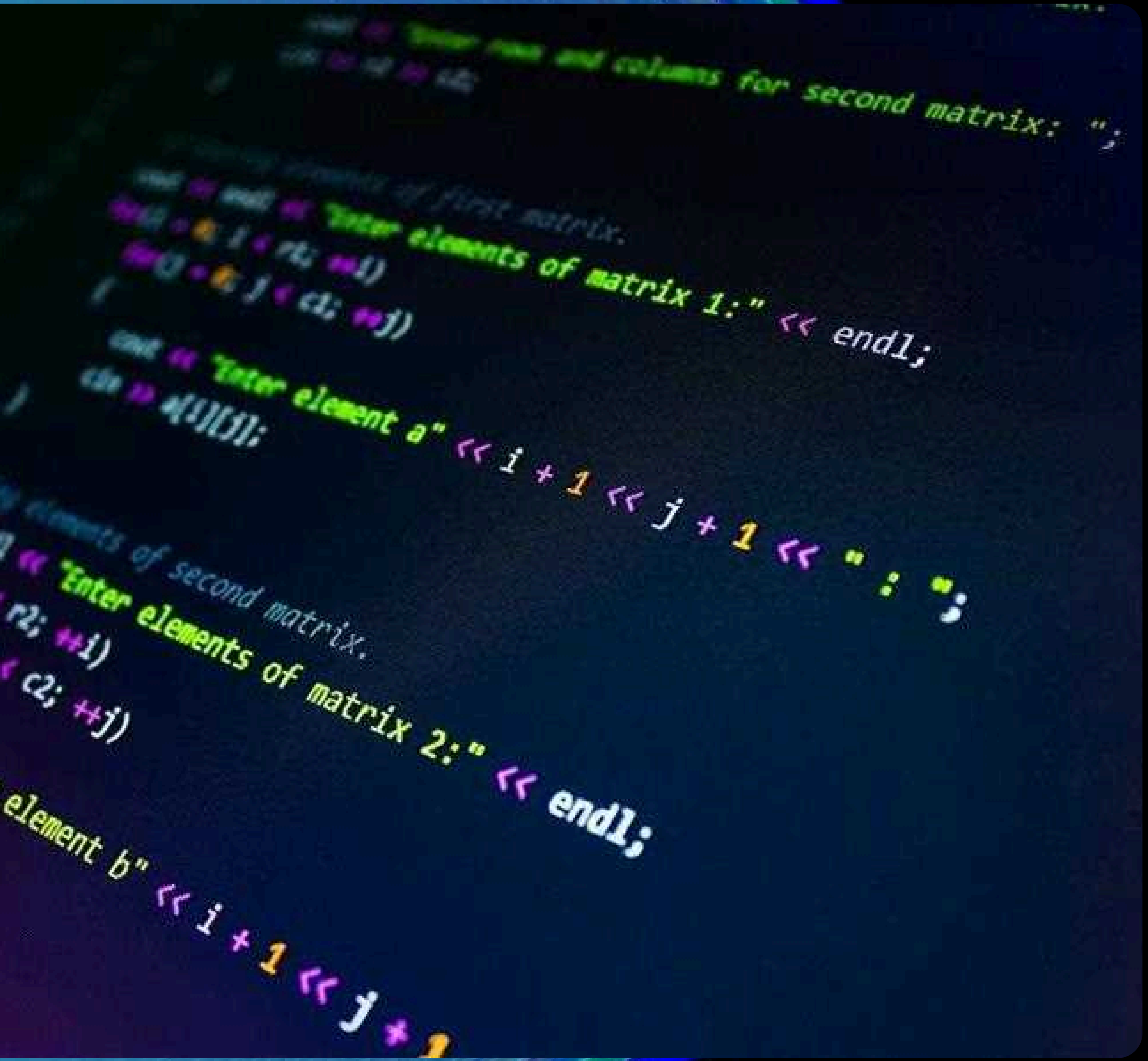
```
    string sInput;
    int iLength, iN;
    double dblTemp;
    bool again = true;

    while (again) {
        iN = -1;
        again = false;
        getline(cin, sInput);
        system("cls");
        stringstream(sInput) >> dblTemp;
        iLength = sInput.length();
        if (iLength < 4) {
            ain = true;
                                  3] != '.') {
```

```
Shoulda::Matchers.configure do |config|
  config.integrate do |with|
    with.test_framework :rspec
    with.library :rails
  end
end
# Add additional requires below this line
```

Why Learn Computer Programming?

