PPT JAVA ASSIGNMENT-1

Q1. What is the difference between Compiler and Interpreter?

- Compiler: A compiler is a software program that translates the entire source code of a program into machine code before execution. It analyzes the entire code and generates an executable file, which can be executed directly by the computer's processor. Examples of compiled languages are C, C++, and Java (which uses a combination of compilation and interpretation).

- Interpreter: An interpreter, on the other hand, reads and executes the source code line by line. It translates and executes each line of code one at a time. Interpreted languages do not produce an executable file but directly interpret and execute the code. Examples of interpreted languages are Python, Ruby, and JavaScript.

Q2. What is the difference between JDK, JRE, and JVM?

- JDK (Java Development Kit): JDK is a software development kit that includes tools necessary for developing Java applications. It provides the Java compiler (javac), runtime environment (JRE), and other development tools like debuggers and documentation.

- JRE (Java Runtime Environment): JRE is the runtime environment for executing Java applications. It includes the Java Virtual Machine (JVM) and the necessary libraries and files required to run Java programs. JRE does not include development tools.

- JVM (Java Virtual Machine): JVM is the virtual machine that executes Java bytecode. It is responsible for running Java applications on different platforms by converting bytecode into machine-specific instructions. JVM provides memory management, garbage collection, and runtime environment for Java programs.

Q3. How many types of memory areas are allocated by JVM?

JVM allocates memory in several areas:

- Heap: It is the runtime data area where objects are allocated and deallocated. The heap is shared among all threads and is divided into two main areas: Young Generation and Old Generation.

- Stack: Each thread in Java has its own stack memory. It is used to store local variables, method parameters, and method invocations. Stack memory is private to each thread and is not shared.

- Method Area: It stores class-level data like static variables, method bytecode, and constant pool. It is shared among all threads and contains information about classes and methods.

- PC Register: It holds the address of the current instruction being executed.

- Native Method Stack: It is used to store native method information and is separate from the Java stack.

Q4. What is JIT compiler?

JIT stands for Just-In-Time compiler. It is a part of the JVM that dynamically compiles bytecode into machine code during runtime. Instead of interpreting the entire bytecode, the JIT compiler identifies frequently executed code (hotspots) and optimizes them for better performance. It can significantly improve the execution speed of Java applications.

Q5. What are the various access specifiers in Java?

In Java, there are four access specifiers:

- Public: Public members are accessible from anywhere, both within the same class and from other classes.

- Protected: Protected members are accessible within the same class, subclasses, and other classes in the same package.

- Default (no specifier): Default members are accessible within the same class and other classes in the same package. It is also known as package-private.

- Private: Private members are accessible only within the same class and not accessible from outside the class.

Q6. What is a compiler in Java?

A compiler is a software program that translates human-readable source code written in a high-level programming language (such as Java) into machine-readable code (such as bytecode or machine code) that can be executed by the computer's processor. The Java compiler (javac) compiles Java source code into platform-independent bytecode, which can then be executed by the Java Virtual Machine (JVM).

Q7. Explain the types of variables in Java?

In Java, there are three types of variables:

- Local Variables: Local variables are declared within a method, constructor, or a block. They are only accessible within their scope and do not retain their values once the scope is exited.

- Instance Variables: Instance variables are declared within a class but outside any method. They belong to an instance of the class and each instance of the class has its own copy of instance variables.

- Class (Static) Variables: Class variables are declared with the `static` keyword within a class. They are shared among all instances of the class and have the same value for all objects.

Q8. What are the Datatypes in Java?

Java provides several built-in data types:

- Primitive Data Types: boolean, byte, short, int, long, float, double, char.

- Reference Data Types: Classes, interfaces, arrays, enumerations, and other reference types.

Q9. What are the identifiers in Java?

In Java, identifiers are used to name variables, methods, classes, packages, and other program entities. They are user-defined names and must follow certain rules:

- They can consist of letters, digits, underscore (_), or dollar sign ($).

- They must start with a letter, underscore, or dollar sign (cannot start with a digit).

- They are case-sensitive.

- They cannot be a reserved word (keyword) in Java.

Q10. Explain the architecture of JVM.

The architecture of the JVM consists of three main components:

- Class Loader: Responsible for loading classes into memory. It loads the bytecode, verifies its integrity, and resolves references to other classes.

- Execution Engine: Executes the bytecode by interpreting it or using just-in-time (JIT) compilation. It includes the interpreter, JIT compiler, and runtime system.

- Runtime Data Areas: These areas include the method area, heap, stack, PC register, and native method stack. They are used to store data, bytecode, objects, method information, and other runtime elements required during program execution.