Sukkur Institute of Business Administration University
Department of Computer Science
BS – II (CS/SE/AI) Spring 2024

Object Oriented Programming

Lab # 01: To become familiar with Downloading, installing & setting up JDK to write, compile, execute and debug Java programs

Instructor: Engr. Zainab Umair Kamangar

Lab Report Rubrics (Add the points in each column, then add across the bottom row to find the total score)					**Total Marks**
S.No	**Criterion**	**0.5**	**0.25**	**0.125**	
1	Accuracy	☐ Desired output	☐ Minor mistakes	☐ Critical mistakes	
2	Timing	☐ Submitted within the given time	☐ 1 day late	☐ More than 1 day late	

Submission Profile

Name:
Enrollment ID:
Comments:

Submission date (dd/mm/yy):
Receiving authority name and signature:

_____

Instructor Signature

Note: Submit this lab hand-out in the next lab with attached solved activities and exercises

Objectives

After performing this lab, students will be able to

- What is JAVA?
- Features of Java.
- JAVA Basics.
- Downloading, and Installing JDK and setting path.
- Writing HelloWorld.java in Text Editor
- Learn to write, compile, and execute simple Java programs using Notepad
- JAVA variables & data types.
- Input & Output
- Java Variable Type Conversion & Type Casting

What is JAVA?

Java was developed by Sun Microsystems in 1995. It is a **programming language** as well as **platform.** Java is among the most popular programming languages out there, mainly because of how versatile and compatible it is.

Java is general purpose programming language as it is used for software development, mobile applications, web servers, and client-side web applications. It is the native language of the Android operating system, which operates on Android phones and tablets.

You can use Java to:

- Build mobile apps: From the games you play to the banking apps you use, many of them are powered by Java!
- Develop websites: Java helps create the backbone of many websites, making them interactive and dynamic.
- Craft desktop software: From productivity tools to creative software, Java can be used to build programs for your computer.
- Power the internet: Java is used in servers that keep the internet running smoothly, behind the scenes! There are many java versions that has been released. A timeline of Java versions: https://www.codejava.net/java-se/java-se-versions-history
- Download Java for yourself: https://www.oracle.com/java/technologies/downloads/

Platform: Any hardware or software environment in which a program runs, is known as a platform. Since Java has a runtime environment (JRE) and API, it is called a platform.

Java Platforms

According to Oracle, there are four platforms of the Java programming language

- Java Platform, Standard Edition (Java SE)
- Java Platform, Enterprise Edition (Java EE)
- Java Platform, Micro Edition (Java ME)
- JavaFX/OpenJFX

Ready to dive into the world of Java? There are tons of resources available online and in libraries to help you get started. Remember, the journey of a thousand programs begins with a single line of code!

