FUNDAMENTALS OF THE JAVA PROGRAMMING LANGUAGE

2

Understand Java terminology and environment

Edited By : Hazleena Binti Osman

Lesson Learning Outcome

- Understand Java terminology and environment
    - ❖ Describe the history of Java technology.
    - ❖ Identify the features of Java programming language.
    - ❖ State anatomy of the Java programs:
        a. comment
        b. reserved word
        c. modifiers
        d. statements
        e. blocks
        f. classes
        g. method
        h. main method

History of Java

Year	History
1990	C++ was found not fit to control electronic devices. So research for a new programming language started.
1991-92	A new programming language OAK was found. Later renamed as JAVA.
1993-94	Java became a perfect language to develop Internet-based applications
1995	Sun Microsystems introduced Java-enabled Web browser “Hot Java”.

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Edited By : Hazleena Binti Osman

Characteristics of Java

- **Java Is Simple**
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Java is partially modeled on C++, but greatly simplified and improved. Some people refer to Java as "C++--" because it is like C++ but with more functionality and fewer negative aspects.

Edited By : Hazleena Binti Osman

Characteristics of Java

- Java Is Simple
- **Java Is Object-Oriented**
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Java is inherently object-oriented. Although many object-oriented languages began strictly as procedural languages, Java was designed from the start to be object-oriented. Object-oriented programming (OOP) is a popular programming approach that is replacing traditional procedural programming techniques.

One of the central issues in software development is how to reuse code. Object-oriented programming provides great flexibility, modularity, clarity, and reusability through encapsulation, inheritance, and polymorphism.

Edited By : Hazleena Binti Osman

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- **Java Is Distributed**
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Distributed computing involves several computers working together on a network. Java is designed to make distributed computing easy. Since networking capability is inherently integrated into Java, writing network programs is like sending and receiving data to and from a file.

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- **Java Is Interpreted**
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

You need an interpreter to run Java programs. The programs are compiled into the Java Virtual Machine code called bytecode. The bytecode is machine-independent and can run on any machine that has a Java interpreter, which is part of the Java Virtual Machine (JVM).

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- **Java Is Robust**
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Java compilers can detect many problems that would first show up at execution time in other languages.

Java has eliminated certain types of error-prone programming constructs found in other languages.

Java has a runtime exception-handling feature to provide programming support for robustness.

Edited By : Hazleena Binti Osman

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- **Java Is Secure**
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Java implements several security mechanisms to protect your system against harm caused by stray programs.

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- **Java Is Architecture-Neutral**
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Write once, run anywhere

With a Java Virtual Machine (JVM), you can write one program that will run on any platform.

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- **Java Is Portable**
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

Edited By : Hazleena Binti Osman

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- **Java's Performance**
- Java Is Multithreaded
- Java Is Dynamic

Java's performance Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- **Java Is Multithreaded**
- Java Is Dynamic

Multithread programming is smoothly integrated in Java, whereas in other languages you have to call procedures specific to the operating system to enable multithreading.

Edited By : Hazleena Binti Osman

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- **Java Is Dynamic**

Java was designed to adapt to an evolving environment. New code can be loaded on the fly without recompilation. There is no need for developers to create, and for users to install, major new software versions. New features can be incorporated transparently as needed.

Edited By : Hazleena Binti Osman

JDK Editions

- Java Standard Edition (J2SE)
  - J2SE can be used to develop client-side standalone applications or applets.
- Java Enterprise Edition (J2EE)
  - J2EE can be used to develop server-side applications such as Java servlets and Java ServerPages.
- Java Micro Edition (J2ME).
  - J2ME can be used to develop applications for mobile devices such as cell phones.