- Essential skill in modern world
- Enables problem-solving and critical thinking
- Opens opportunities in various industries

Binary to High-Level Languages



```
"Enter rows and columns for second matrix: ";
cin >> r2 >> c2;

// Storing elements of first matrix.
cout << endl << "Enter elements of matrix 1:" << endl;
for(i = 0; i < r1; ++i)
for(j = 0; j < c1; ++j)
{
cout << "Enter element a" << i + 1 << j + 1 << " : ";
cin >> a[i][j];
}

elements of second matrix.
<< "Enter elements of matrix 2:" << endl;
r2; ++i)
c2; ++j)

element b" << i + 1 << j + 1
```

- Computers understand binary (0s and 1s)
- High-level languages are human-readable
- Compilers and interpreters translate to binary

Modern Applications of Programming

PROGRAMMING FOR INSIGHTS

- Research and Data Analysis
- Government Digital Services
- Web Development and Design
- Cybersecurity Solutions

```
sp_iter = self.stub.Get
atuses = {}
nc for data in resp_iter:
 status = Status(
     status_id=data.id, name
)
```

Programming in Daily Life

- Smartphones and Apps
- Online Shopping and Banking
- Social Media and Communication

Popular Programming Languages

- Python: Versatile and easy to learn
- Java: Platform-independent and widely used
- JavaScript: Web development and interactive content

Thynk Unlimited

Introduction to Java

- Released by Sun Microsystems in 1995
- Portable, secure, and robust
- 'Write Once, Run Everywhere' philosophy

Defining the Craft of Coding

Thynk Unlimited

Why Java?

From Early Algorithms to Modern

- Simple & Easy to Learn: Syntax is straightforward and readable
- Platform Independent (Write Once, Run Anywhere - WORA): Runs on any device with JVM
- Object-Oriented: Follows principles of encapsulation, inheritance, polymorphism, and abstraction
- Secure and Robust: Java includes runtime checking and strong memory management
- Multithreaded & High Performance: Supports parallel execution of tasks

Java Virtual Machine (JVM)

- •Abstract Machine: Enables cross-platform execution by converting bytecode to native machine code.
- Runtime Environment: Manages memory, security, and error handling during program execution.
- Description: Breaks down the role of the JVM, helping students understand platform independence.

Bytecode and Compilation

Working Together in Code

- Java code is compiled to bytecode
- Bytecode is platform-independent
- Executed by the JVM on any OS

NEXT SLIDE

JAVA RUNTIME ENVIRONMENT (JRE)

Definition: JRE is a software package containing the JVM and class libraries required to run Java applications.