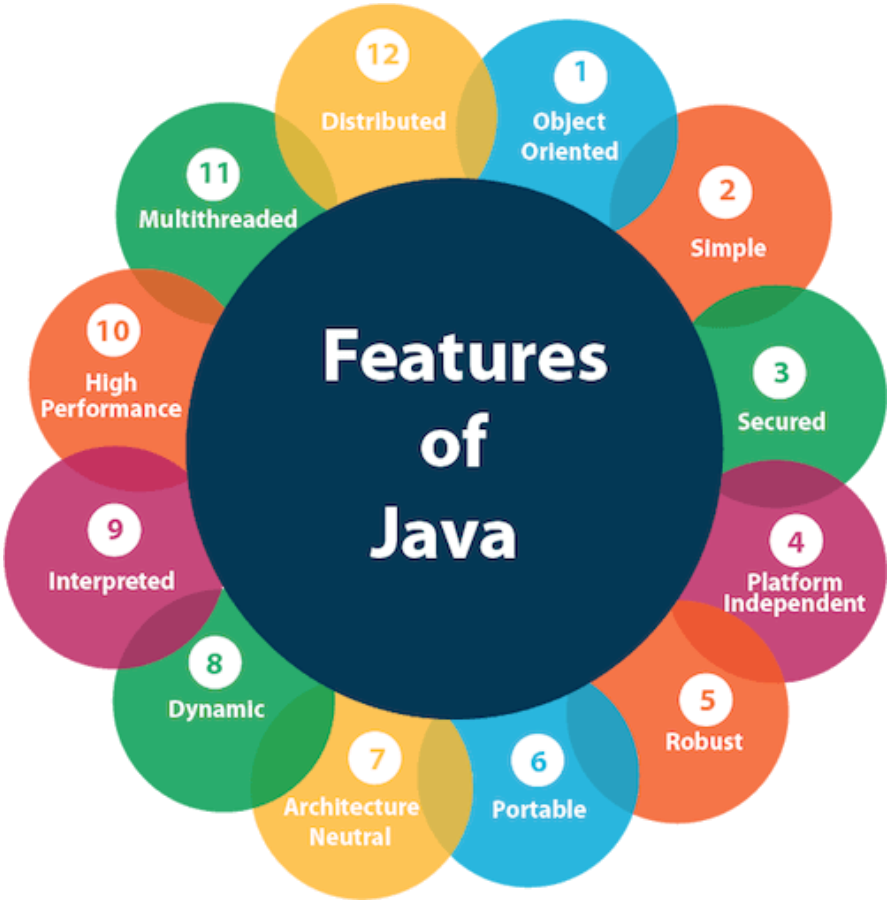
Bonus tip: Check out these websites for fun Java tutorials and projects:

Codecademy: https://www.codecademy.com/learn/learn-java

Udacity: https://www.udacity.com/course/java-programming-nanodegree--nd079

Coursera: https://www.coursera.org/specializations/object-oriented-programming

Features of JAVA



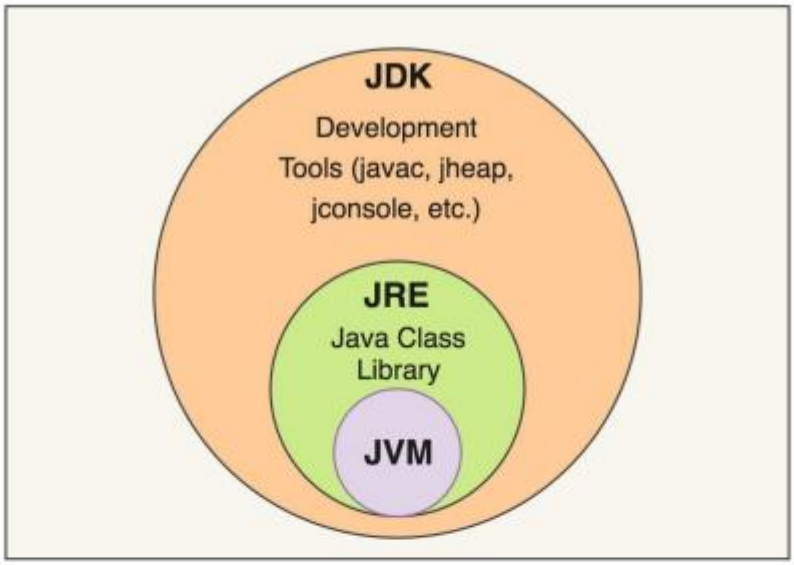
The following are Java's main features:

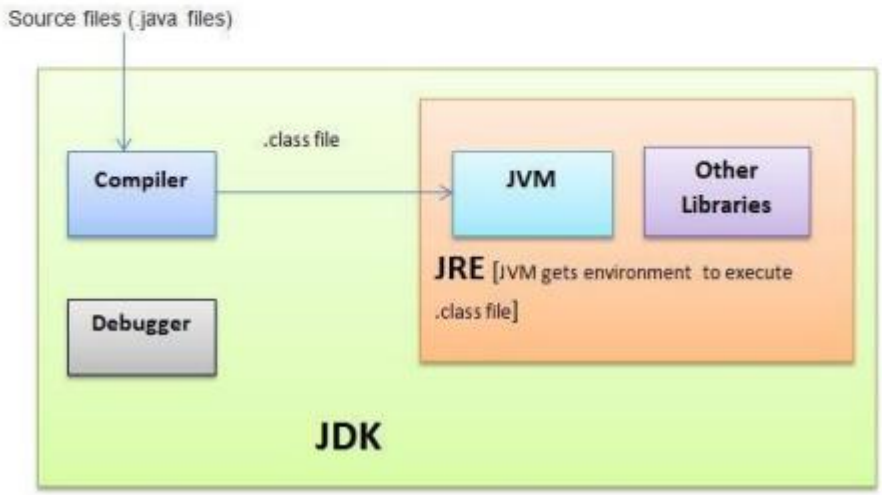
- **Object Oriented** – In Java, everything is an object, creating objects that contain data and methods.
- **Architecture-neutral** – Traditionally, we would have to recompile a program for every system that it was going to run on because all systems have a different idea of what their machine code should look like. Java compiler generates an architecture-neutral object file format called as bytecode, which makes the compiled code executable on many machines but with the presence of Java runtime system.
- **Platform Independent** – Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into byte code. This byte code is distributed over the web and interpreted by the Virtual Machine (JVM) on whichever platform it is being run on.
- **Portable** – Bytecode can be run by any system in which Java is installed. This is because when java is installed, Java virtual machine is also installed that is specific to that system. It is this machine's responsibility to convert the bytecode into the final instructions of that particular machine.

By making it the system's responsibility to do this final conversion, Java has created a write once, run anywhere language where anyone can hand you a Java program and you can run it on your machine

"Write once, run everywhere" WORA

JAVA Basics





JDK

- It stands for **Java Development Kit**, is a software development environment used for developing Java applications and applets.
- It compiles and executes new and already built applications.
- It is a collection of development tools as well as Java Runtime Environment (JRE), an interpreter/loader (Java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc) and other tools needed in Java development.

JRE

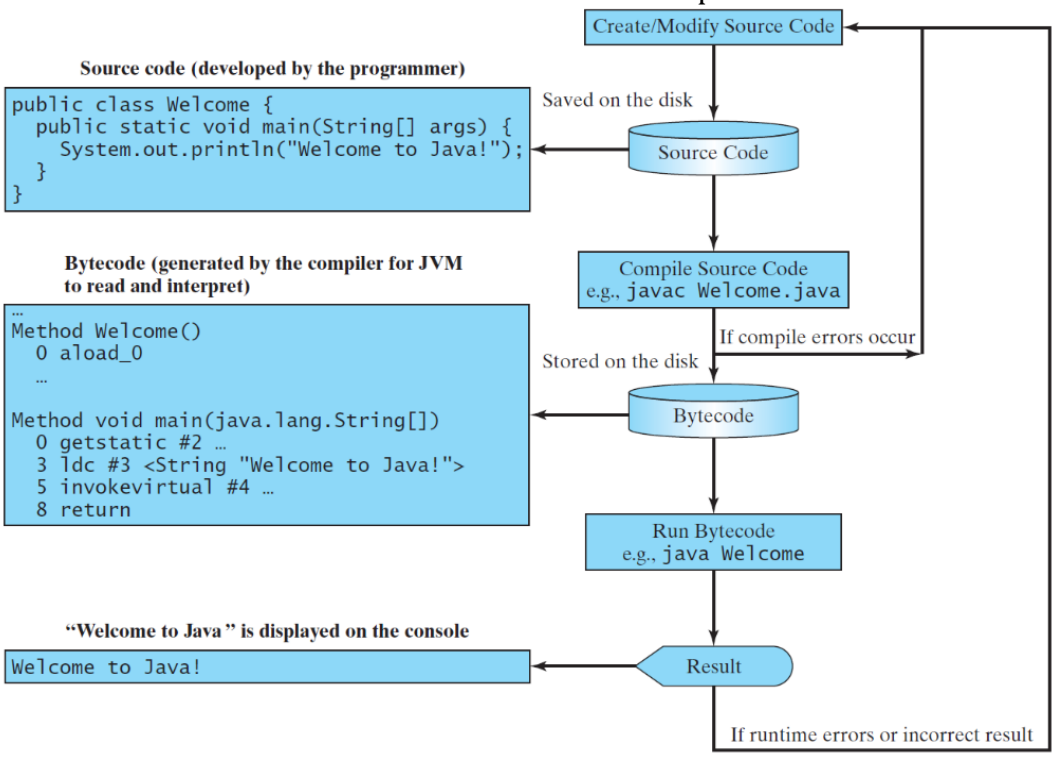
- It stands for **Java Runtime Environment**. The Java Runtime Environment provides the minimum requirements for executing a Java application.
- It consists of the **Java Virtual Machine (JVM), interpreter, JIT, core classes, and supporting files.**