Java IDE Tools

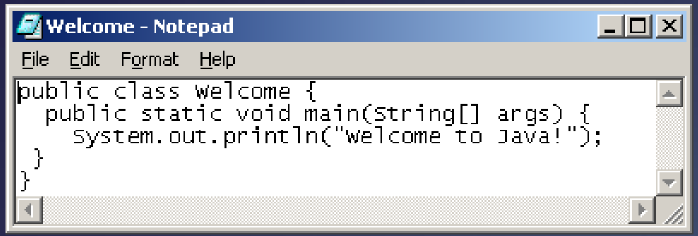
- JCreator
- Borland JBuilder
- NetBeans Open Source by Sun
- Sun ONE Studio by Sun MicroSystems
- Eclipse Open Source by IBM

Edited By : Hazleena Binti Osman

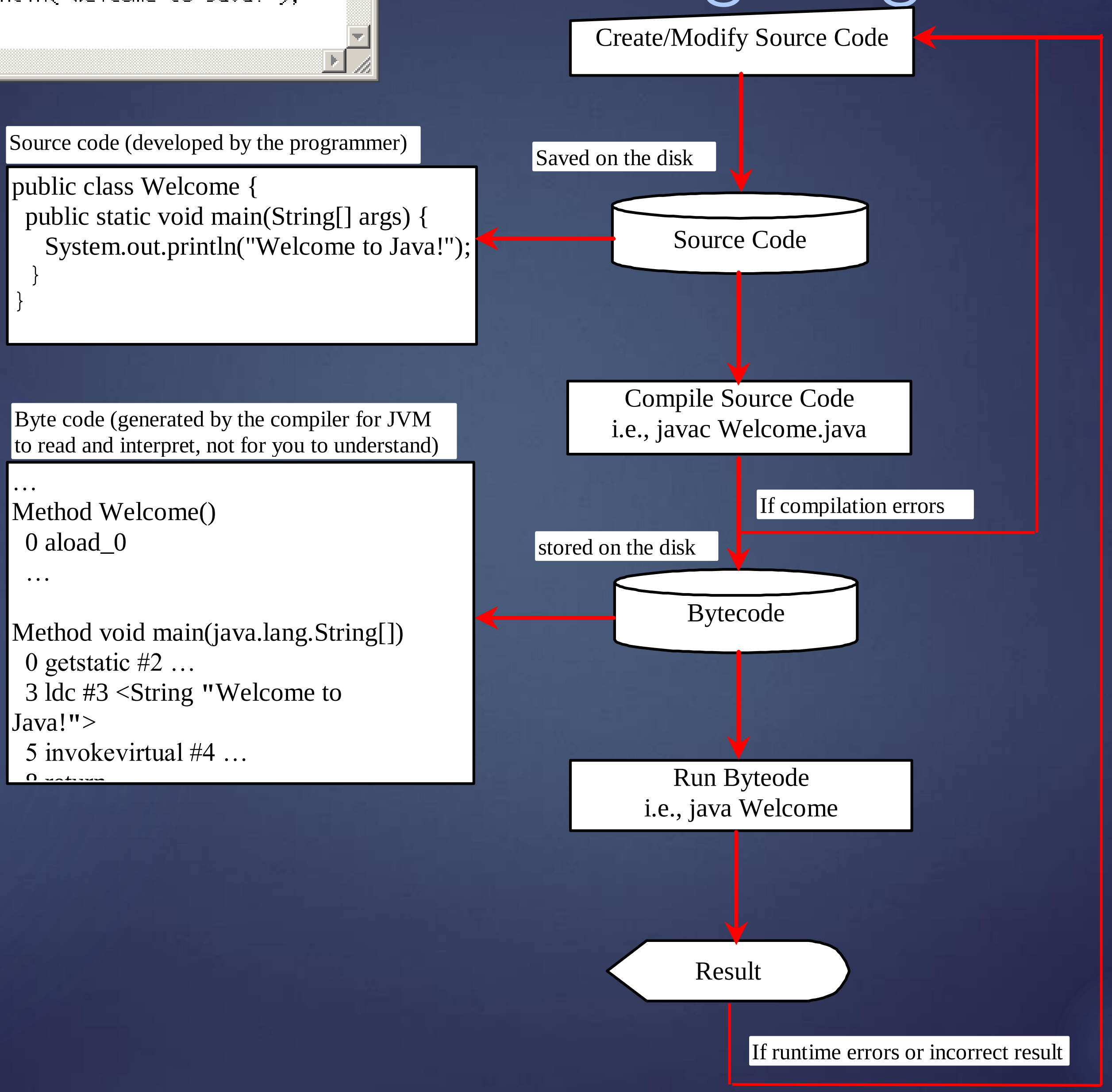
A Simple Java Program

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Creating, Compiling, and Running Programs



```
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```



Source code (developed by the programmer)