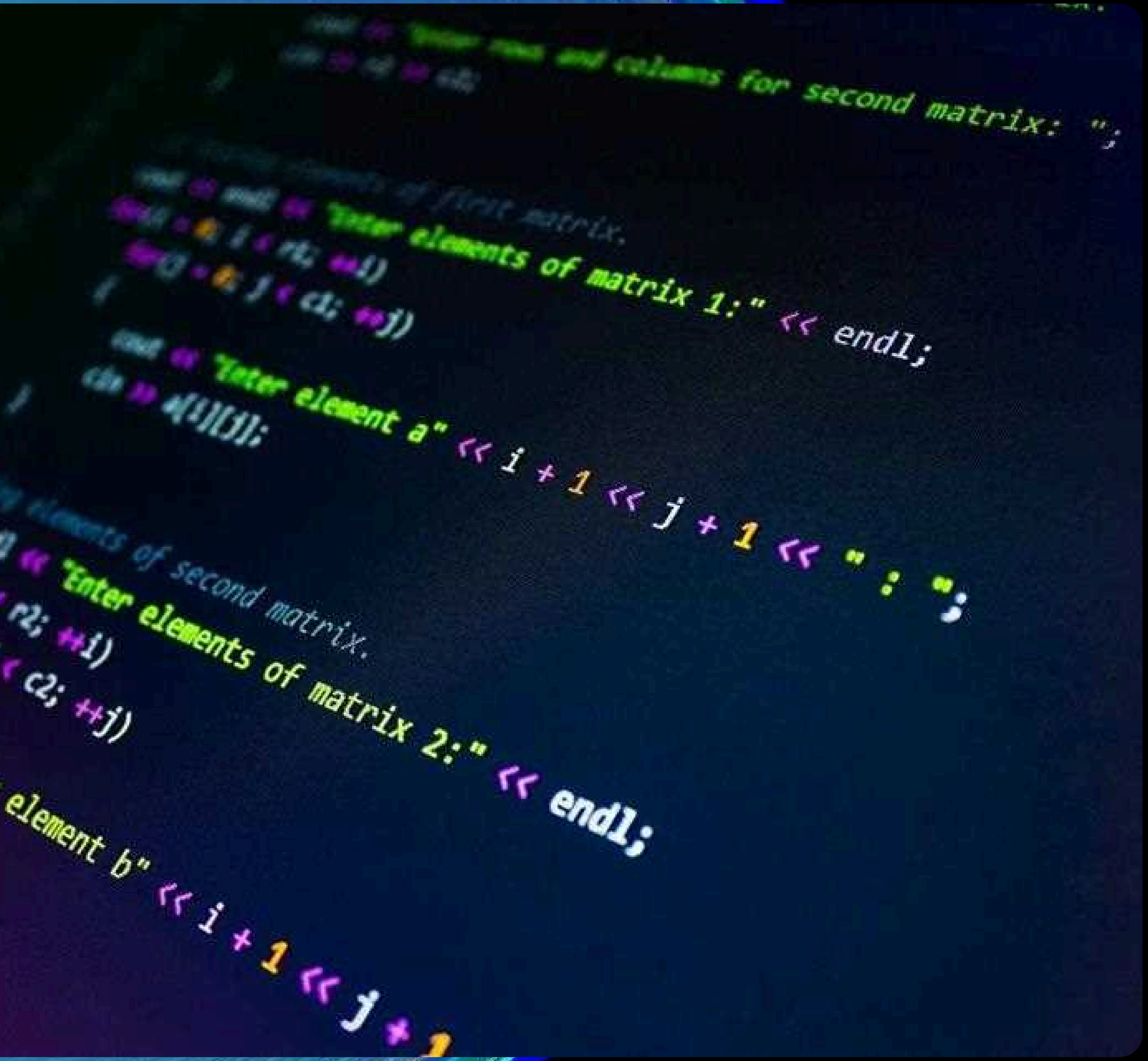
- Purpose: Required for running (but not developing) Java applications on any platform.
- Description: Differentiates between JRE and JDK, clarifying when each is needed.

```
cout << "Enter rows and columns for second matrix: ";
[illegible]

// Storing elements of first matrix.
cout << endl << "Enter elements of matrix 1:" << endl;
for(i = 0; i < r1; ++i)
for(j = 0; j < c1; ++j)
{
    cout << "Enter element a" << i + 1 << j + 1 << " : ";
    cin >> a[i][j];
}
// Storing elements of second matrix.
endl << "Enter elements of matrix 2:" << endl;
i < r2; ++i)
j < c2; ++j)
er element b" << i + 1 << j + 1
```

Java Development Kit (JDK)

Definition: Complete development kit including JRE, compiler, and debugging tools.

- Purpose: Required for writing, compiling, and debugging Java programs.
- Description: Introduces the JDK as the main toolkit for Java developers.



```
for second matrix: ";
"Enter elements of matrix 1:" << endl;
"Enter element a" << i + 1 << j + 1 << " : ";
elements of second matrix.
"Enter elements of matrix 2:" << endl;
element b" << i + 1 << j + 1
```

