Java QB a **l** units

2MQ

Q1 Name three features of the Java programming language.

* Simple and Easy to Learn.
* Object-Oriented Programming
* High Performance

Q2 What are the primitive data types in Java?

primitive data types are the basic building blocks of data Such as boolean

, char, int, short, byte, long, float, and double. The Boolean with uppercas e B is a wrapper class for the primitive data type boolean in Java.

Q3 Define a constructor in Java.

A constructor is a method that sets up an object when it's created. Or

A constructor in Java is a special method that is used to initialize objects. The constructor is called when an object of a class is created.

Q4 What is the purpose of ‘this’ keyword? The `this` keyword in Java is used to:

1. Refer to the current object of the class.
2. Distinguish between instance variables and local variables with the sa me name.

Q5 Describe any two differences between a local variable and an instance variable.

1. \*Scope\*: Local variables are limited to the method/block, while instanc e variables are accessible throughout the class.
2. \*Lifetime\*: Local variables exist only during method execution, while ins tance variables exist as long as the object exists.

Q6 Write a short Java code snippet to declare and initialize an integer var iable and a String variable.

short Java code

int age = 25;

String name = "John";

This code declares and initializes an integer variable `age` with the value

25 and a String variable `name` with the value "John".

Q7 List all four Visibility Modifiers in Java. The access modifiers in Java are

Public Protected Default Private

These modifiers are keywords that determine the accessibility of classes, methods, variables, and more

Q8 Java is architecturally neutral. Justify. Java is architecturally neutral because:

1. Java code is compiled into bytecode, which is platform-independent.
2. Bytecode can run on any platform with a Java Virtual Machine (JVM), wi thout recompilation.

Q9 Define instantiation with an example.

Instantiation is the process of creating an object of a class using the `new` keyword.

Example: `Car myCar = new Car();`

This creates an instance of the `Car` class.

Q10 What is the use of “this” and ” super” keyword?

* `this`: Refers to the current object, used to access its members.
* `super`: Refers to the parent class, used to access its members.

Q11 List out the operators in Java Java operators include:

1. Arithmetic (`+`, `-`, `\*`, `/`, `%`)

1. Assignment (`=`, `+=`, `-=`, etc.)
2. Comparison (`==`, `!=`, `>`, `<`, etc.)
3. Logical (`&&`, `||`, `!`)
4. Bitwise (`&`, `|`, `^`, etc.)

Q12 How would you differentiate local and instance variables?

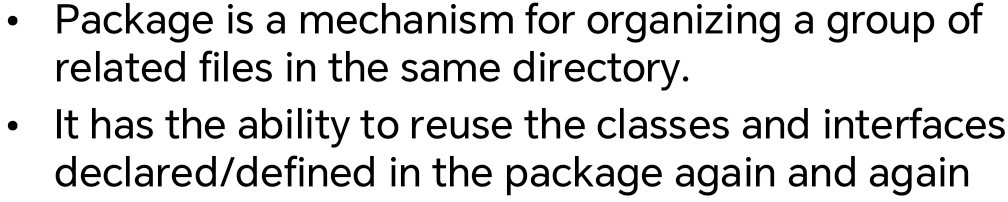
\*Local Variables:\*

* + Declared inside a method/block.
  + No default value.

\*Instance Variables:\*

* + Declared inside a class, outside methods.
  + Have default values.

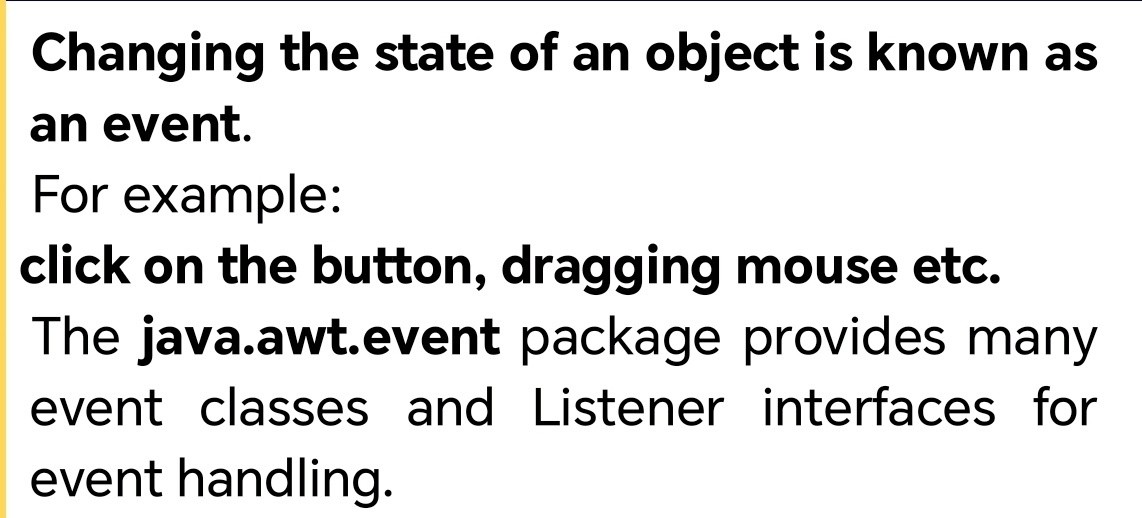
Q13 Elaborate the purpose of package.



Q14 Why do you use Scanner class in java?

The `Scanner` class is used to get input from console, files, or strings allo wing you to read and parse.

Q15 What is an event in Java?



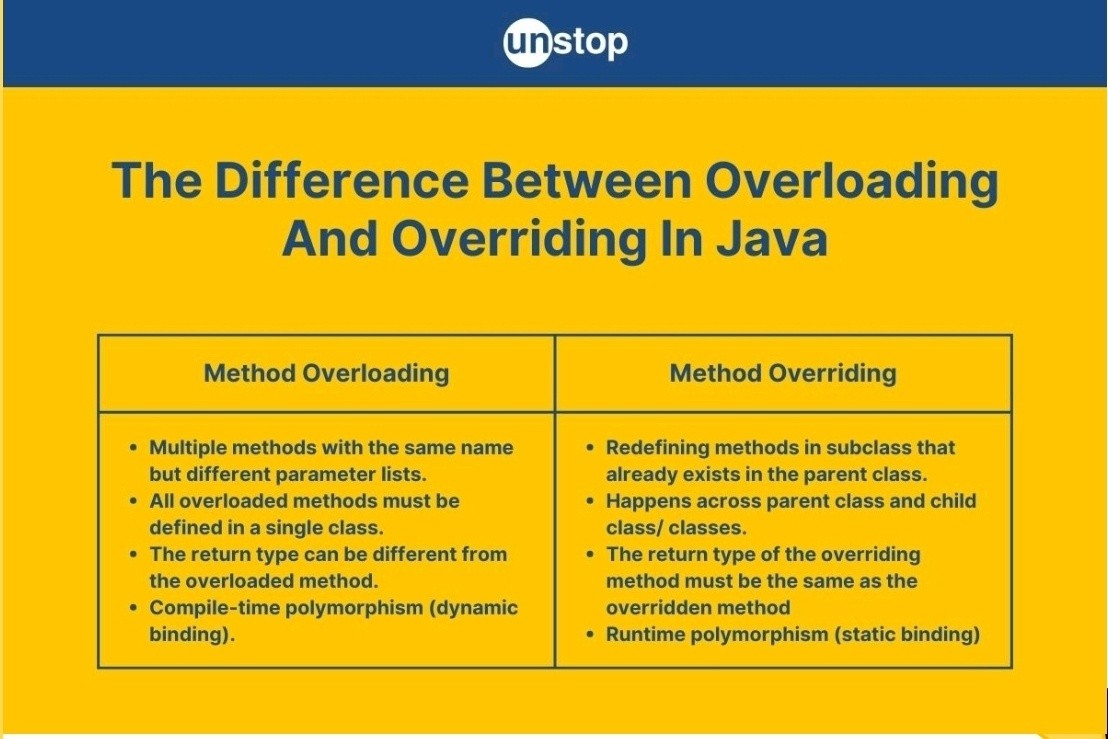
Q16 What is the use of Wrapper class?

Wrapper classes convert primitive data types (like int, char) into objects, enabling them to be used in collections and other object-based framewor

ks.

Examples: `Integer`, `Character`. 5MQ

Q1 Differentiate between Method Overloading and Method Overriding.



Q2 What is the significance of the static keyword in Java?

The `static` keyword in Java is used to associate a variable or method wit h a class, rather than an instance. This means:

* + \*Static variables\*: Shared by all instances of the class.
  + \*Static methods\*: Can be called without creating an instance of the clas s.

Q3 What are the advantages and disadvantages of using arrays in Java?

