1. Constructors and Initialization:  
 a) What is the purpose of a constructor in Java? How is it different from a regular method?  
 b) Can a class have multiple constructors? If yes, explain the concept of constructor overloading.

Ans: constructor is a method to initialize subject of a class.

b. yes,you can create different object with different parameters.

2. Access Modifiers (public, private, protected):  
 a) Explain the difference between public, private, and protected access modifiers in Java.  
 b) Why is it important to use access modifiers in your class attributes and methods? Provide an example scenario.

Public can be accesed by anyone.private only within same class. Protected can be accesed by same class,package.

3. Method Overloading and Overriding:  
 a) What is method overloading in Java? Provide an example of a class with overloaded methods.  
 b) What is method overriding in Java? Explain how it enables polymorphism in object-oriented programming.

a. method overloading is defining Mutiple method with same name but different parameters.

public class Calculator {

public int add(int a, int b)

return a + b;

}

public int add(int a, int b, int c) {

return a + b + c;

}

b. overiding means when the subclass overides the method.polymorphism allows different classes to be treated as same whci helps to re use code.

4. Inheritance and its types:  
 a) What is inheritance in Java? How does it promote code reuse and extensibility?  
 b) Explain the difference between single inheritance and multiple inheritance. Provide an example scenario for each.

Inheritance is when a class can acquire properties of different class.it promotes code reuse and extensibilty by allowing to use properties of different class.

b. the difference is the amount of subclasses each can inherit from.

class Vehicle {

protected String brand;

public void drive() {

System.out.println("Driving the vehicle");

}

}

class Car extends Vehicle {

private int numberOfSeats;

public void park() {

System.out.println("Parking the car");

}

}

public class Main {

public static void main(String[] args) {

Car car = new Car();

car.brand = "Toyota"; // Inherited from Vehicle class

car.drive(); // Inherited from Vehicle class

car.park(); // Specific to Car class

}

}

Examples of multiple inheritance:

class Animal {

public void eat() {

System.out.println("The animal is eating");

}

}

class Flyable {

public void fly() {

System.out.println("The object is flying");

}

}

class Bird extends Animal, Flyable { // Error: Multiple inheritance not supported

// ...

}

5. Interfaces and Abstract Classes:  
 a) What is an interface in Java? How is it different from an abstract class?  
 b) When would you use an abstract class versus an interface in your program? Provide an example scenario for each.

Interface defines a contract of methods that a class implementing it needs to follow.it is different from abstract because inferface only declares methods and while abstract method can contain both.

abstract class Shape {

protected int numSides;

public Shape(int numSides) {

this.numSides = numSides;

}

public abstract double calculateArea();

public void printDetails() {

System.out.println("Number of sides: " + numSides);

System.out.println("Area: " + calculateArea());

}

}

class Rectangle extends Shape {

private double width;

private double height;

public Rectangle(double width, double height) {

super(4);

this.width = width;

this.height = height;

}

@Override

public double calculateArea() {

return width \* height;

}

}

public class Main {

public static void main(String[] args) {

Rectangle rectangle = new Rectangle(5, 3);

rectangle.printDetails();

}

}

interface Playable {

void play();

}

class Guitar implements Playable {

@Override

public void play() {

System.out.println("Playing the guitar");

}

}

class Piano implements Playable {

@Override

public void play() {

System.out.println("Playing the piano");

}

}

public class Main {

public static void main(String[] args) {

Playable guitar = new Guitar();

Playable piano = new Piano();

guitar.play(); // Output: Playing the guitar

piano.play(); // Output: Playing the piano

}

}

6. Packages and Imports:  
 a) What is the purpose of packages in Java? How do they organize and structure your code?  
 b) How would you import a class from a different package? Provide an example of importing a class and using its methods.

The purpose of packages in java is used to organize and structure code.

To import the package, we can use keyword “import “ statement.

import com.example.utils.MathUtils;

public class Main {

public static void main(String[] args) {

int sum = MathUtils.add(5, 3);

int product = MathUtils.multiply(4, 2);

System.out.println("Sum: " + sum); // Output: Sum: 8

System.out.println("Product: " + product); // Output: Product: 8

}

}