```
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Byte code (generated by the compiler for JVM to read and interpret, not for you to understand)

```
…
Method Welcome()
  0 aload_0
  …

Method void main(java.lang.String[])
  0 getstatic #2 …
  3 ldc #3 <String "Welcome to
Java!">
  5 invokevirtual #4 …
```

Edited By : Hazleena Binti Osman

animation

Trace a Program Execution

Enter main method

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Edited By : Hazleena Binti Osman

animation

Trace a Program Execution

Execute statement

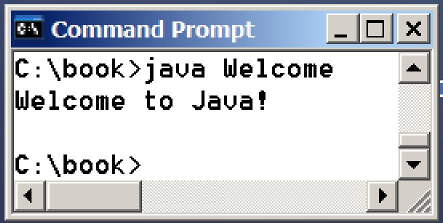
```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Edited By : Hazleena Binti Osman

animation

Trace a Program Execution

```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```



```
C:\book>java Welcome
Welcome to Java!

C:\book>
```

print a message to the console

Edited By : Hazleena Binti Osman

Anatomy of a Java Program

- Comments
- Package
- Reserved words
- Modifiers
- Statements
- Blocks
- Classes
- Methods
- The main method

Edited By : Hazleena Binti Osman

Comments

In Java, comments are preceded by two slashes

(//) in a line, or enclosed between /* and */ in one or multiple lines.

When the compiler sees //, it ignores all text after // in the same line.

When it sees /*, it scans for the next */ and ignores any text between /* and */.

Edited By : Hazleena Binti Osman

Reserved Words

Reserved words or keywords are words that have a specific meaning to the compiler and cannot be used for other purposes in the program. For example, when the compiler sees the word class, it understands that the word after class is the name for the class. Other reserved words in Listing 1.1 are public, static, and void. Their use will be introduced later in the book.

Edited By : Hazleena Binti Osman

Modifiers

Java uses certain reserved words called modifiers that specify the properties of the data, methods, and classes and how they can be used.

Examples of modifiers are public and static. Other modifiers are private, final, abstract, and protected.

Edited By : Hazleena Binti Osman

Statements

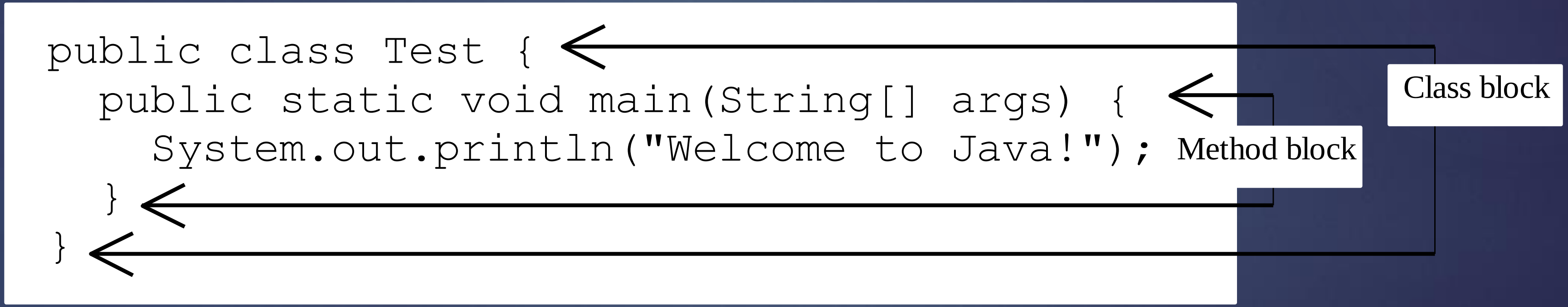
A statement represents an action or a sequence of actions.

The statement System.out.println("Welcome to Java!") in the program in example is a statement to display the greeting "Welcome to Java!"