\*Advantages of Arrays:\*

1. Fast access to elements using index.
2. Memory-efficient for storing multiple values.

\*Disadvantages of Arrays:\*

1. Fixed size that cannot be changed dynamically.
2. Can store only homogeneous data (same data type).

Q4 Write a Java program to concatenate two strings using the StringBuff er class.

public class StringBufferConcat {

public static void main(String[] args) { StringBuffer sb = new StringBuffer("Hello"); sb.append(" World");

System.out.println(sb.toString()); // Output: Hello World

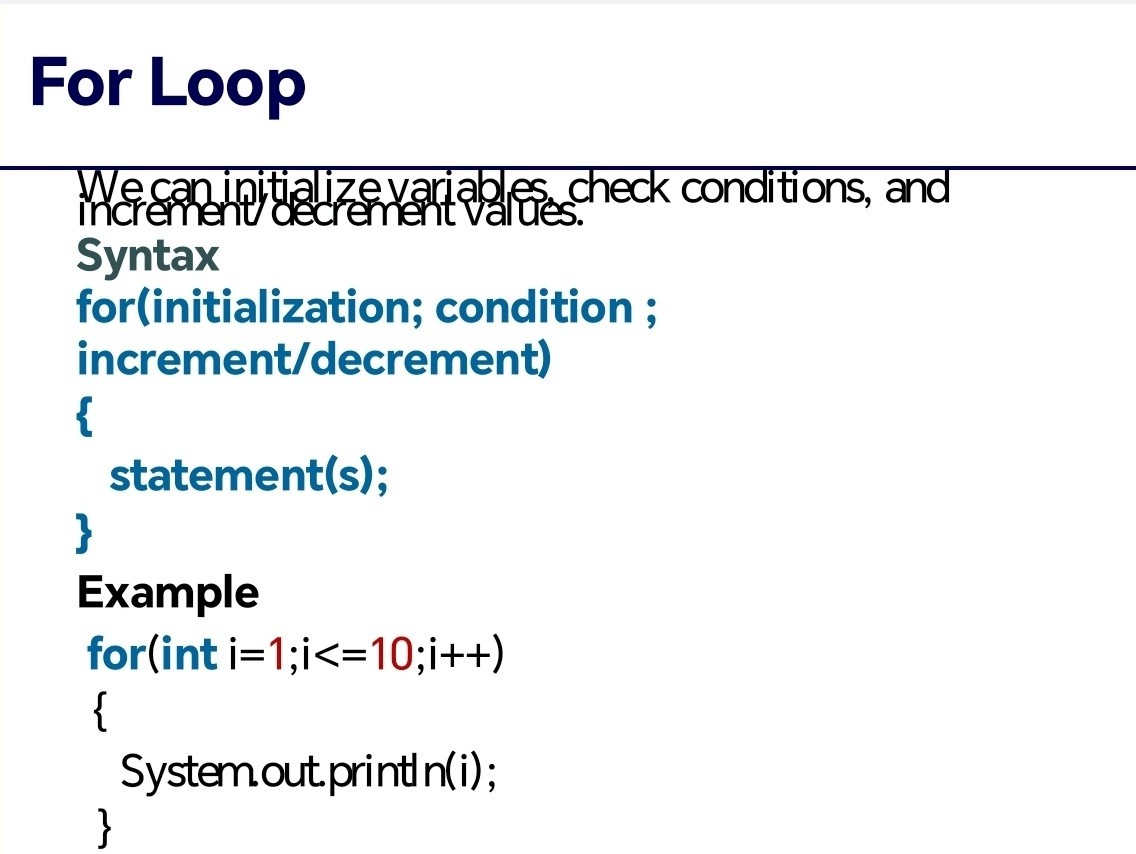
}

}

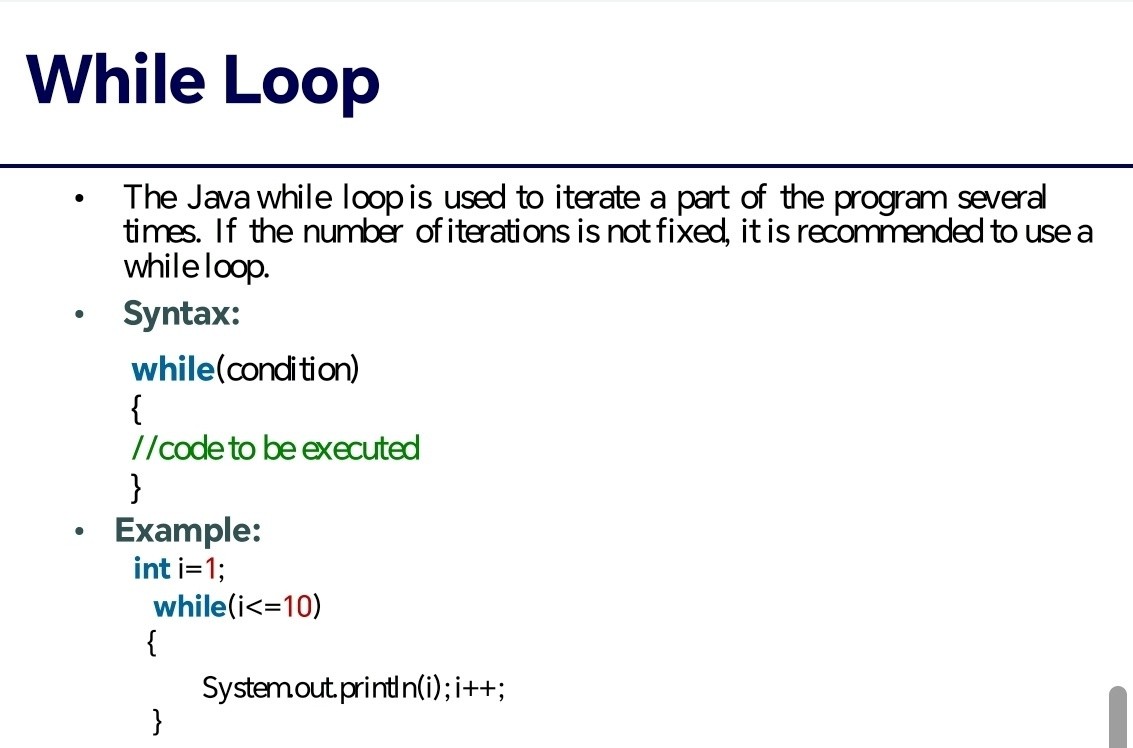
Q5 Compare and contrast the use of for loops, while loops, and do-while l oops.

\*Comparison of Loops:\*

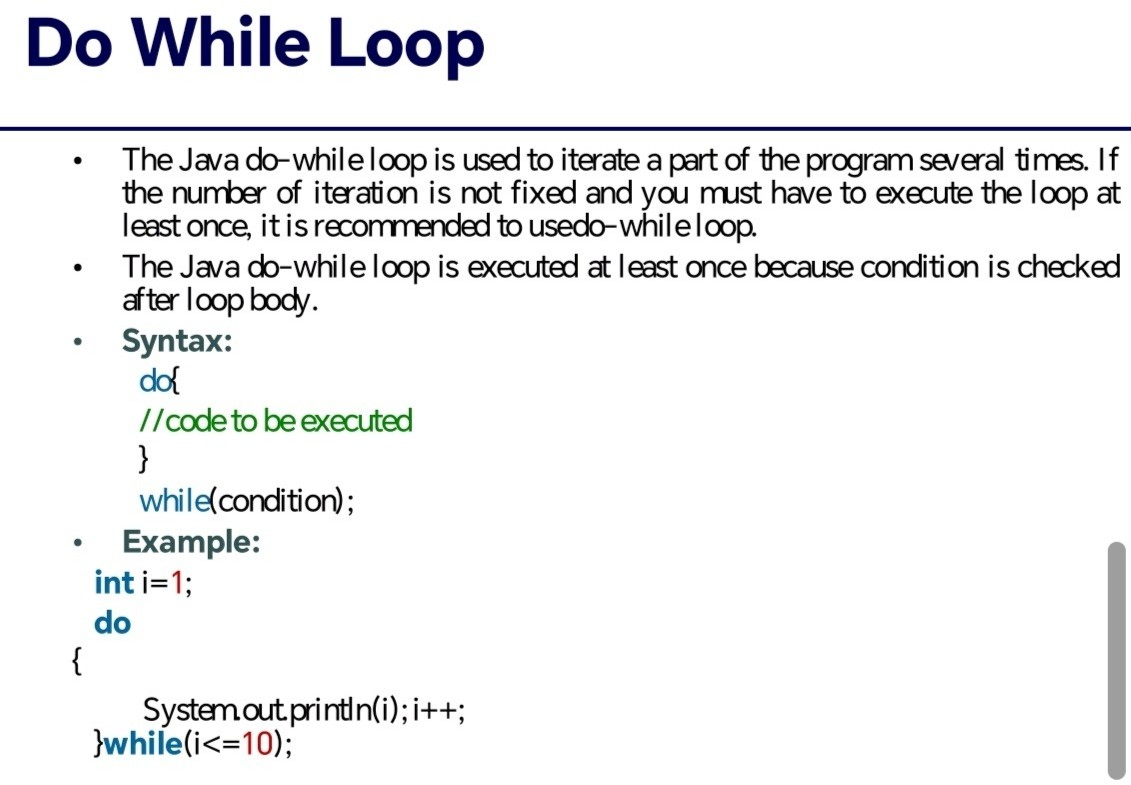
1. \*For Loop\*: Used when the number of iterations is known.



1. \*While Loop\*: Used when the number of iterations is unknown.



1. \*Do-While Loop\*: Similar to while loop, but the body is executed at lea st once.



\*Key differences:\*

* + \*For loop\*: Initialization, condition, and increment/decrement are in one line.
  + \*While loop\*: Condition is checked before executing the loop body.
  + \*Do-while loop\*: Loop body is executed at least once before checking t he condition.

Q6 Write a Java program to print the numbers from 1 to 10 using a for loo p.

public class PrintNumbers {

public static void main(String[] args) { for (int i = 1; i <= 10 ; i++) {

System.out.println(i);

}

}

}

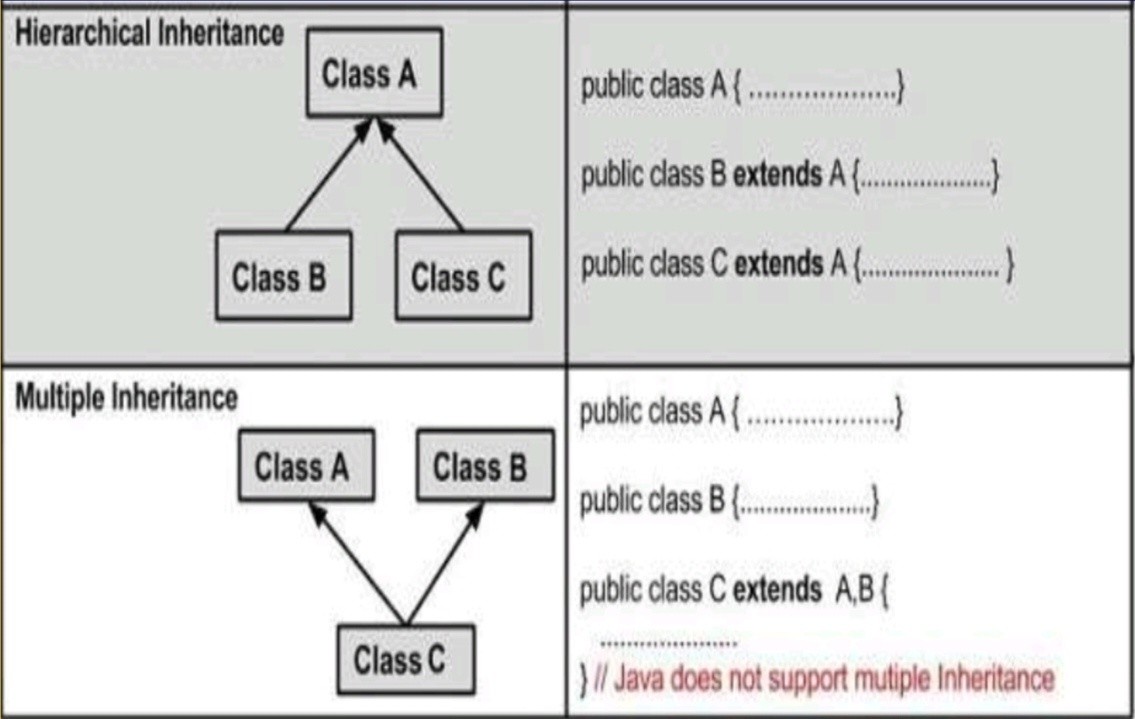
Q7 Elaborate Inheritance and its types in Java with an appropriate examp le.