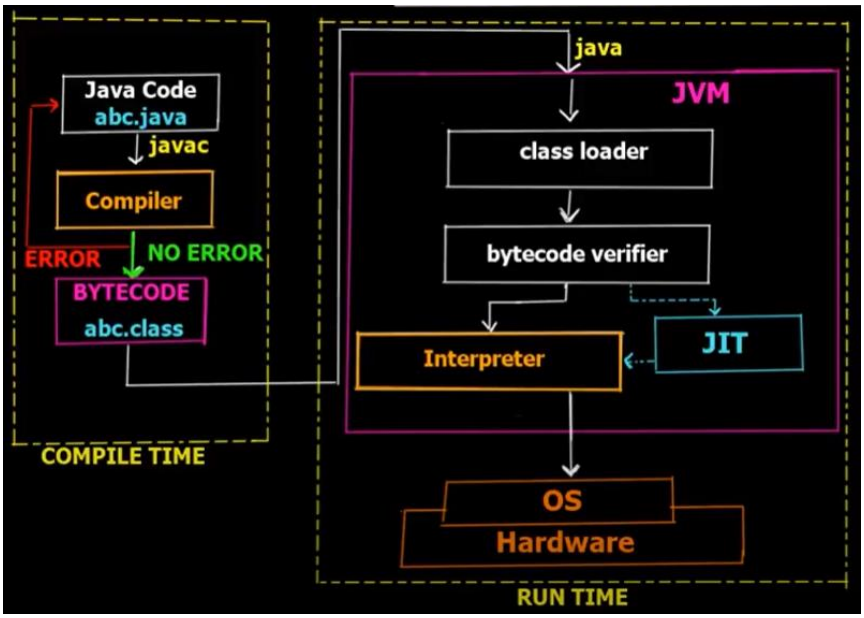
JVM

- It stands for **Java Virtual Machine (JVM)**
- It is responsible for executing bytecode where interpreter provides machine code for the current machine and has JIT as well.

JIT

- It stands for **Just-in-time Compiler**, is the part of the Java Virtual Machine (JVM) that is used to speed up the execution time.
- JIT interprets parts of the bytecode that have similar functionality at the same time and hence reduces the amount of time needed for full interpretation.





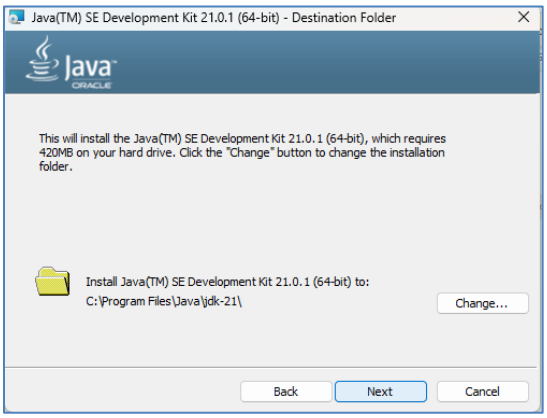
Installing JDK and setting path

To develop Java applications on our computers, we require a JDK. Visit the link below to download the JDK setup.

