Parth Joshi

Java Basics & OOPs Assignment - Answers

## Java Basics & OOPs Assignment Questions

## 1. Java Basics

## 1. What is Java? Explain its features.

Java is a high-level, object-oriented programming language developed by Sun Microsystems . It is designed to have as few implementation dependencies as possible, making it platform independent.  
Features:  
- Simple and Secure  
- Object-Oriented  
- Platform Independent   
- Robust and Portable  
- High Performance  
- Multithreaded

## 2. Explain the Java program execution process.

1. Write Java code and save it as .java file.  
2. Compile the file using `javac` to produce a .class file   
3. Run the bytecode using the Java Virtual Machine with `java` command.

## 3. Write a simple Java program to display 'Hello World'.

## public class first {

## public static void main(String[] args) {

## System.out.println("Hello, World!");

## }

## }

## 4. What are data types in Java? List and explain them.

Data types specify the size and type of values that can be stored.  
- Primitive Data Types: byte, short, int, long, float, double, char, boolean  
- Non-Primitive Data Types: String, Arrays, Classes, Interfaces

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

- JVM (Java Virtual Machine): Runs Java bytecode on any platform.  
- JRE (Java Runtime Environment): JVM + libraries to run Java applications.  
- JDK (Java Development Kit): JRE + tools to develop Java applications.

## 6. What are variables in Java? Explain with examples.

Variables are containers for storing data values.  
public class Assignment {

   public Assignment() {

   }

   public static void main(String[] var0) {

      int var1 = 10;

      byte var2 = 20;

      var1 += var2;

      System.out.println("The value of a is " + var1);

   }

}

**7. What are the different types of operators in Java?**

- Arithmetic Operators: +, -, \*, /, %  
- Relational Operators: ==, !=, >, <, >=, <=  
- Logical Operators: &&, ||, !  
- Assignment Operators: =, +=, -=, etc.  
- Bitwise Operators: &, |, ^, ~, <<, >>

## 8. Explain control statements in Java (if, if-else, switch).

- if: Executes if condition is true.  
- if-else: Executes one block if true, another if false.  
- switch: Selects for execution one statement from multiple options based on a value.

## 9. Write a Java program to find whether a number is even or odd.

public class EvenOdd {  
 public static void main(String[] args) {  
 int num = 4;  
 if (num % 2 == 0)  
 System.out.println("Even");  
 else  
 System.out.println("Odd");  
 }  
}

## 10. What is the difference between while and do-while loop?

- while: Condition is checked before loop body.  
- do-while: Loop body runs once before checking condition.

## 2. Object-Oriented Programming (OOPs)

Answer not available.

## 1. What are the main principles of OOPs in Java? Explain each.

1. Encapsulation: Binding data and code together.  
2. Inheritance: One class inherits fields/methods from another.  
3. Polymorphism: One interface, many implementations.  
4. Abstraction: Hiding internal details and showing functionality.

## 2. What is a class and an object in Java? Give examples.

- Class: Blueprint for objects. Example:  
  
class Car {  
 String color;  
}  
```  
- Object: Instance of a class. Example:  
  
Car c = new Car();

## 3. Write a program using class and object to calculate area of a rectangle.

class Rectangle {  
 int length, breadth;  
  
 Rectangle(int l, int b) {  
 length = l;  
 breadth = b;  
 }  
  
 int area() {  
 return length \* breadth;  
 }  
  
 public static void main(String[] args) {  
 Rectangle r = new Rectangle(5, 3);  
 System.out.println("Area: " + r.area());  
 }  
}

## 4. Explain inheritance with real-life example and Java code.

Inheritance means one class (child) inherits properties of another (parent).  
Example:  
  
class Animal {  
 void sound() {  
 System.out.println("Animal sound");  
 }  
}  
  
class Dog extends Animal {  
 void bark() {  
 System.out.println("Bark");  
 }  
}

## 5. What is polymorphism? Explain with compile-time and runtime examples.

Polymorphism allows one interface to be used for different data types.  
- Compile-time (method overloading):  
  
void show(int a) {}  
void show(String a) {}  
  
- Runtime (method overriding):  
  
class A { void display() {} }  
class B extends A { void display() {} }

## 6. What is method overloading and method overriding? Show with examples.

- Overloading: Same method name, different parameters.  
  
void show(int a) {}  
void show(String a) {}  
  
- Overriding: Subclass provides specific implementation.  
  
class A { void show() {} }  
class B extends A { void show() {} }

## 7. What is encapsulation? Write a program demonstrating encapsulation.

Encapsulation is wrapping data and code into a single unit.  
  
class Student {  
 private String name;  
  
 public void setName(String n) { name = n; }  
 public String getName() { return name; }  
}

## 8. What is abstraction in Java? How is it achieved?

Abstraction hides internal details and shows essential features.  
Achieved by:  
- Abstract Classes  
- Interfaces

## 9. Explain the difference between abstract class and interface.

- Abstract class: Can have method body and constructor. Use `abstract` keyword.  
- Interface: Only abstract methods (Java 8+ supports default methods).  
One class can implement multiple interfaces but can extend only one abstract class.

## 10. Create a Java program to demonstrate the use of interface.

interface Animal {  
 void sound();  
}  
  
class Dog implements Animal {  
 public void sound() {  
 System.out.println("Bark");  
 }  
}