●

Inheritance: A child class gets properties and methods from a parent clas

s.





Q8 Compare and contrast the effects of using final with variables, method

s, and classes.

\*Final Keyword in Java:\*

1. \*Final Variables\*:
   * Value cannot be changed once assigned.
   * Must be initialized when declared or in a constructor.
2. \*Final Methods\*:
   * Cannot be overridden in subclasses.
   * Useful for ensuring method implementation remains unchanged.
3. \*Final Classes\*:
   * Cannot be subclassed or extended.
   * Useful for security or when a class's implementation should not be mo dified.

\*Key effects:\*

* \*Variables\*: Constant values.
* \*Methods\*: Prevents method overriding.
* \*Classes\*: Prevents inheritance.

Q9 Design an interface called Drawable with methods for drawing a shap

e. Create two classes, Circle and Rectangle, that implement the Drawable interface.

Program

// Drawable interface interface Drawable {

void draw();

double getArea();

}

// Circle class implementing Drawable class Circle implements Drawable {

private double radius;

public Circle(double radius) { this.radius = radius;

}

@Override

public void draw() {

System.out.println("Drawing a circle with radius " + radius);

}

@Override

public double getArea() {

return Math.PI \* radius \* radius;

}

}

// Rectangle class implementing Drawable class Rectangle implements Drawable {

private double width; private double height;

public Rectangle(double width, double height) { this.width = width;

this.height = height;

}

@Override

public void draw() {

System.out.println("Drawing a rectangle with width " + width + " and height " + height);

}

@Override

public double getArea() { return width \* height;

}

}

// Main class to test public class Main {

public static void main(String[] args) {

Drawable circle = new Circle(5.0 ); circle.draw();

System.out.println("Circle area: " + circle.getArea());

Drawable rectangle = new Rectangle(4.0 , 6.0 );

rectangle.draw();

System.out.println("Rectangle area: " + rectangle.getArea());

}

}

Q10 Explain any 5 features of Java. 5 key features of Java:

1. \*Platform Independence\*: Java code can run on any platform with a JV M.
2. \*Object-Oriented\*: Java follows OOP principles like encapsulation, inhe ritance, and polymorphism.
3. \*Garbage Collection\*: Automatic memory management through garba ge collection.
4. \*Multithreading\*: Supports concurrent execution of multiple threads.
5. \*Robust Security\*: Built-in security features to prevent common progra mming errors.

Q11 How would you define inheritance? Explain access modifiers.

* + Inheritance: A child class gets properties and methods from a pare nt class.
  + The access modifiers in Java are
  + Public
  + Protected
  + Default
  + Private
  + These modifiers are keywords that determine the accessibility of cl asses, methods, variables, and more

Q12 Write a program to demonstrate method overloading.



Q13 Discuss the relationship between Abstract classes and Interfaces.

* + Abstract Class: can have fields(properties) and methods
  + Interface: An interface is a blueprint of a class. The interface is a m

echanism to achieve abstraction in java.

Q14 What is exception? Explain different exception handlers in brief.

\*Exception:\*

An exception is an event that disrupts the normal flow of a program's inst ructions.

\*Exception Handlers:\*

1. \*try\*: Code that might throw an exception.
2. \*catch\*: Handles the exception.
3. \*finally\*: Executes code regardless of exceptions.
4. \*throw\*: Throws an exception explicitly.
   1. \*throws\*: Declares potential exceptions in a method.

Q15 Write a program to read data from a text file and write it to another f ile.

import java.io.FileReader; import java.io.FileWriter; import java.io.IOException;

public class FileReadWrite {

public static void main(String[] args) {

try (FileReader reader = new FileReader("source.txt"); FileWriter writer = new FileWriter("destination.txt")) {

int character;

while ((character = reader.read()) != -1) { writer.write(character);

}

System.out.println("Data copied successfully.");

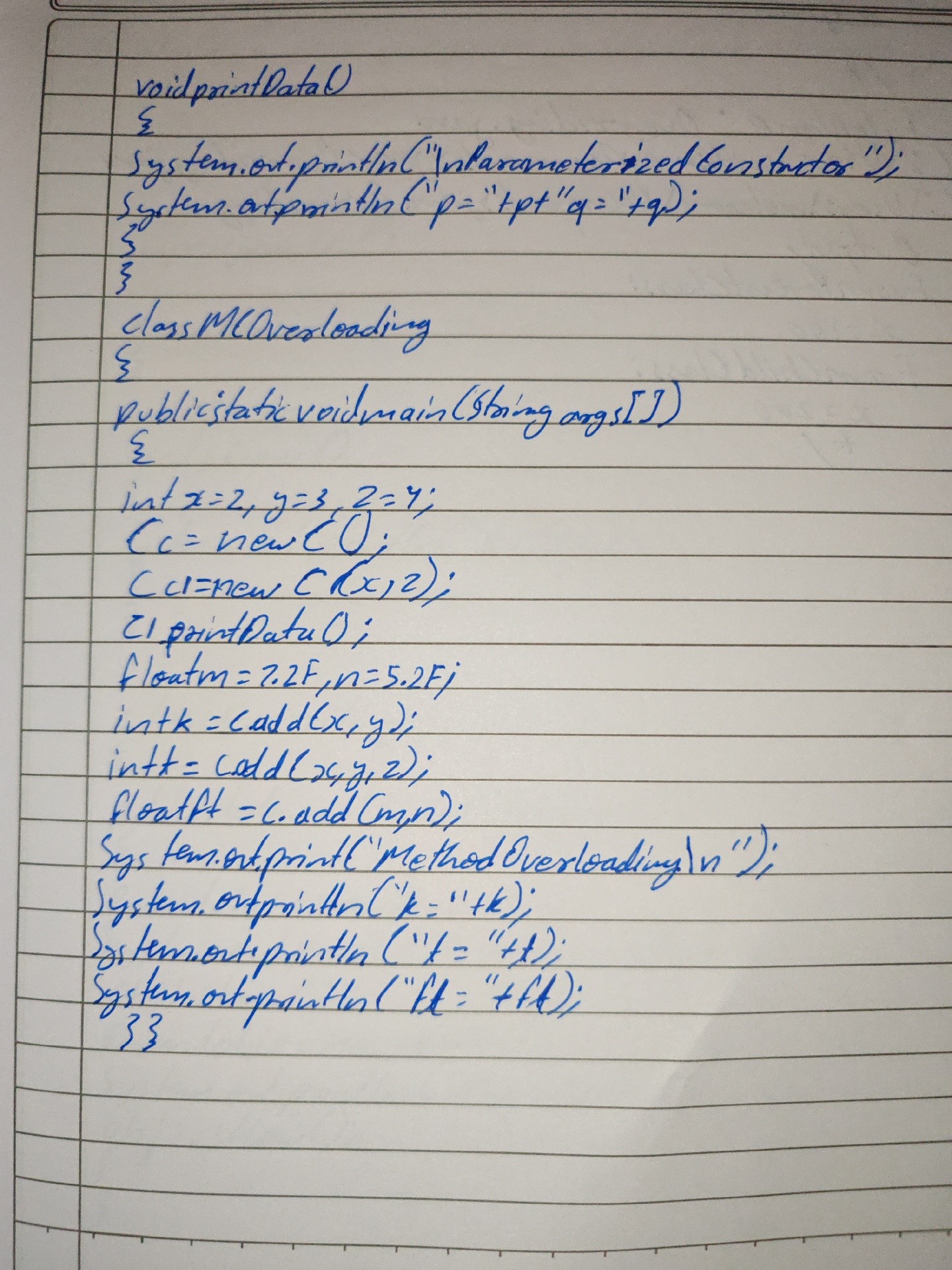
} catch (IOException e) { System.out.println("Error: " + e.getMessage());