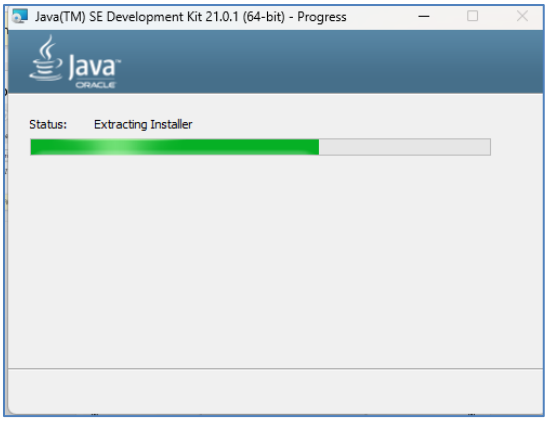
https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.exe

Double click downloaded JDK and click YES to allow JDK to make changes to system and then click NEXT.

Installation location by default is C:\Program Files\Java\jdk-17\



Installation in progress



JDK successfully installed, click CLOSE

We need to set Java path so that whenever java program will be compiled or executed it can locate JDK easily.

Path can be set in two ways i.e. **temporarily and permanently**.

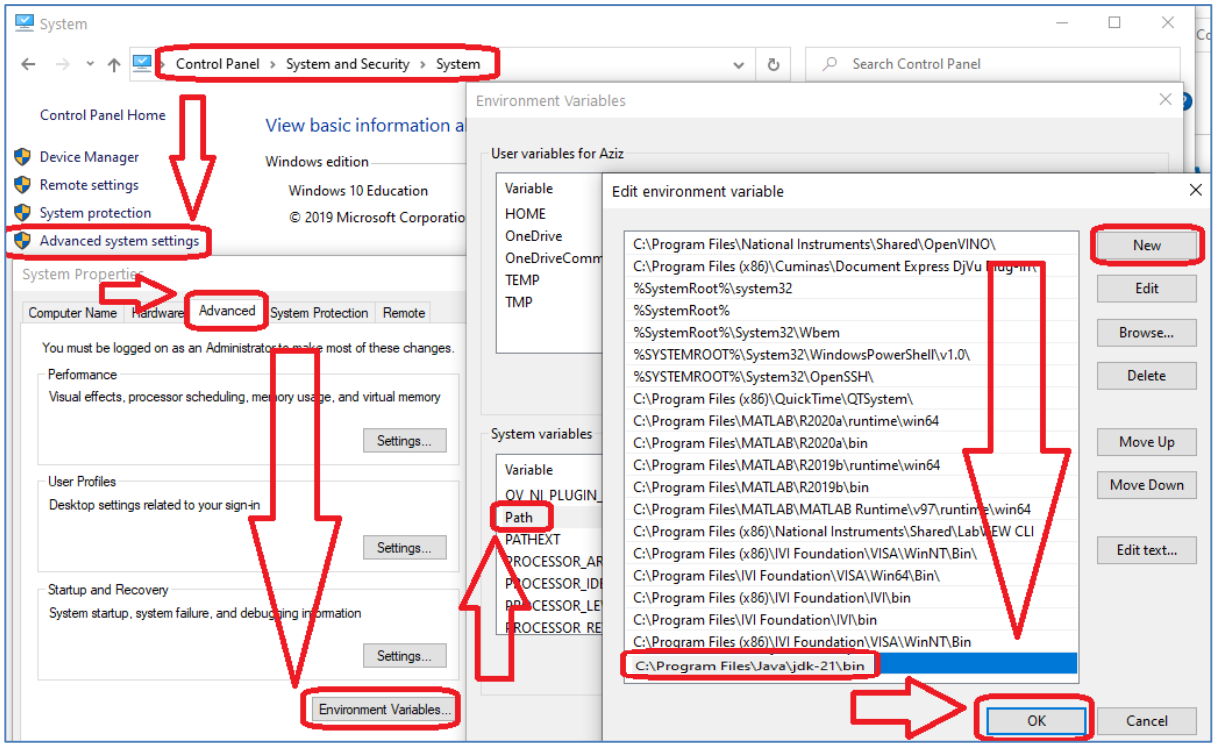
To set java path **temporarily**, open command prompt and run command

"set path=C:\Program Files\Java\jdk-21\bin".

