Define Topics and With Code Java Basic

1. **Introduction to Java**

What is Java?

Definition: Java is a high-level, object-oriented programming language developed by Sun Microsystems (now owned by Oracle). It is platform-independent, meaning code written in Java can run on any device that has the Java Virtual Machine (JVM).

Example Program:

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

Explanation: This simple program prints "Hello, World!" to the console, showcasing the basic structure of a Java program.

History and Features of Java

Definition: Java was first released in 1995 by Sun Microsystems. Key features of Java include platform independence, object-oriented programming, robustness, security, and multithreading capabilities.

Example Program:

```java
public class JavaFeatures {
    public static void main(String[] args) {
        System.out.println("Java is platform-independent, secure, and robust!");
    }
}
```

Explanation: This program highlights the features of Java by printing a statement that emphasizes its key characteristics.

Installing JDK and Setting Up the Environment

Definition: The Java Development Kit (JDK) includes tools necessary for developing Java applications, such as the compiler and runtime environment. Setting up involves downloading the JDK and configuring the `PATH` environment variable to include the JDK's `bin` directory.

Example Program:

```java
public class SetupCheck {
    public static void main(String[] args) {
        System.out.println("JDK is installed and environment is set up!");
    }
}
```

Explanation: Once the environment is correctly set up, this program confirms it by successfully compiling and running the code.

Writing and Running Your First Java Program

Definition: Writing a basic Java program involves creating a `.java` file, defining a class, and including the `main` method where the program's execution begins.

Example Program:

```java
public class FirstProgram {
    public static void main(String[] args) {
        System.out.println("This is my first Java program!");
    }
}
```

Explanation: This program demonstrates the basic structure of a Java application, including the class declaration and the main method.

2. **Java Basics**

1) Variables and Data Types

Definition: Variables are containers for storing data values, and data types specify the type of data a variable can hold (e.g., `int`, `double`, `char`, `boolean`).

Example Program:

```java
public class VariablesExample {
    public static void main(String[] args) {
        int age = 25;

        double salary = 45000.50;

        char grade = 'A';

        boolean isJavaFun = true;

        System.out.println("Age: " + age);

        System.out.println("Salary: " + salary);

        System.out.println("Grade: " + grade);

        System.out.println("Is Java Fun? " + isJavaFun);

    }

}
```

Explanation: This program declares variables of different data types and prints their values.

(2) Operators (Arithmetic, Logical, Relational, Bitwise)

Definition: Operators are symbols that perform operations on variables and values. Arithmetic operators perform basic math, logical operators are used in boolean expressions, relational operators compare values, and bitwise operators perform bit-level operations.

Example Program:

```java
public class OperatorsExample {
    public static void main(String[] args) {
        int a = 10;
        int b = 5;
        System.out.println("Addition: " + (a + b));       // Arithmetic
        System.out.println("Is a > b? " + (a > b));       // Relational
        System.out.println("Logical AND: " + (a > b && b > 0)); // Logical
        System.out.println("Bitwise AND: " + (a & b));    // Bitwise
    }
}
```

Explanation: This program demonstrates the use of different types of operators in Java.

(3) Control Statements (if-else, switch-case)

Definition: Control statements are used to determine the flow of execution based on conditions. `if-else` handles conditional branching, while `switch-case` is used for selecting one of many code blocks to execute.

Example Program (if-else):

```java
public class IfElseExample {
    public static void main(String[] args) {
        int number = 10;
        if (number > 0) {
            System.out.println("The number is positive.");
        } else {
            System.out.println("The number is non-positive.");
        }
    }
}
```

(4)Example Program (switch-case):**

```java
public class SwitchCaseExample {
    public static void main(String[] args) {
        int day = 3;
        switch (day) {
            case 1:
                System.out.println("Monday");
                break;
            case 2:
                System.out.println("Tuesday");
                break;
            case 3:
                System.out.println("Wednesday");
                break;
            default:
                System.out.println("Invalid day");
                break;
        }
    }
}
```

Explanation: These programs demonstrate how to control the flow of execution based on conditions.

(5) Loops (for, while, do-while)

Definition: Loops are used to execute a block of code repeatedly as long as a specified condition is true. `for`, `while`, and `do-while` are the primary loop constructs in Java.

(1)Example Program (for loop):**

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

(2)Example Program (while loop):**

```java
```

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

(3)Example Program (do-while loop):**

```java
public class DoWhileLoopExample {
    public static void main(String[] args) {
        int i = 1;
        do {
            System.out.println("Iteration: " + i);
            i++;
        } while (i <= 5);
    }
}
```

Explanation: These programs demonstrate the use of different loops to perform repetitive tasks.

(6) Arrays

Definition: Arrays are used to store multiple values of the same type in a single variable, accessible by an index.

Example Program:

```java
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {1, 2, 3, 4, 5};
        for (int i = 0; i < numbers.length; i++) {
            System.out.println("Element at index " + i + ": " + numbers[i]);
        }
    }
}
```

```

**Explanation:** This program creates an array of integers and prints each element using a loop.
```