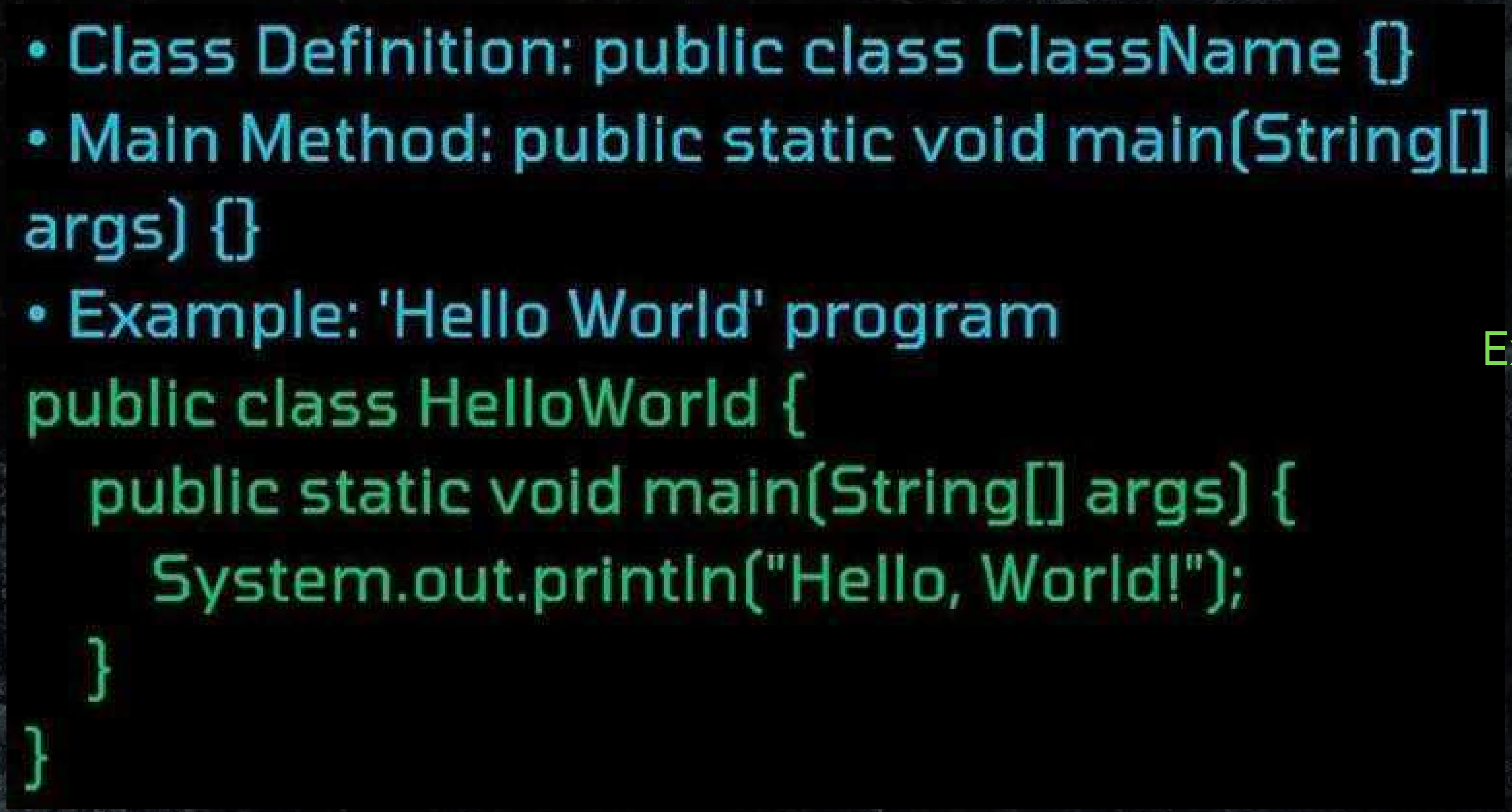
Object-Oriented Programming (OOP)

- Organizes code into objects
- Promotes modular and reusable design
- Fundamental paradigm in Java

SETTING UP JAVA DEVELOPMENT ENVIRONMENT

- Download and install Java Development Kit (JDK) from Oracle or OpenJDK
- Install an IDE: IntelliJ IDEA, Eclipse, NetBeans, or VS Code for writing Java programs
- Configure Environment Variables: Set PATH and JAVA_HOME for command-line execution
- Verify installation: Run java -version and javac -version in the terminal