Every statement in Java ends with a semicolon (;).

Blocks

A pair of braces in a program forms a block that groups components of a program.



```
public class Test {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Edited By : Hazleena Binti Osman

Classes

The class is the essential Java construct. A class is a template or blueprint for objects. To program in Java, you must understand classes and be able to write and use them. The mystery of the class will continue to be unveiled throughout this book.

For now, though, understand that a program is defined by using one or more classes.

Edited By : Hazleena Binti Osman

Methods

What is System.out.println?

It is a method: a collection of statements that performs a sequence of operations to display a message on the console.

It can be used even without fully understanding the details of how it works. It is used by invoking a statement with a string argument. The string argument is enclosed within parentheses.

In this case, the argument is "Welcome to Java!" You can call the same println method with a different argument to print a different message.

Edited By : Hazleena Binti Osman

main Method

The main method provides the control of program flow. The Java interpreter executes the application by invoking the main method.

The main method looks like this:

```
public static void main(String[] args) {
  // Statements;
}
```

Edited By : Hazleena Binti Osman

Java Architecture

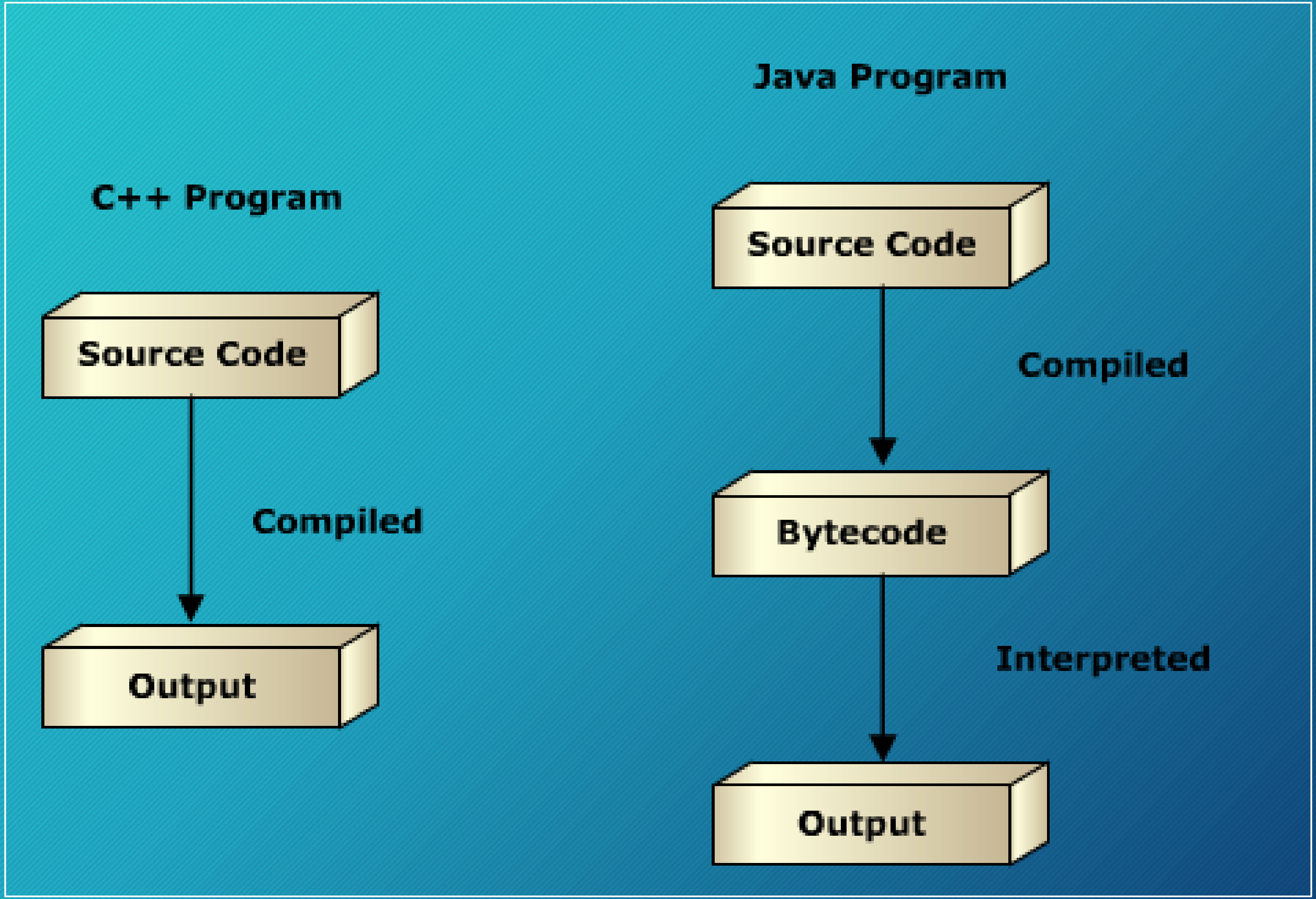
- Java is platform-independent as it can run on a wide variety of computers using different OS.
- There are four important components in Java architecture:
  - Java Source Code
  - Java Compiler
  - Java Byte code (Object code)
  - Java Virtual Machine (JVM)

Java Architecture - Components

- The components can be explained as follows:
  - **Java Source Code** - Program written in the form of text using Java.
  - **Java Compiler** - Used to convert source code into binary program that consists of byte code. It creates .class file.
  - **Java Byte Code** (Object code) - Byte code is a set of instructions that are machine-independent. Executed by JVM.
  - **Java Virtual Machine** (JVM) - Is a Java runtime system. Converts the bytecode in .class file to machine language.