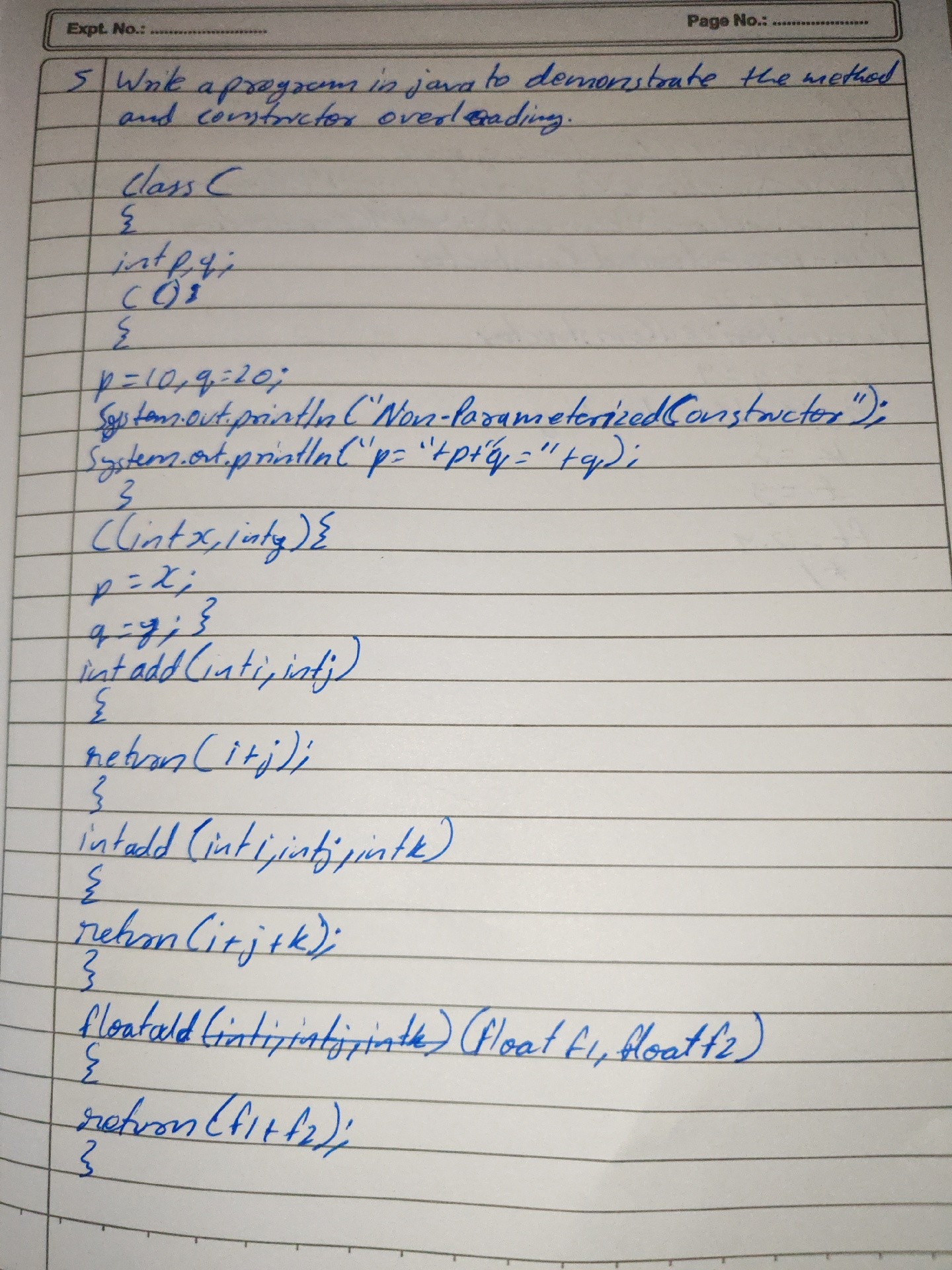
}

}

}

Q16 Write a program to demonstrate constructors.





Q17 What is an array? List out its advantages and disadvantages

\*Array:\*

An array is a collection of elements of the same data type stored in contig

uous memory locations.

\*Advantages:\*

1. \*Efficient storage\*: Arrays store multiple values in a single variable.
2. \*Fast access\*: Elements can be accessed directly using their index.
3. \*Cache-friendly\*: Contiguous memory allocation improves performanc e.

\*Disadvantages:\*

1. \*Fixed size\*: Arrays have a fixed size that cannot be changed dynamica lly.
2. \*Homogeneous data\*: Arrays can only store elements of the same data type.
   1. \*Memory waste\*: If not fully utilized, arrays can waste memory spa ce.

Q18 Discuss the different types of mouse events in Java

Mouse Events\*: Generated by mouse actions, such as clicks, movements, and drags.

\*Mouse Events in Java:\*

1. \*MouseListener Events:\*
   * mousePressed
   * mouseReleased
   * mouseClicked
   * mouseEntered
   * mouseExited
2. \*MouseMotionListener Events:\*
   * mouseDragged
   * mouseMoved

Q19 Write a program to print student details (name, class, age, gender, fe es) by accepting input using Scanner class

import java.util.Scanner;

public class StudentDetails {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter student name: "); String name = scanner.nextLine();

System.out.print("Enter student class: "); String className = scanner.nextLine();

System.out.print("Enter student age: "); int age = scanner.nextInt();

scanner.nextLine(); // Consume newline left-over

System.out.print("Enter student gender: "); String gender = scanner.nextLine();

System.out.print("Enter student fees: "); double fees = scanner.nextDouble();

System.out.println("\nStudent Details:"); System.out.println("Name: " + name); System.out.println("Class: " + className); System.out.println("Age: " + age); System.out.println("Gender: " + gender); System.out.println("Fees: " + fees);

scanner.close();

}

}

Q20 WAP to demo passing parameters to an applet. import java.applet.Applet;

import java.awt.Graphics;

/\*

<applet code="ParamApplet" width="30 0 " height="20 0 ">

<param name="name" value="John Doe">

<param name="age" value="30 ">

</applet>

\*/

public class ParamApplet extends Applet {

String name; int age;

public void init() {

name = getParameter("name");

age = Integer.parseInt(getParameter("age"));

}

public void paint(Graphics g) { g.drawString("Name: " + name, 20, 50 );

g.drawString("Age: " + age, 20 , 70 );

}

}

7/8/10 marks

Q1 Discuss different control structures.

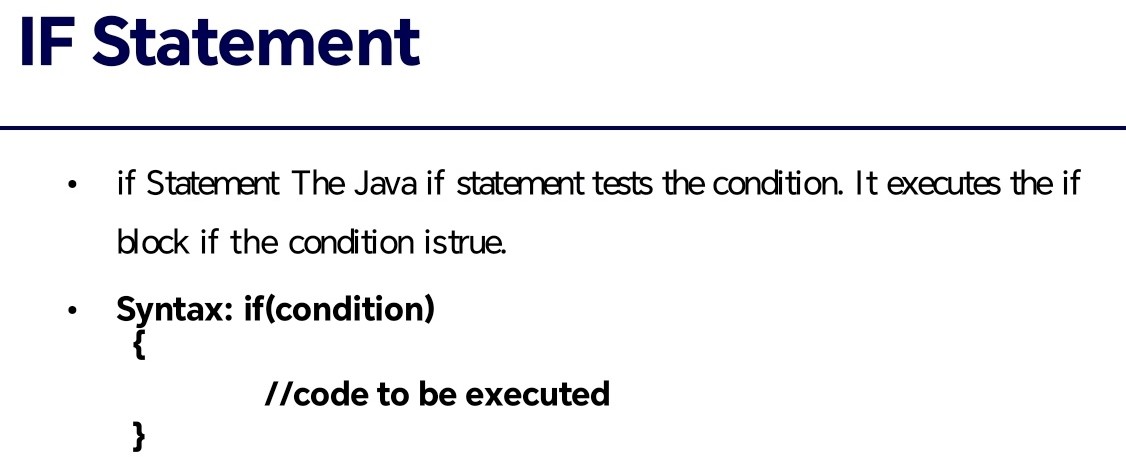
* Control Flow Statements

Control flow statements in Java determine the order in which the code is

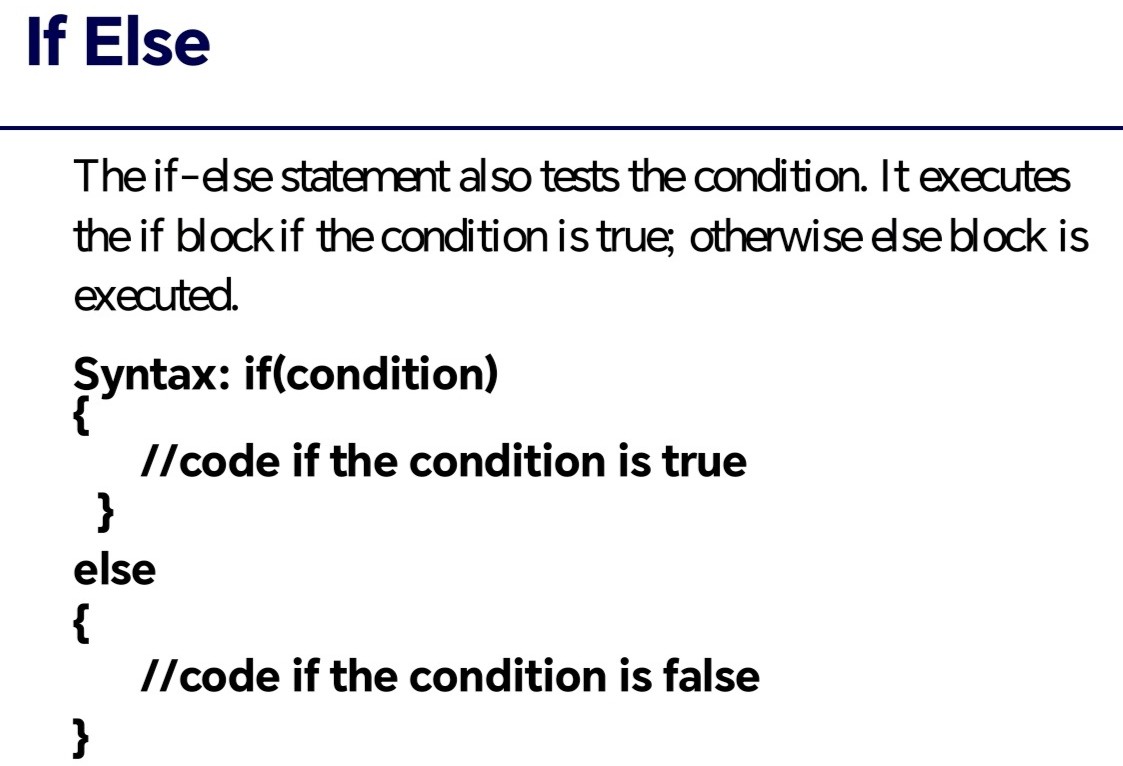
executed. They include conditional statements, loops, and jump statemen ts.