JAVA CLASS STRUCTURE

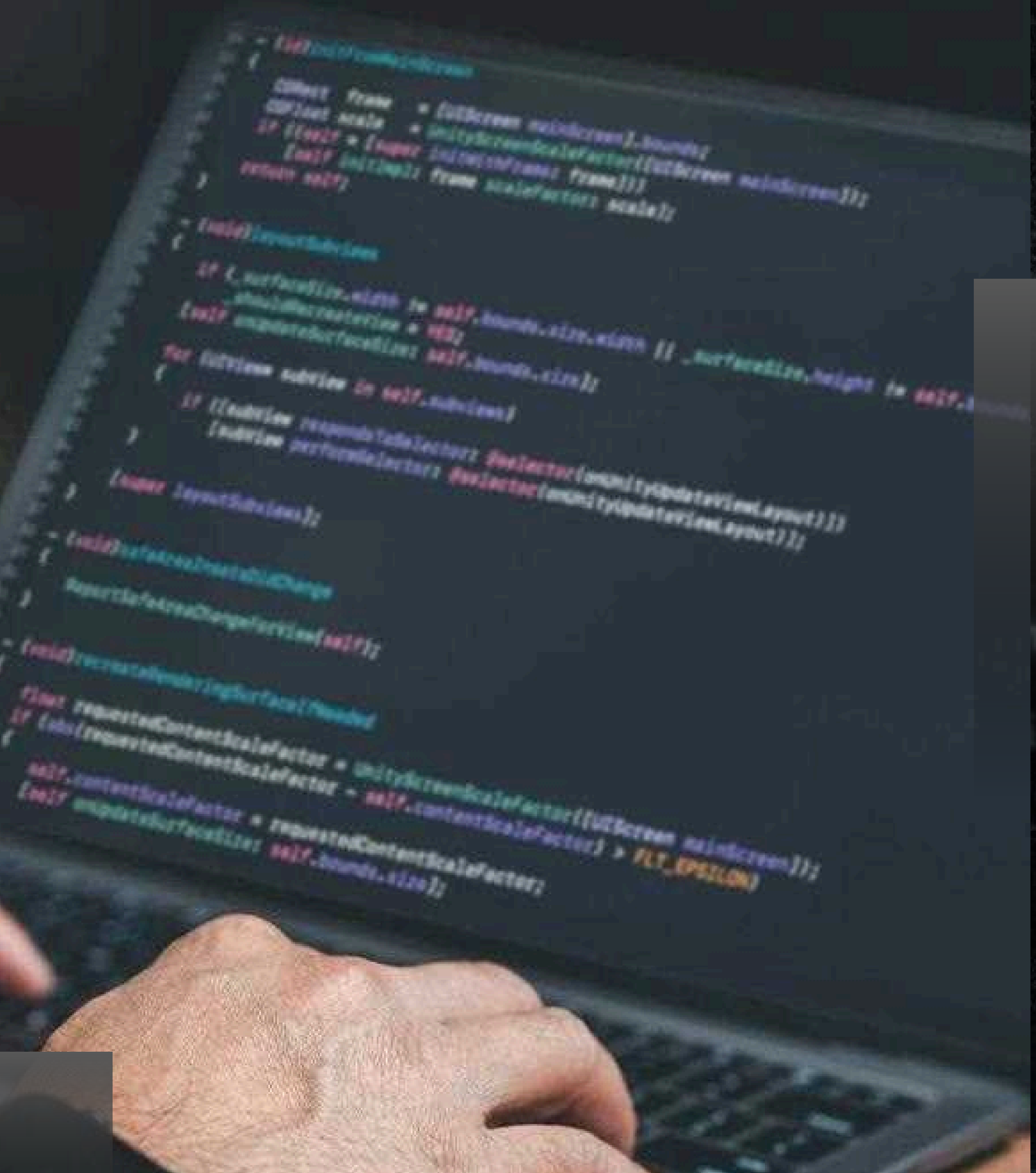


- Class Definition: public class ClassName {}
- Main Method: public static void main(String[] args) {}
- Example: 'Hello World' program

```java
public class HelloWorld {
   public static void main(String[] args) {
      System.out.println("Hello, World!");
   }
}
```

