Group - A

1.Why main() is declared as public & static in JAVA?

Ans. Java **main()**method is always static, so that compiler can call it without the creation of an object or before the creation of an object of the class.

In any Java program, the **main()** method is the starting point from where compiler starts program execution. So, the compiler needs to call the main() method.

If the**main()** is allowed to be non-static, then while calling the **main()** method JVM has to instantiate its class.

While instantiating it has to call the constructor of that class, There will be ambiguity if the constructor of that class takes an argument.

Static method of a class can be called by using the class name only without creating an object of a class.

The **main()**method in Java must be declared **public**, **static**and **void**. If any of these are missing, the Java program will compile but a runtime error will be thrown.

1. Why JAVA is platform independent?

Ans: The meaning of platform-independent is that the java compiled code(byte code) can run on all operating systems.  
A program is written in a language that is a human-readable language. It may contain words, phrases, etc which the machine does not understand. For the source code to be understood by the machine, it needs to be in a language understood by machines, typically a machine-level language. So, here comes the role of a compiler. The compiler converts the high-level language (human language) into a format understood by the machines. Therefore, a compiler is a program that translates the source code for another program from a programming language into executable code.  
This executable code may be a sequence of machine instructions that can be executed by the CPU directly, or it may be an intermediate representation that is interpreted by a virtual machine. This intermediate representation in Java is the **Java Byte Code.**

1. Describe different object oriented features in JAVA.

Ans: OOPs concept brings this **data**and **behavior**in a single place called **“class”** and we can **create any number of objects** to represent different state for each object.

Object-oriented programming System(OOPs) is a programming paradigm based on the concept of “objects” that contain data and methods. The primary purpose of object-oriented programming is to increase the flexibility and maintainability of programs. Object oriented programming brings together data and its behaviour(methods) in a single location(object) makes it easier to understand how a program works. We will cover each and every feature of OOPs in detail so that you won’t face any difficultly understanding **OOPs Concepts.**

**Object Oriented Features in Java**

**Features of OOPs:**

1. Classes
2. Objects
3. Data Abstraction
4. Encapsulation
5. Inheritance
6. Polymorphism

4.Explain the use of super in JAVA?

Ans: The**super** keyword in java is a reference variable that is used to refer parent class objects.  The keyword “super” came into the picture with the concept of Inheritance. It is majorly used in the following contexts:

**1. Use of super with variables:**This scenario occurs when a derived class and base class has same data members. In that case there is a possibility of ambiguity for the JVM. We can understand it more clearly

5. How does JAVA Garbage collector work?

Ans: Garbage collection works by employing several GC algorithm e.g. Mark and Sweep. There are different kinds of garbage collector available in Java to collect different area of heap memory e.g. you have serial, parallel and concurrent garbage collector in Java. A new collector called G1 (Garbage first) are also introduced in JDK 1.7. First step to learn about GC is to understand when an object becomes eligible to garbage collection.

Since JVM provides memory management, Java developers only care about creating object, they don't care about cleaning up, that is done by garbage collector, but it can only collect objects which has no live strong reference or it's not reachable from any thread. If an object, which is suppose to be collected but still live in memory due to unintentional strong reference then it's known as memory leak in Java. ThreadLocal variables in Java web application can easily cause memory leak.

Group - B

1. “In JAVA the variables are initialized before any method called even constructor.”- Explain with an example.

Ans: Static variable in Java is variable which belongs to the class and initialized only once at the start of the execution. It is a variable which belongs to the class and not to object(instance) Static variables are initialized only once, at the start of the execution. These variables will be initialized first, before the initialization of any instance variables A single copy to be shared by all instances of the class A static variable can be accessed directly by the class name and doesn’t need any object

1. Write an example where the return type of the function is the object itself.

Ans: User-defined functions and class methods can define return types as object references (as class or interface types). When an object is passed locally, class instances are always returned by reference. Thus, only a reference to an object is returned, not the object itself. When an object is passed between an AppServer and an ABL client, the object is serialized by the sender, and the receiver uses this information to create a copy of the object. (See *OpenEdge Application Server: Developing AppServer Applications* for additional restrictions on objects passed between an AppServer an an ABL client.)

1. What is the function of “finally” and “final” ?

Ans:

(FINAL):: final(lowercase) is a reserved keyword in java. We can’t use it as an identifier as it is reserved. We can use this keyword with variables, methods and also with classes. The final keyword in java has different meaning depending upon it is applied to variable, class or method.

(FINALLY):: Just as final is a reserved keyword, so in same way finally is also a reserved keyword in java i.e, we can’t use it as an identifier. The finally keyword is used in association with a [try/catch block](https://www.geeksforgeeks.org/flow-control-in-try-catch-finally-in-java/) and guarantees that a section of code will be executed, even if an exception is thrown. The finally block will be executed after the try and catch blocks, but before control transfers back to its origin.