Once the command prompt is closed, java path needs to be set again.

To set java path **permanently**, this path "C:\Program Files\Java\jdk-21 \bin" should be added as per following steps.

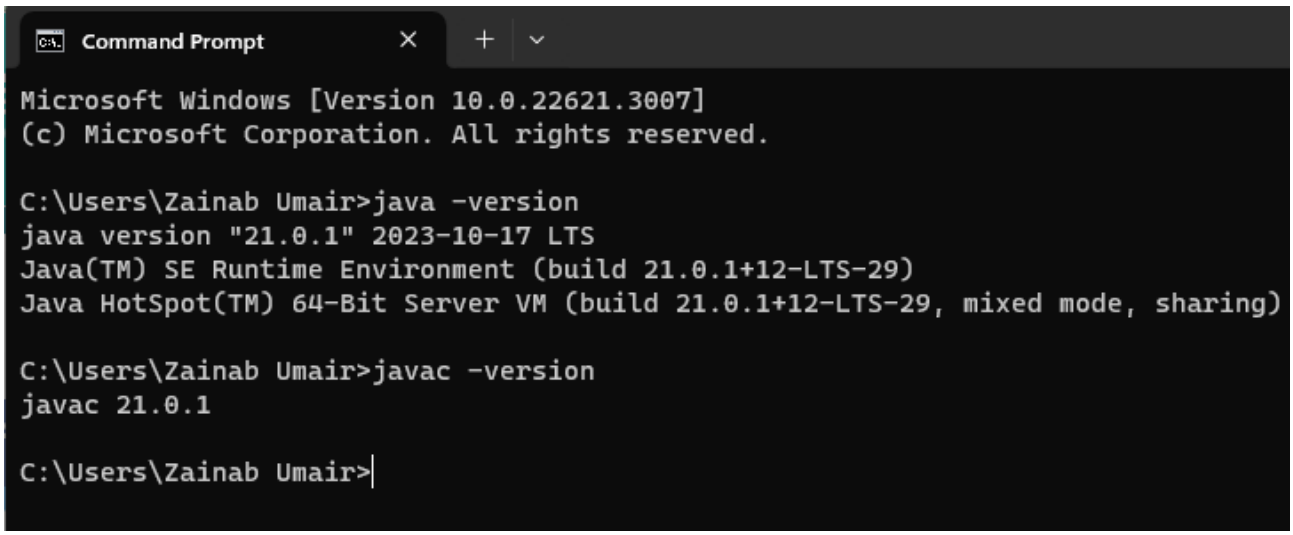
Go to **Control Panel > System and Security > System**, and then click **Advanced system settings** to view System properties window. In system properties window press advanced tab and click on **Environment variables** to add new path. Finally click ok and you are all set.



To check if the Java compiler is installed, open command prompt and run command "javac -version".

If the command returns something such as "javac 21.0.1", it means JDK having Java compiler is installed successfully.

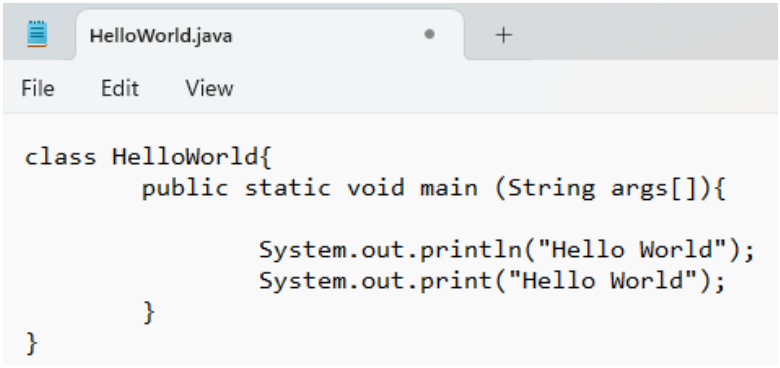
To check version information of JDK, JRE and JVM, open command prompt and run command "java -version".