# Selection(conditional) Statements

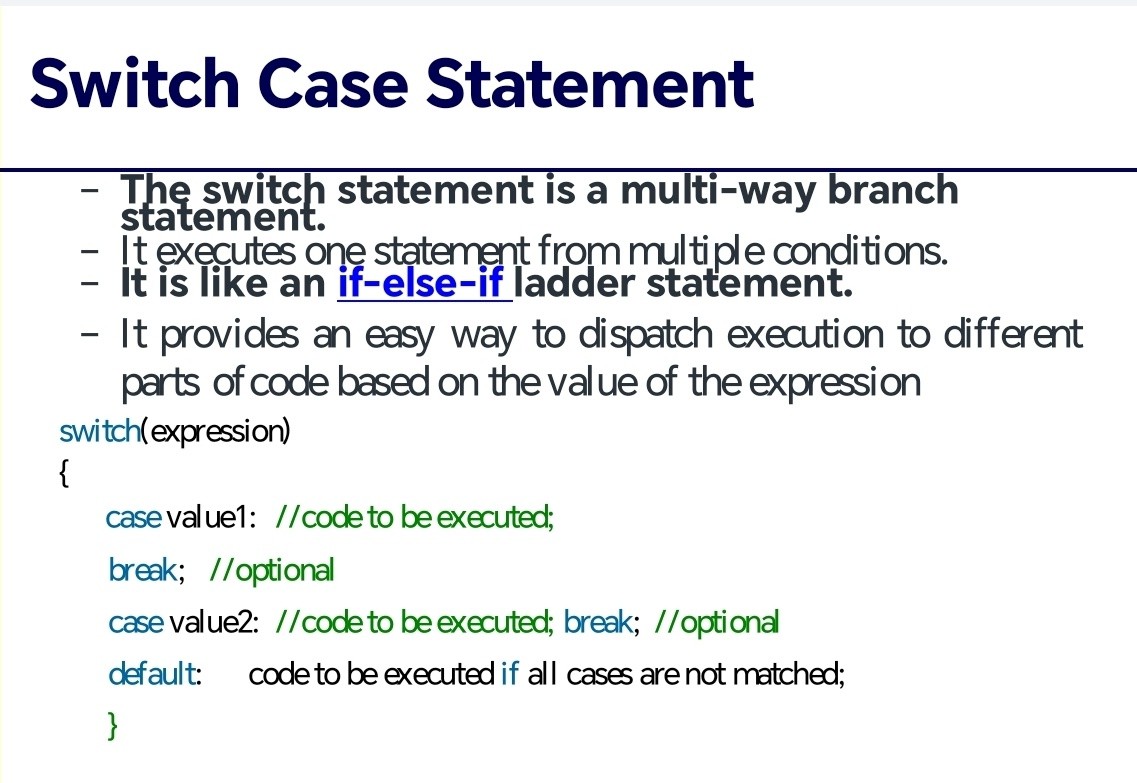
1. If Statement: Used for a single condition.



1. If-Else Statement: Used for two conditions.

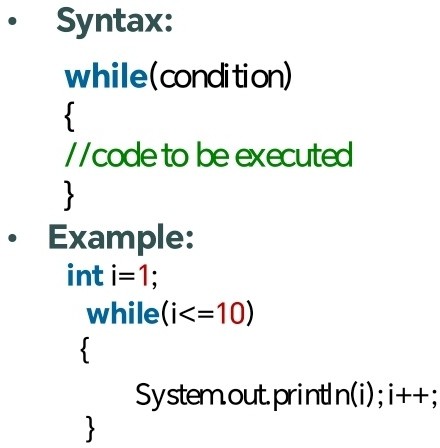


1. Switch Statement: Used for multiple conditions

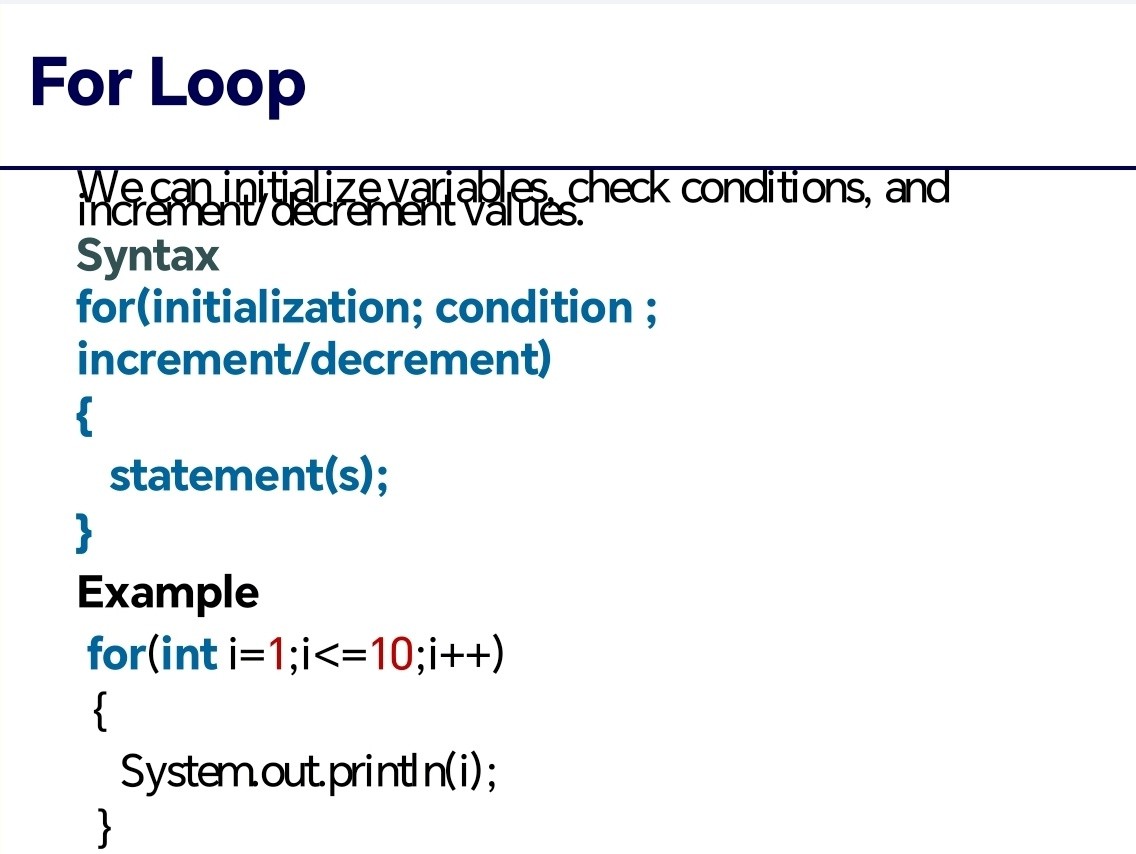


# Iteration Statements

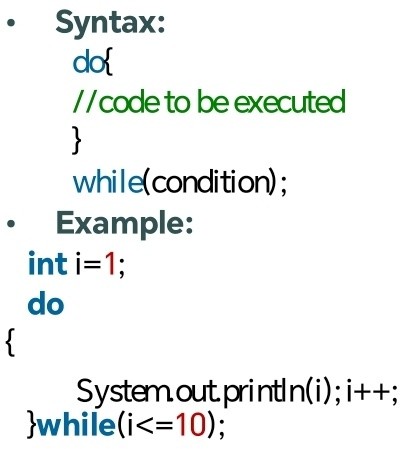
1. While Loop: Repeats while a condition is true.



