**Evidence Sample question:**

1. Write a Java program that creates an array with 100 randomly chosen integers. It prompts the user for an index and displays the corresponding element value. If the index is out of bounds, it shows "Out of Bounds" with an exception handler.
2. Design a Java program using abstraction for a vehicle management system. Create an abstract class Vehicle with properties: registrationNumber (String), brand (String), and year (int). Include abstract methods start() and stop().

Implement concrete subclasses Car and Motorcycle inheriting from Vehicle. In Car, display "Car engine started" in start() and "Car engine stopped" in stop(). In Motorcycle, display "Motorcycle engine started" in start() and "Motorcycle engine stopped" in stop().

Provide code for these classes and demonstrate creating instances to start and stop engines.

1. Design a Java program for a simple student management system using Binary I/O operations. Create a Student class with properties name (String) and age (int). Implement methods to write student information to a binary file named "students.dat" and read/display this information from the file. Show the program's functionality by creating instances of the Student class, writing their data to the file, and then reading and displaying it.

Provide the Java code showcasing Binary I/O operations.

1. Design a Java banking application with abstraction. Create an abstract class BankAccount with properties: accountNumber (String), accountHolder (String), balance (double), and methods: a constructor to initialize properties, deposit(double amount) to add funds, and withdraw(double amount) (abstract) to deduct funds.

Implement subclasses SavingsAccount and CheckingAccount inheriting from BankAccount. In SavingsAccount, withdrawals are possible if the balance allows it. In CheckingAccount, withdrawals can go negative up to an overdraft limit.

Provide the code for these classes and demonstrate creating instances and performing deposit and withdrawal operations.

**Descriptive question**

1. Difference between overriding and overloading.
2. If a method in a subclass has the same signature as a method in its superclass with the same return type, is the method overridden or overloaded?
3. What keyword do you use to define a subclass?
4. What is the advantage of using exception handling?
5. What is the keyword throw used for? What is the keyword throws used for?
6. What are the differences between abstract classes and interfaces?
7. What is encapsulation?
8. What are the differences between text I/O and binary I/O?
9. Why we use Serializable Interface.