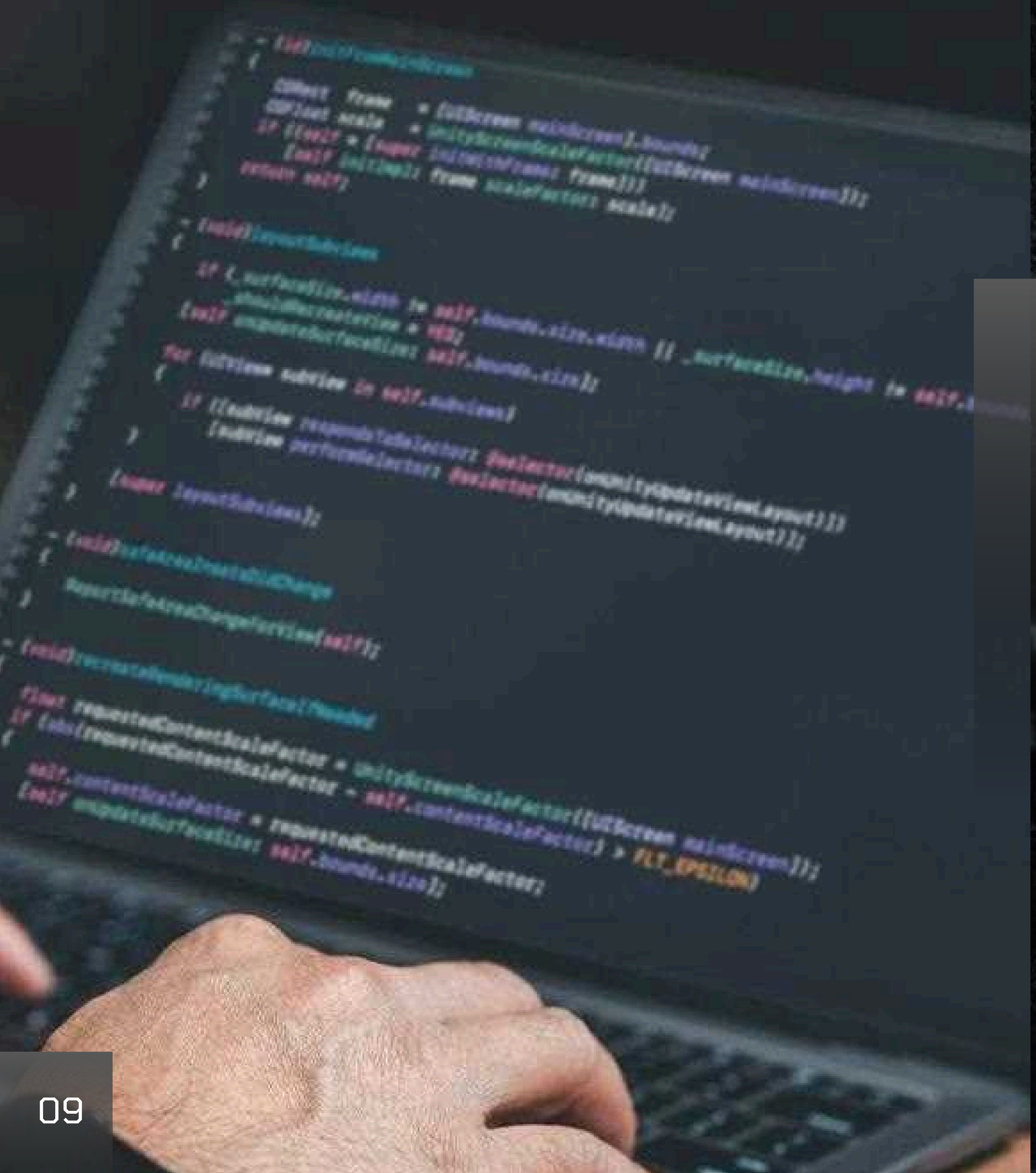
Explanation of the Code

- public class HelloWorld – Declares a class named HelloWorld
- public static void main(String[] args) – Entry point of the program
- System.out.println("Hello, World!"); – Prints text to the console



COMPILING AND RUNNING JAVA PROGRAMS

- Compilation: javac ClassName.java
- Execution: java ClassName
- Bytecode (.class) is executed by the JVM



VARIABLES IN JAVA

- A variable is a container for storing data values
- Java has different types of variables:
  - int: Stores integers (e.g., 10, 20)
  - double: Stores floating-point numbers (e.g., 5.99)
  - char: Stores single characters (e.g., 'A')
  - boolean: Stores true/false values

```
int age = 25;
double price = 19.99;
char grade = 'A';
boolean isStudent = true;
```

DATA TYPES AND JAVA OPERATORS

- Primitive Data Types: int, float, double, char, boolean, etc.
- Non-Primitive Data Types: Strings, Arrays, Objects, etc.
- Arithmetic Operators: +, -, *, /, %
- Comparison Operators: ==, !=, >, <, >=, <=
- Logical Operators: &&, ||, !