1. For Loop: Repeats with a counter variable.

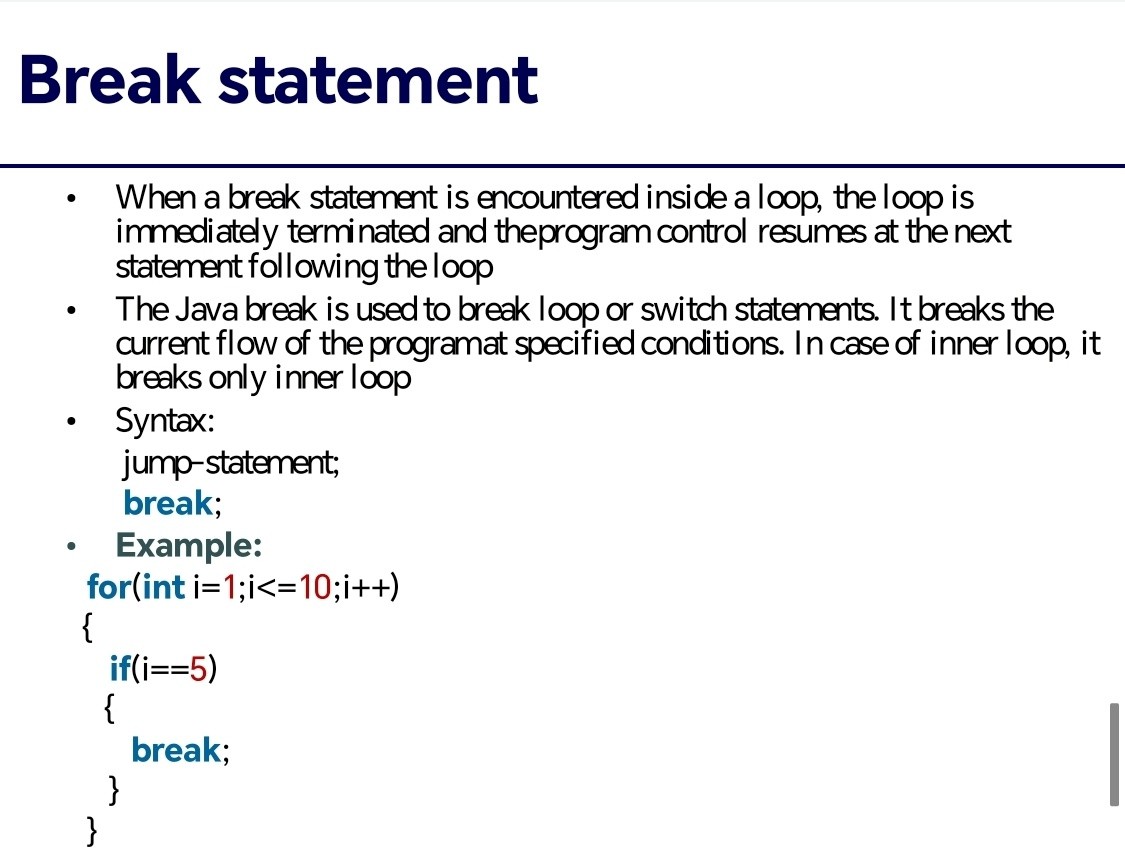


1. Do-While Loop: Repeats while a condition is true.

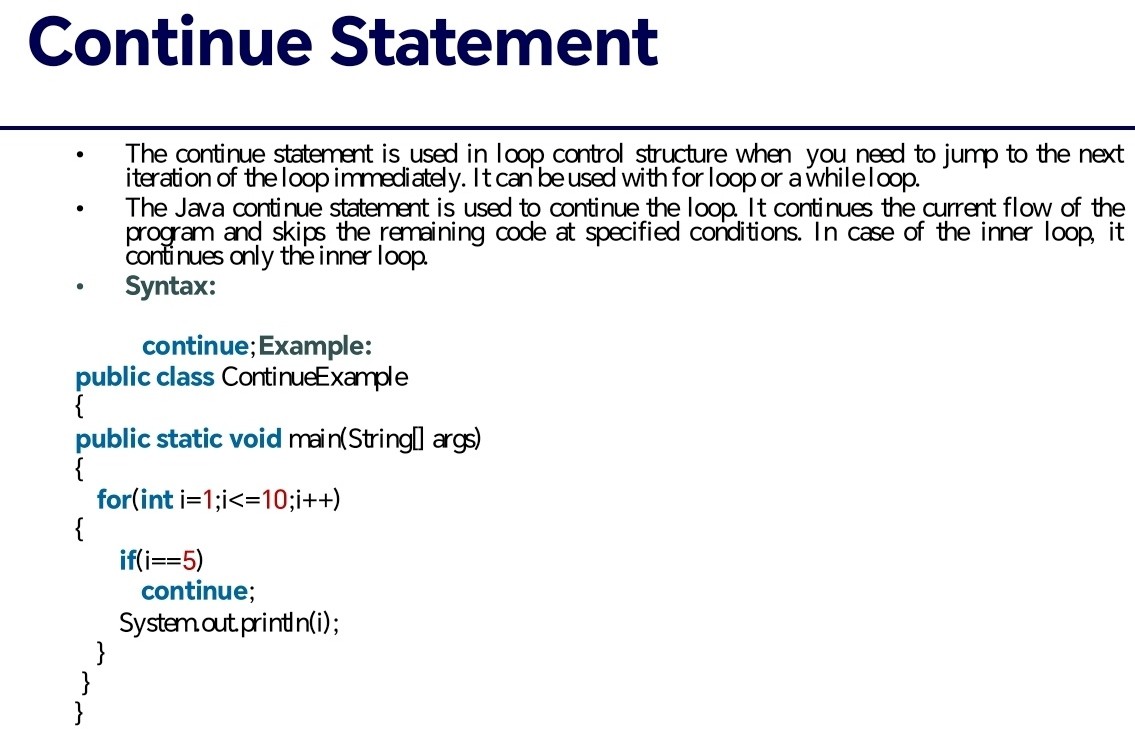


Jump Statements

1. Break Statement: Exits a loop or switch statement.



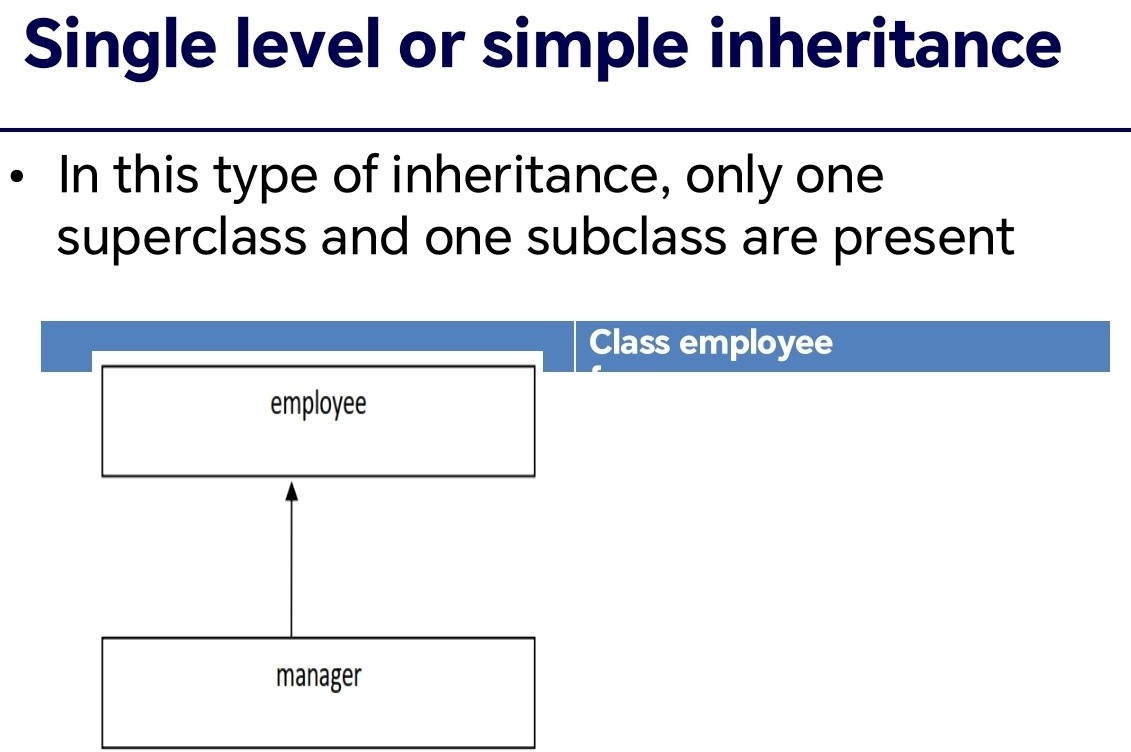
1. Continue Statement: Skips the current iteration.



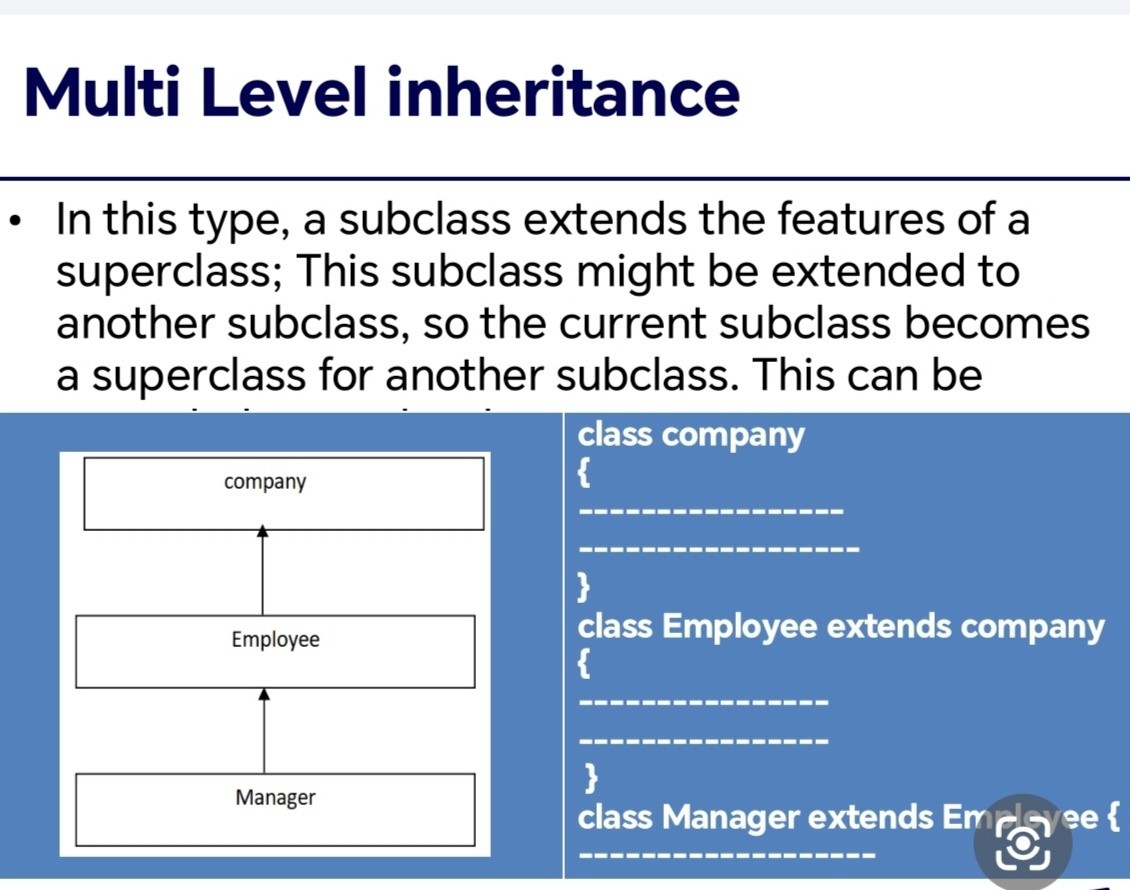
* 1. Return Statement: Exits a method and returns a value.

Q2 Explain various inheritances with suitable examples.