C++ and Java Comparison

Compilation model of C++ and Java.



FUNDAMENTALS OF THE JAVA PROGRAMMING LANGUAGE

2

Understand Java terminology and environment

Edited By : Hazleena Binti Osman

Lesson Learning Outcome

- Understand Java terminology and environment
  - ❖ Show programming style and documentation in Java:
    - a. appropriate comments and comments style
    - b. naming conventions
    - c. proper indention and spacing
    - d. block styles
  - ❖ Explain the java architecture components:
    - a. Java Source Code
    - b. Java Complier
    - c. Java Virtual Machine
  - ❖ Prepare a simple Java programs in text editor.
  - ❖ Show the output of Java programs using javac and java command in command line prompt.
  - ❖ Identify type of errors in the source code :
    - a. compile-time errors
    - b. runtime errors
    - c. logic errors

Java Architecture

- Java is platform-independent as it can run on a wide variety of computers using different OS.
- There are four important components in Java architecture:
  - Java Source Code
  - Java Compiler
  - Java Byte code (Object code)
  - Java Virtual Machine (JVM)

Java Architecture - Components

- The components can be explained as follows:
  - **Java Source Code** - Program written in the form of text using Java.
  - **Java Compiler** - Used to convert source code into binary program that consists of byte code. It creates .class file.
  - **Java Byte Code** (Object code) - Byte code is a set of instructions that are machine-independent. Executed by JVM.
  - **Java Virtual Machine** (JVM) - Is a Java runtime system. Converts the bytecode in .class file to machine language.

Programming style and documentation in Java:

- Based on the following program

```
/* Activity For Today */
class Act       //  define class name
{ // beginning of class block
// main program where the program start
public static void main(String[] args)
{  // open curly brace for main block
         //statement to print string
System.out.println("Hello, my name is Mak Kiah.");
System.out.println("\tI'm from Pulau Pinang, Malaysia.");
   } //close for main block
} //end of class
```

Edited By : Hazleena Binti Osman

Prepare a simple Java programs in text editor.