Writing HelloWorld.java in Text Editor

Follow the below given steps:

i. Run notepad and enter below given code. Save this file with Class name and end it with ".java" extension. Save it on desktop for now!



ii. Go to search bar in taskbar, write cmd and Open Command Prompt then write the following commands;

   a. cd desktop //for going to desktop
   b. javac HelloWorld.java
   c. java HelloWorld

The Java programming language compiler (javac) takes your source file and translates the code into instructions known as bytecodes. "Java ClassName" will enable Java virtual machine to run your application/code.

Variables and data types in JAVA

In Java, there are three types of variables:

- Local Variables
- Instance Variables
- Static Variables

Local Variables

Local Variables are declared inside the body of a method.

***Scope*:** Variables declared inside a method have method level scope and cannot be accessed outside the method.

Instance Variables

Instance variables are defined without the 'static' keyword. They are defined outside a method within a class. Access modifiers can be given for instance variables. They are Object-specific.

***Scope*:** Dependent on the access modifier.

Class/Static Variables

It is declared with the keyword 'static', outside the method within a class. Static variables are created when the program starts and destroyed when the program stops. There would only be one copy of each static variable per class, regardless of how many objects are created.

***Scope*:** Visibility is like instance variables. However, most static variables are declared public since they must be available for users of the class.

Example: Difference between static and instance variable

```
public class MyClass {
   int instanceVariable;       // Instance variable
   static int staticVariable;  // Static variable
   // Constructor
   public MyClass(int value) {
      this.instanceVariable = value;
        this.staticVariable = value;
   }
   public static void main(String[] args) {
      // Creating two objects of MyClass
      MyClass obj1 = new MyClass(10);
      MyClass obj2 = new MyClass(20);
      // Accessing instance variables using object references
      System.out.println("Instance Variable");
      System.out.println("Object 1: " + obj1.instanceVariable);
      System.out.println("Object 2: " + obj2.instanceVariable);
      // Accessing static variables using object references
      System.out.println("\nStatic Variable");
      System.out.println("Object 1: " + obj1.staticVariable);
```

```
        System.out.println("Object 2: " + obj2.staticVariable);
    }
}
```

Output:

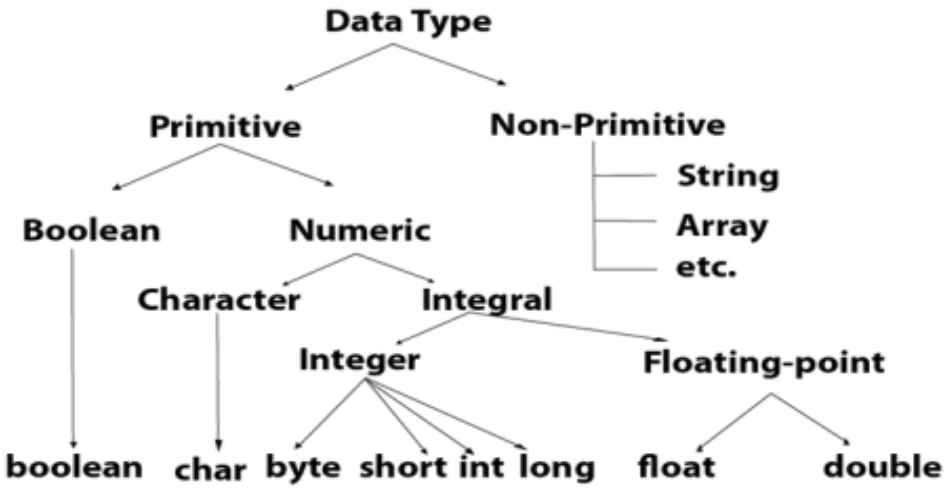
```
C:\Users\Zainab Umair\Desktop\Java Zainab>java MyClass
Instance Variable
Object 1: 10
Object 2: 20

Static Variable
Object 1: 20
Object 2: 20
```