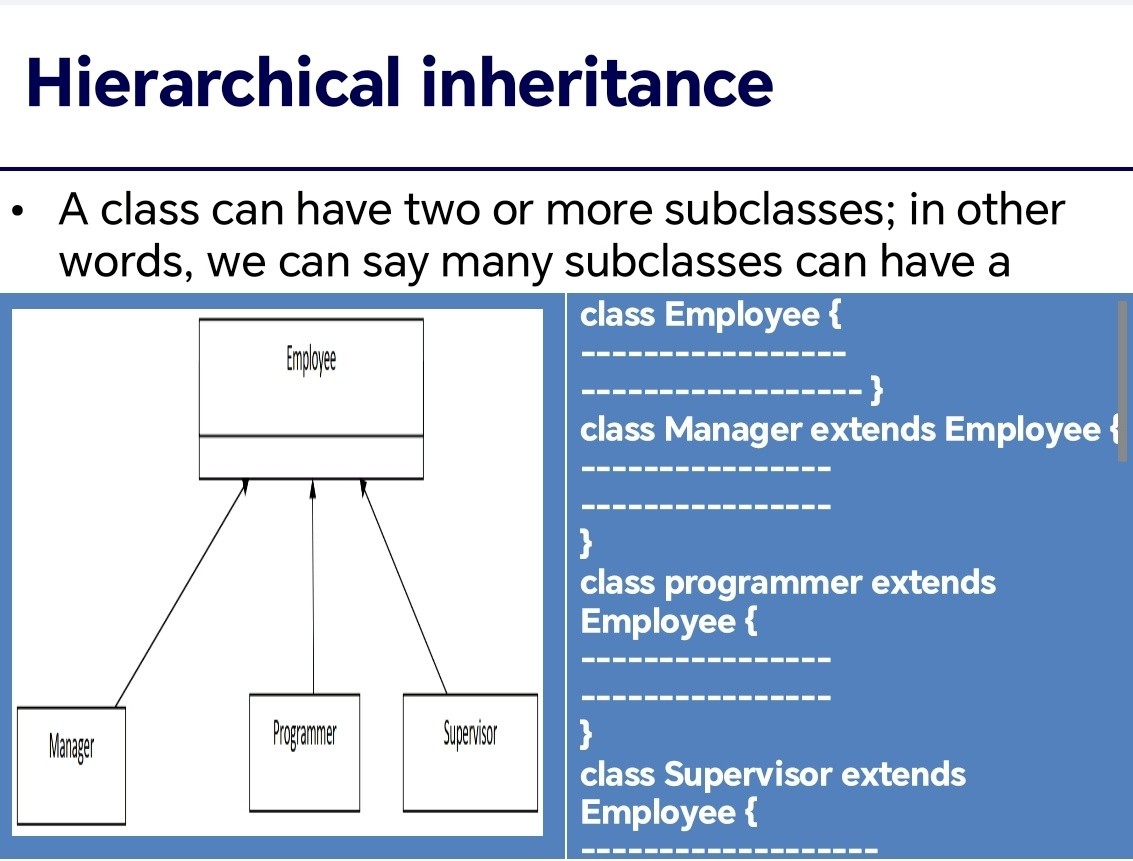
Types of Inheritance: - Single: One child class gets from one parent class



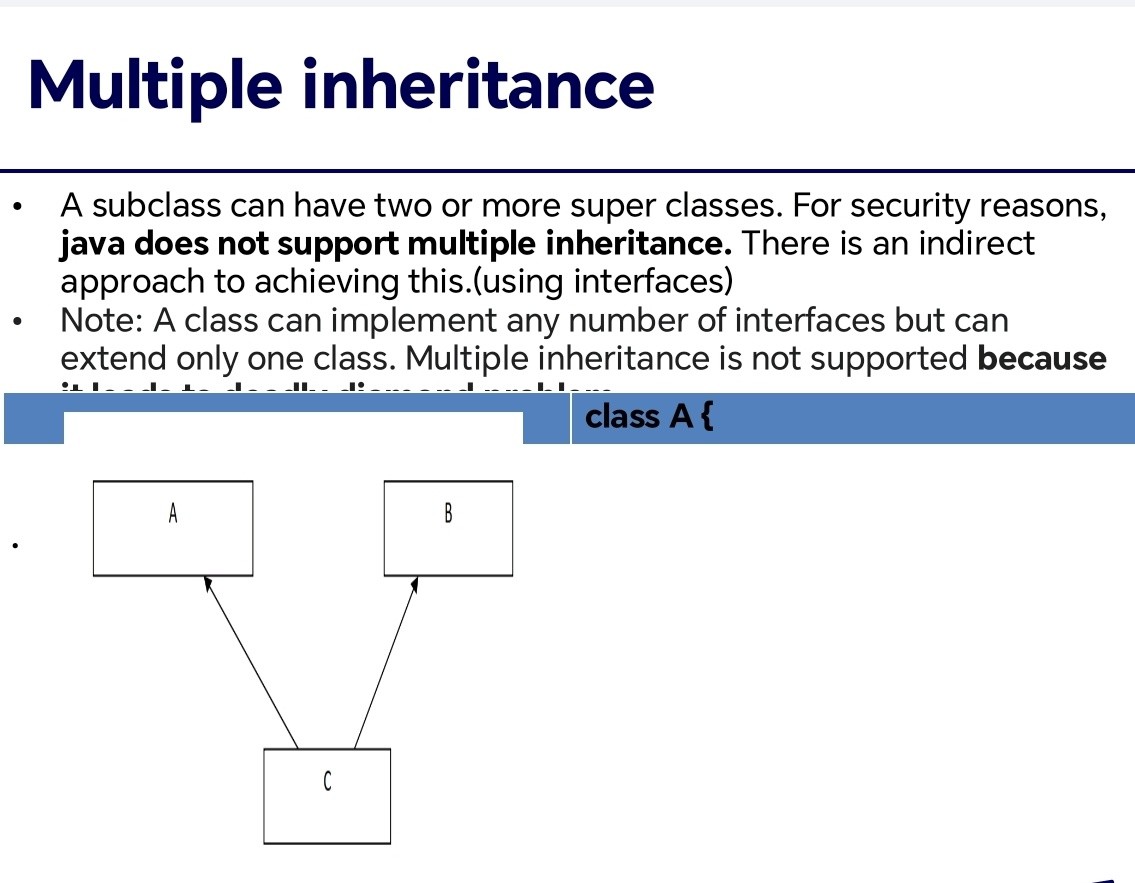
* Multilevel: Child class gets from parent class, which gets from another p arent class



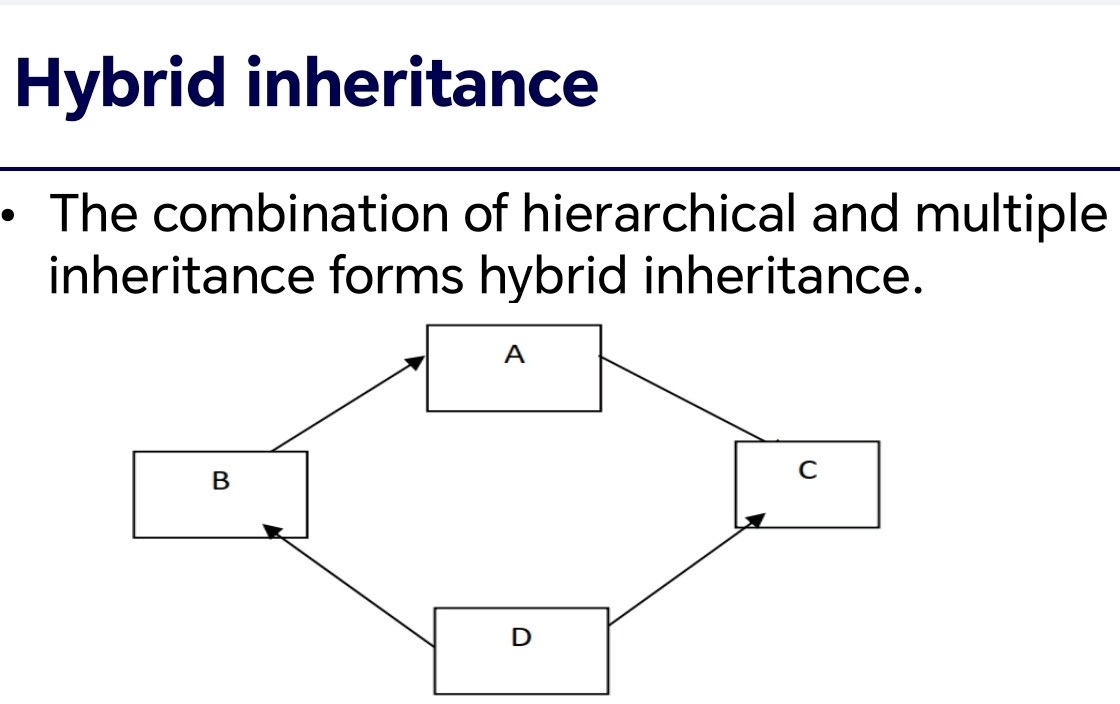
* Hierarchical: Multiple child classes get from one parent class



Multiple Inheritance: Subclass inherits from multiple superclasses (throug h interfaces).



