by Kunal Sir

Assignment:Abstraction (interface)

Part A: MCQ's

1. The main goal of abstraction as explained in the PDF is to:
A. Hide variables
B. Increase execution speed
C. Reduce complexity by exposing only essential details
D. Support polymorphism

2. Essential knowledge in abstraction refers to:
A. Internal logic of method
B. Constructor execution
C. Method name, parameters, return type, and purpose
D. JVM memory allocation

3. Which keyword is used to design an interface in Java?
A. class
B. abstract
C. interface
D. implements

4. According to the PDF, interface variables are:
A. private and static
B. public only
C. public, static, and final
D. protected and final

5. Why must interface variables be initialized?
A. Because they are static
B. Because they are final
C. Because constructors initialize them
D. Because JVM enforces it

6. Till JDK 1.7, interface methods were:
A. Concrete
B. Static
C. Abstract only
D. Default

7. Which statement about interface methods is TRUE?
A. They can have method body
B. They end with semicolon
C. They must be protected
D. They must be static

8. Interface methods are by default:
A. private abstract
B. public abstract
C. protected
D. final

9. Can we create an object of an interface?
A. Yes
B. No
C. Only using new keyword
D. Only in JVM

10. Which relationship is correct?
A. Class extends interface
B. Interface implements class
C. Class implements interface
D. Interface creates class

11. Java supports multiple inheritance using interface because:
A. Interfaces have constructors
B. Interfaces avoid ambiguity
C. Interfaces use static methods
D. JVM ignores conflicts

12. Why does interface not contain constructor?
A. Constructors are private
B. Interfaces cannot be instantiated
C. No instance variables exist in interface
D. JVM restriction

13. Which of the following is a marker interface?
A. Runnable
B. Serializable
C. Comparable
D. Callable

14. Marker interface is also called:
A. Abstract interface
B. Functional interface
C. Tag interface
D. Nested interface

15. Purpose of marker interface is to:
A. Store data
B. Perform inheritance
C. Provide runtime information to JVM
D. Increase security

16. Interface reference can refer to:
A. Interface object
B. Abstract class object
C. Implemented class object
D. Any class object

17. If reference is of interface type, accessible methods are:
A. All class methods
B. Only interface methods
C. Only static methods
D. Only default methods

18. Which statement correctly explains method access using an interface reference?
A. Interface reference can access all methods of implementing class
B. Interface reference can access only static methods
C. Interface reference can access only methods declared in the interface
D. Interface reference cannot call any method

19. Which concept is BEST demonstrated by ATM example?
A. Encapsulation
B. Abstraction
C. Inheritance
D. Polymorphism

20. Interface acts like RBI because it:
A. Controls execution
B. Stores money
C. Provides rules for implementation
D. Creates objects

Part B: Problem Statements

---

1. ATM Machine System

Problem Statement:
Design an `ATMService` interface that defines common ATM operations such as withdraw, deposit, and balance enquiry. Different banks can implement this interface according to their internal logic while the user interacts with a common ATM screen.

Sample Input:
Withdraw amount = 2000

Sample Output:
"Amount 2000 withdrawn successfully"

---

2. Banking Rules System (RBI Example)

Problem Statement:
Create a `BankRules` interface that defines rules like minimum balance and interest rate. All banks must follow these rules while implementing their own banking services.

Sample Input:
Minimum Balance Check

Sample Output:
"Minimum balance rule applied"

---

3. Payment System

Problem Statement:
Design a `Payment` interface with a `pay()` method. Implement it for Credit Card, UPI, and Net Banking payments so the system can switch payment modes dynamically.

Sample Input:
Payment Mode = UPI, Amount = 500

Sample Output:
"Payment of 500 completed using UPI"

---

4. Notification System

Problem Statement:
Create a `Notification` interface to send alerts. Implement it for Email and SMS so the notification method can change without affecting business logic.

Sample Input:
Message = "Order Confirmed"

Sample Output:
"Email notification sent: Order Confirmed"

---

5. Shape Drawing Application

Problem Statement:
Design a `Shape` interface with a `draw()` method. Different shapes implement the interface to provide their own drawing logic.

Sample Input:
Shape = Circle

Sample Output:
"Drawing Circle"

---

6. Vehicle System

Problem Statement:
Create a `Vehicle` interface with a `start()` method. Implement it for Car and Bike to demonstrate common behavior with different implementations.

Sample Input:
Vehicle = Bike

Sample Output:
"Bike started"

---

7. Database Connectivity

Problem Statement:
Design a `DBConnection` interface so the application can connect to different databases without changing core logic.

Sample Input:
Database = MySQL

Sample Output:
"Connected to MySQL Database"

---

8. Printer Management System

Problem Statement:
Create a `Printer` interface with a `print()` method. Implement it for Laser and Inkjet printers.

Sample Input:
Document = Resume.pdf

Sample Output:
"Printing Resume.pdf using Laser Printer"

---

9. File Handling System

Problem Statement:
Design a `FileOperation` interface for file operations such as read and write. Different file systems can implement it.

Sample Input:
Operation = Read

Sample Output:
"File read successfully"

---

10. Online Shopping Discount System

Problem Statement:
Create a `DiscountPolicy` interface to apply discounts. Different discount strategies can be implemented.

Sample Input:
Amount = 1000, Discount = Festival

Sample Output:
"Final amount after festival discount: 900"

---

11. Employee Salary Calculation

Problem Statement:
Design a `SalaryCalculator` interface for calculating salary of full-time and part-time employees.

Sample Input:
Employee Type = Part-Time, Hours = 20

Sample Output:
"Salary calculated: 4000"

---

12. Login Authentication System

Problem Statement:
Create an `Authenticator` interface to authenticate users using password or OTP methods.

Sample Input:
Login Method = OTP

Sample Output:
"User authenticated using OTP"

---

13. Media Player System

Problem Statement:
Design a `Playable` interface for playing audio and video files.

Sample Input:
Media Type = Audio

Sample Output:
"Playing audio file"

---

14. Cloud Storage Service

Problem Statement:
Create a `StorageService` interface to upload files to different cloud platforms.

Sample Input:
File = data.txt

Sample Output:
"File uploaded to AWS Cloud"

---

15. Logging Framework

Problem Statement:
Design a `Logger` interface to log messages to different destinations such as file or database.

Sample Input:
Log Message = "Login successful"

Sample Output:
"Log stored in file"