CONTROL STATEMENTS

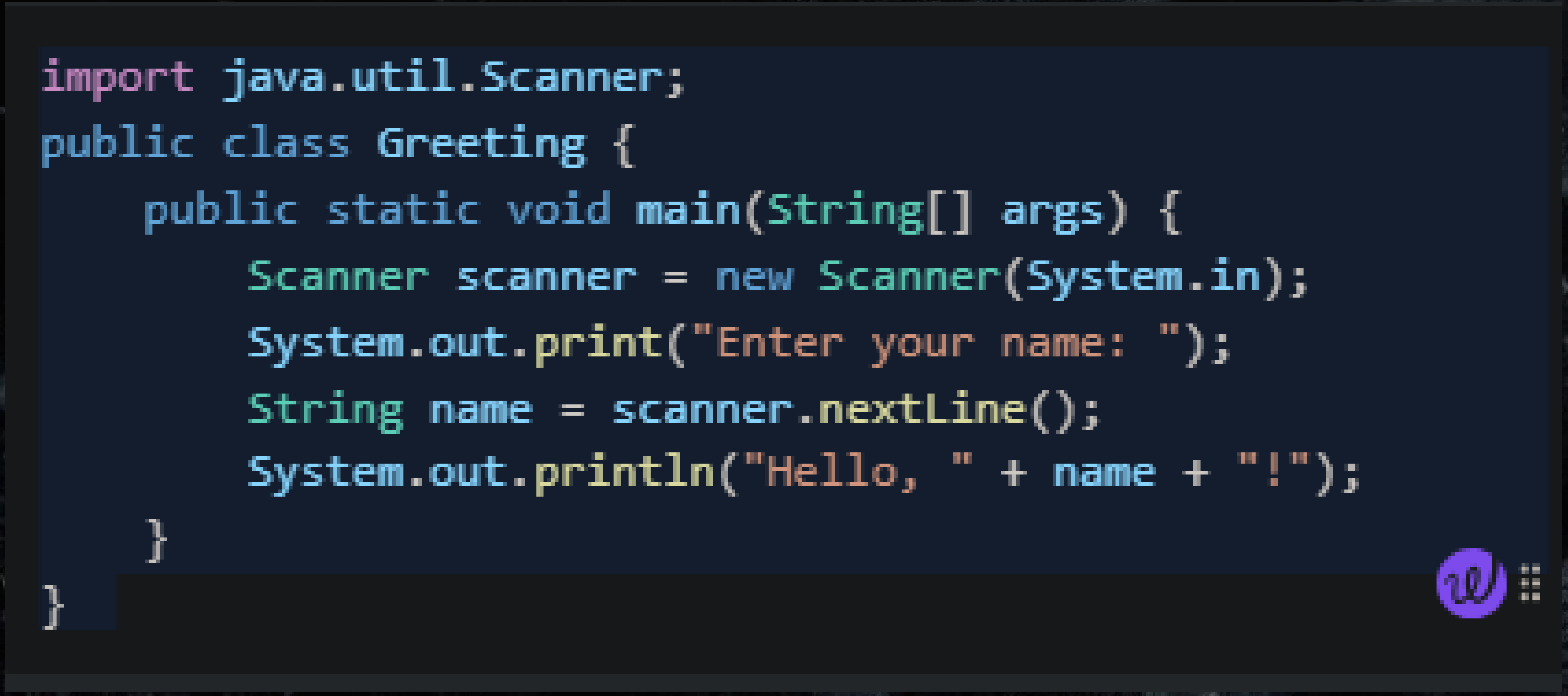
- Conditional Statements: if, else, switch
- Loops: for, while, do-while
- Break and Continue statements

```
public class IfElseExample {
    public static void main(String[] args) {
        int number = 10;

        if (number > 0) {
            System.out.println("The number is positive.");
        } else if (number < 0) {
            System.out.println("The number is negative.");
        } else {
            System.out.println("The number is zero.");
        }
    }
}
```

```
public class ForLoopExample {
    public static void main(String[] args) {
        for (int i = 1; i <= 5; i++) {
            System.out.println("Count: " + i);
        }
    }
}
```

```
public class WhileLoopExample {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 5) {
            System.out.println("Number: " + i);
            i++;
        }
    }
}
aph text
```

WRITING A SIMPLE JAVA PROGRAM



```java
import java.util.Scanner;
public class Greeting {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter your name: ");
        String name = scanner.nextLine();
        System.out.println("Hello, " + name + "!");
    }
}
```

THANK YOU

THANK YOU
THANK YOU