```
/* Activity Simple Program */

class Act       //  define class name
{               // beginning of class block

   // main program
   public static void main(String[] args)
   {
         //statement to print string
         System.out.println("Hello, my name is Mak Kiah.");
         System.out.println("\t I'm from Pulau Pinang, Malaysia.");
   } //close for main block

} //end of class
```

Edited By : Hazleena Binti Osman

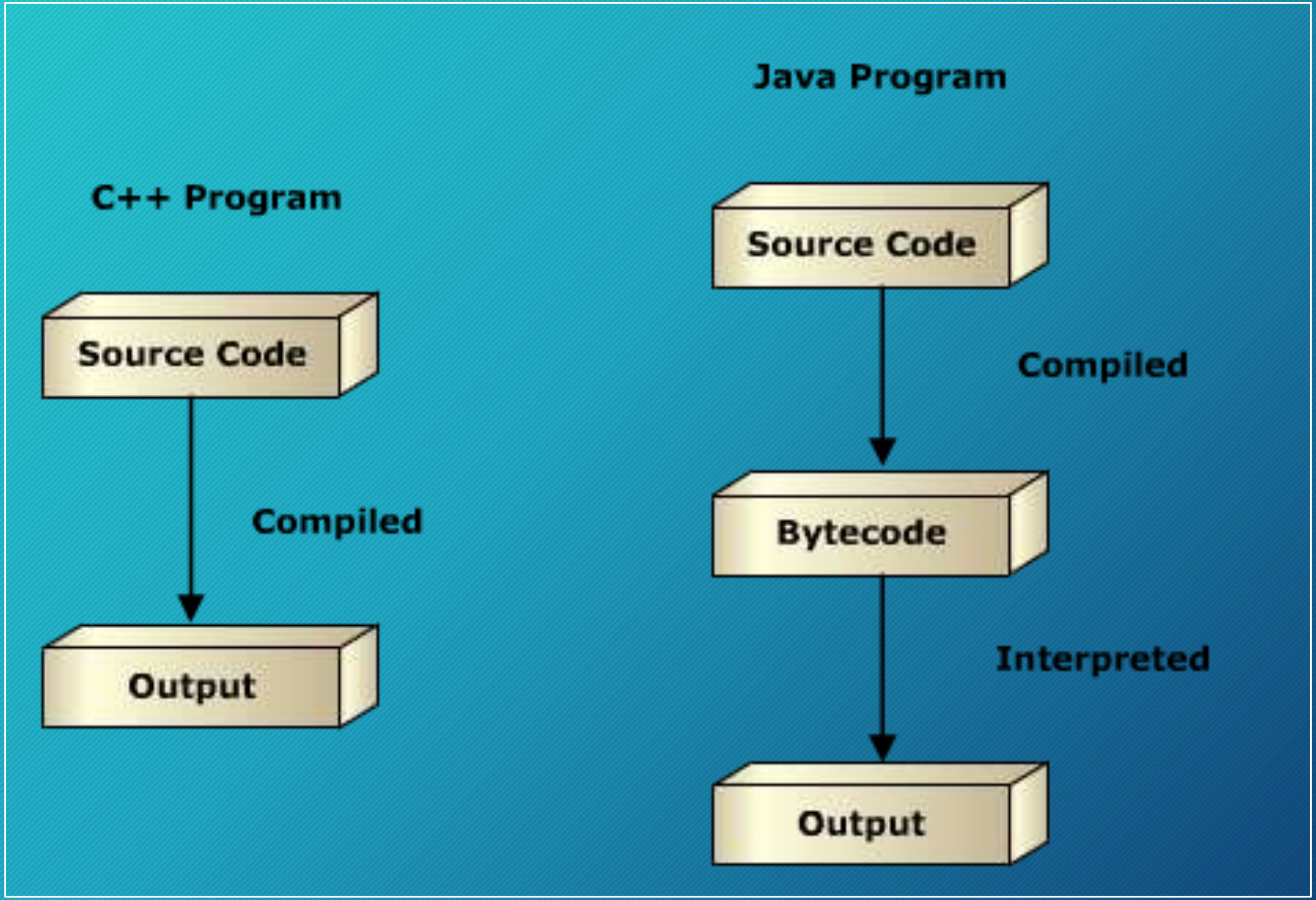
Type Of Errors

- There are 3 types of error in Java programming:
    a. compile-time errors
    b. runtime errors
    c. logic errors

definition presented by the student

C++ and Java Comparison

Compilation model of C++ and Java.


Java Program
C++ Program
Source Code
Source Code
Compiled
Compiled
Bytecode
Interpreted
Output
Output