Assignment -1

Unit-1

1. What is Java?
2. List and explain the primitive data types in Java.
3. Define literals with examples (decimal, octal, hexadecimal, binary, string, character).
4. What are Java keywords? Give examples.
5. Define JVM, JDK, and JRE.
6. Explain the history of Java and its evolution from Oak.
7. Explain the major features of Java that make it platform-independent and secure.
8. Explain Java buzzwords with examples.
9. Describe the role of Java Class Libraries (APIs) in programming.
10. Explain entry-controlled and exit-controlled loops with suitable examples.
16. Compare Java with other programming languages like C/C++.
17. Differentiate between local, instance, and static variables with examples.
18. Compare for, while, and do-while loops with examples.
19. Differentiate between break and continue with examples.
20. Evaluate the difference between implicit type conversion and explicit type casting in Java.
21. Justify why Java is considered secure and robust.
22. Assess the importance of indentation and formatting in Java programs.
23. Assess the limitations of Scanner class for user input.
24. Develop a Java class implementing all four OOP principles (abstraction, encapsulation, inheritance, polymorphism).
25. Create a Java program that demonstrates use of operators (arithmetic, relational, logical, ternary, and shorthand assignments).

Unit-2

1. What is Class and Object? Explain the difference between them.
2. What is the general form of a class in Java?
3. List and explain Java's access modifiers (private, protected, public, default).
4. What are constructors in Java? Differentiate between default and parameterized constructors
   OR
   Explain types of constructors.
5. What are static fields and static methods in Java?
   OR
   Explain static fields and static methods in detail.
6. Explain encapsulation in Java. What are its benefits?
7. Explain the use of the final keyword in detail.
8. Explain how objects are created in Java using the new operator.
9. Explain mutator (setter) and accessor (getter) methods with examples.
10. Explain object initialization blocks and static blocks in detail.
11. Write a Java program to demonstrate constructor overloading with examples.
12. Write a Java program to demonstrate method overloading (compile-time polymorphism).
13. Write a Java program to demonstrate method overriding (run-time polymorphism).
14. Write a Java program that passes objects as parameters to methods.
15. Design a Java class BankAccount with methods to deposit, withdraw, and display balance using access modifiers properly.
16. Compare method overloading and method overriding with suitable examples.
17. Differentiate between instance variables and reference variables in Java.
18. Analyse the difference between static and non-static methods.
19. Compare the usage of this and super keywords with examples.
20. Differentiate between nested classes and inner classes in Java.
21. Evaluate the advantages of using access modifiers for data security.
22. Assess the role of constructors in object initialization compared to setter methods.
23. Judge the importance of recursion in Java compared to iterative solutions.
24. Develop a Java program demonstrating the use of static fields, static blocks, and static methods.
25. Create a Java class implementing all four OOP principles (abstraction, encapsulation, inheritance, polymorphism).