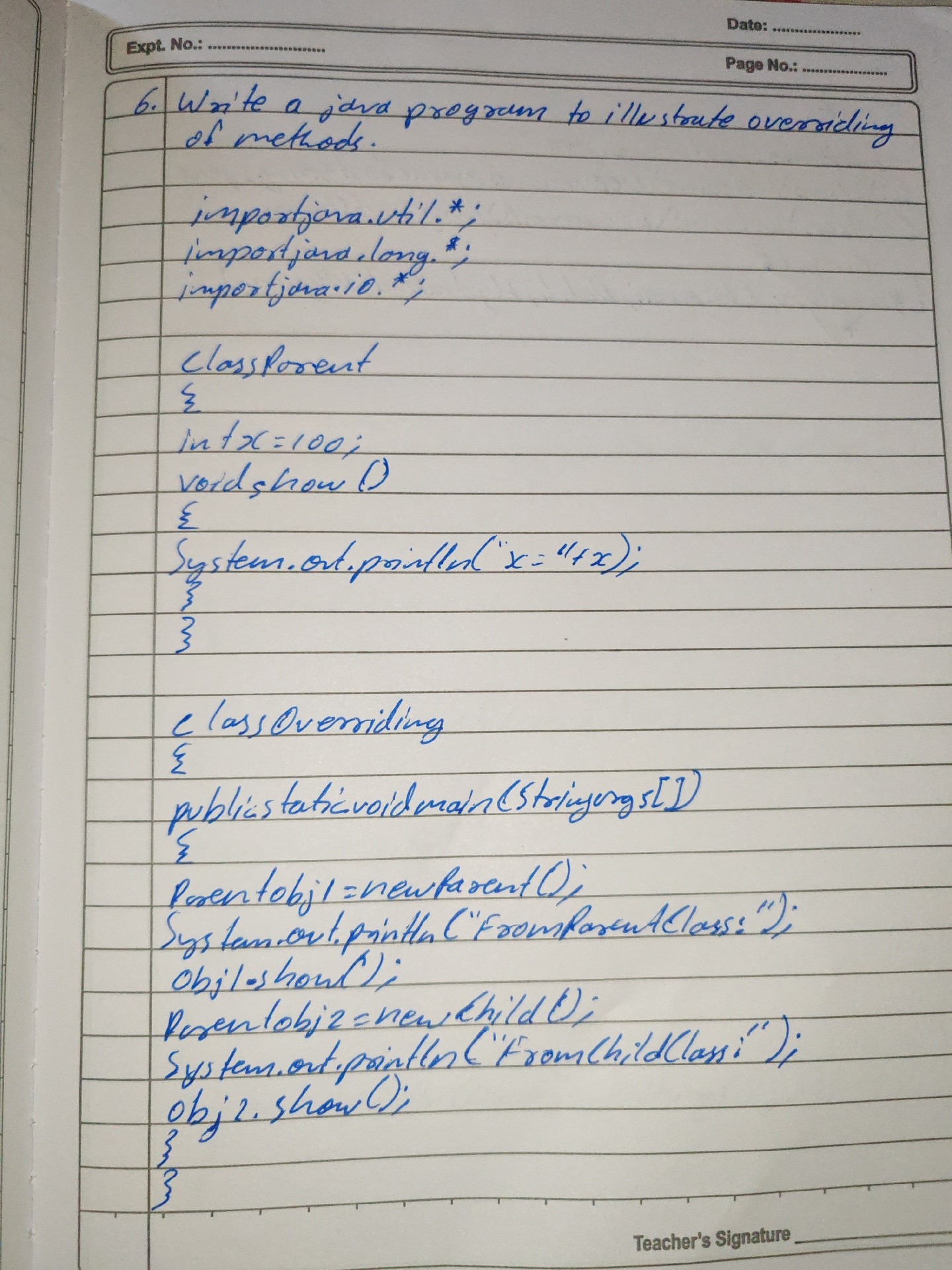
Hybrid Inheritance:Combination of multiple and multilevel inheritance.



Q3 Discuss the procedure for user defined package. Steps to Create a User-Defined Package:\*

1. Plan the package structure and choose a unique name.
2. Create a directory with the same name as the package.
3. Define the package in Java files using the `package` keyword.
4. Compile the Java files using `javac` with the `-d` option.
5. Use the package in other classes by importing it with the `import` state ment.

Q4 Write a program to demonstrate method overriding.



Q5 Explain the process of creating user-defined exceptions in Java. Provi de a practical example of a user-defined exception and how to handle it.

\*User-Defined Exceptions:\*

1. Extend the `Exception` class or its subclass.
2. Define a constructor with a message.
3. Use `throw` to throw the exception.
4. Handle it using `try-catch` blocks.

\*Example:\*

class MyException extends Exception { public MyException(String message) {

super(message);

}

}

public class Main {

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

throw new MyException("Custom error");

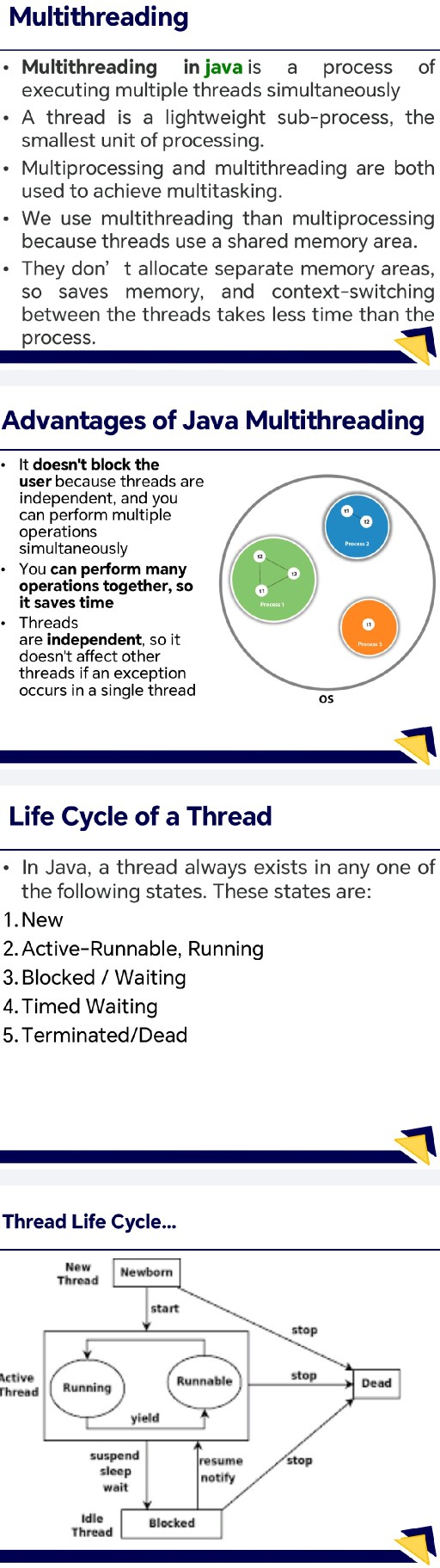
} catch (MyException e) { System.out.println(e.getMessage());

}

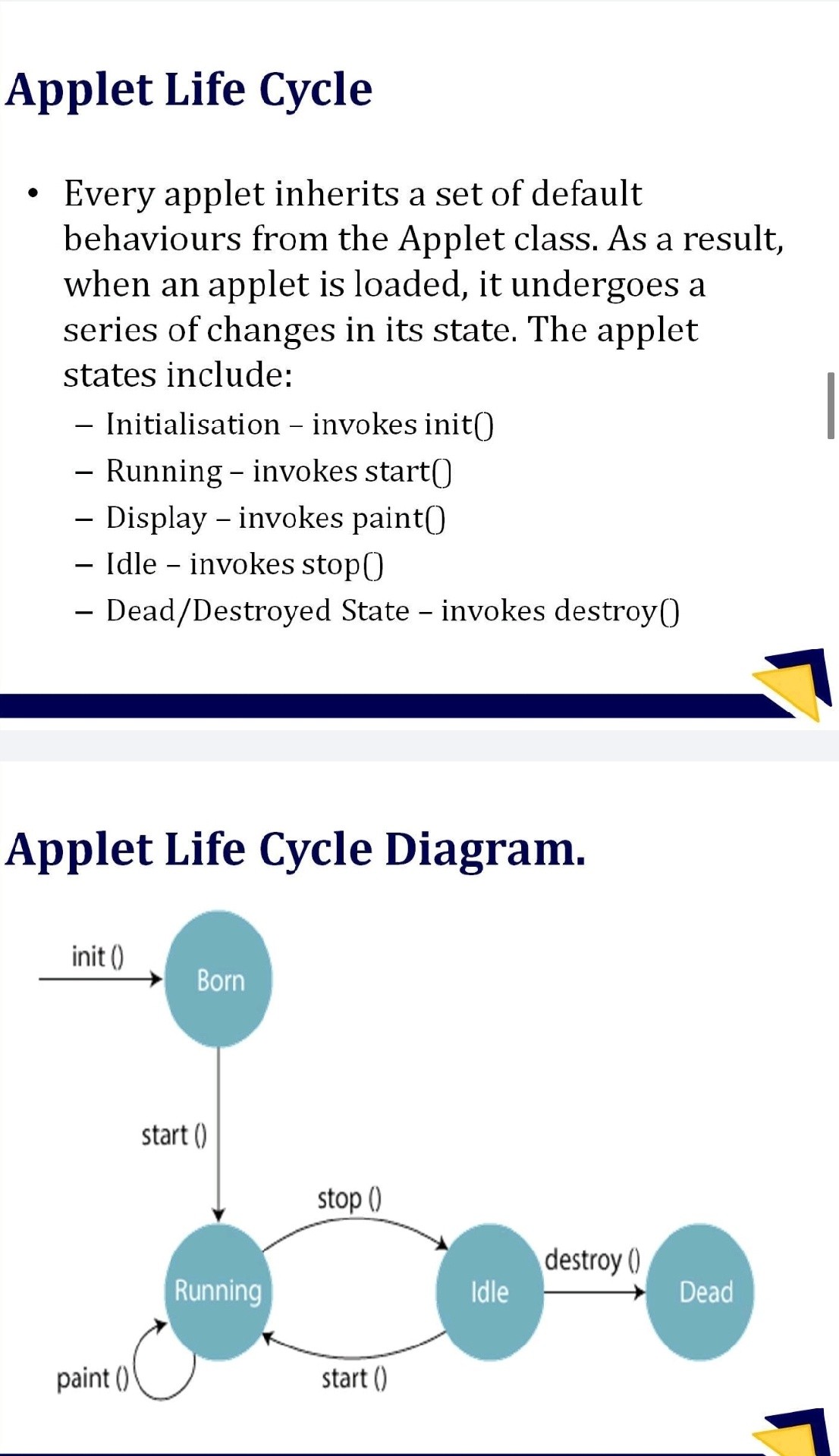
}

}

Q6 What is multithreading? Explain thread life cycle with a neat diagram.



Q7 Explain applet life cycle.



Q8 Give the classification of DataInputStream.

\*DataInputStream:\*

A Java class for reading primitive data types from an input stream in a m achine-independent way.

\*DataInputStream Classification:\*

`DataInputStream` is a subclass of `FilterInputStream` and implements th e `DataInput` interface.

\*Hierarchy:\*

* `java.lang.Object`

- `java.io.InputStream`

- `java.io.FilterInputStream`

- `java.io.DataInputStream`

\*Implements:\*

* `java.io.DataInput`