Data Types in Java

Data types classify the different values to be stored in the variable. In java, there are two types of data types:

- Primitive Data Types
- Non-primitive Data Types



Data Type	Default Value	Default size
Boolean	false	1 bit
Char	'\u0000'	2 byte
Byte	0	1 byte
Short	0	2 byte
Int	0	4 byte
Long	0L	8 byte
Float	0.0f	4 byte
Double	0.0d	8 byte

Input & Output

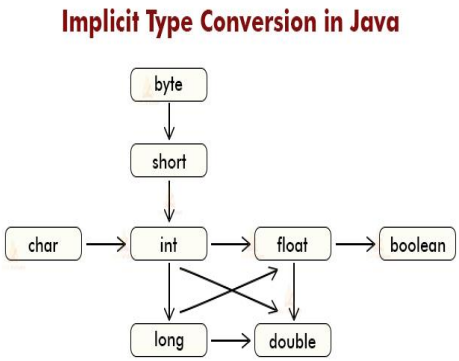
Output in Java Syntax:
System.out.println("Hello World");

Input in Java Syntax:
// import library
import java.util.Scanner;
// Creating scanner object
Scanner ip = new Scanner(System.in); //system.in represents that the input is given via keyboard
Taking input from user
int ipFrmUser = ip.nextInt(); // Integer Input
double ipDbUser = ip.nextDouble(); // Double Input

Java Variable Type Conversion & Type Casting

A variable of one type can receive the value of another type. Here there are 2 cases.
Case 1: A variable of smaller capacity is assigned to another variable of bigger capacity.

```
double d ;
int i = 10;
d = i;
```



This process is Automatic, and non-explicit is known as **type conversion**

Case 2: Variable of larger capacity is be assigned to another variable of smaller capacity.

```
double d   = 10;
int i;
i = (int) d
```

Type Cast Operator

In such cases, you have to explicitly specify the type cast operator. This process is known as **type casting**.
In case, you do not specify a type cast operator; the compiler gives an error. Since this rule is enforced by the compiler, it makes the programmer aware that the conversion he is about to do may cause some loss in data and prevents accidental losses.

Example: To Understand Type Casting

```
class Demo {
 public static void main(String args[]) {
  byte x;
  int a = 270;
  double b = 128.128;
  System.out.println("int converted to byte");
  x = (byte) a;
  System.out.println("a and x " + a + " " + x);
  System.out.println("double converted to int");
  a = (int) b;
  System.out.println("b and a " + b + " " + a);
  System.out.println("\ndouble converted to byte");
  x = (byte)b;
  System.out.println("b and x " + b + " " + x);
 }
}
```

Output:

```
int converted to byte
a and x 270 14
double converted to int
b and a 128.128 128

double converted to byte
b and x 128.128 -128
```

Exercises

Question 1: (JAVA Environment Installation & Error Messages)

Set up a Java development environment. In the main ( ) method of your program try to compile the following invalid Java code snippets. Record the error messages you receive. What do you think each error message indicates?

System.out.printn(“Hello World”)

System.out.printn(Hello World)

System.out.println”Hello World”;

println(“Hello World);

To generate one final error message, remove one of the brackets from the end of your program. Now what message do you receive?

Question: 2 (Input)

Write a program that takes user input and displays it.

Question: 3 (Basic Arithmetic Operation)

Write a program that takes two double values from the user and performs basic arithmetic operations (addition, subtraction, multiplication, division).

Question: 4 (Type casting)

Perform division using two int variables and store the result in float variable and print the results.

Question: 5 Can you cast string into int?

Question: 6 Why JAVA when there are other OOP languages?