1. Compare Object Oriented and Procedure Oriented Programming paradigms listing advantages and disadvantages of each.

Object Oriented Programming (OOP):

- OOP is a programming paradigm based on the concept of objects that contain data (attributes) and methods (functions).
- It is mainly used for designing complex and large-scale applications.
- Example: Java, C++, Python.

Features of OOP:

1. Encapsulation: Bundling data and methods in a single unit (class).
2. Inheritance: Reusing the properties of an existing class.
3. Polymorphism: One function behaving differently based on input.
4. Abstraction: Hiding internal details and exposing only functionality.

Advantages of OOP:

1. Reusability: Code can be reused using inheritance.
2. Security: Encapsulation hides data from unauthorized access.
3. Scalability: Easy to manage and extend the code.
4. Maintainability: Changes in code do not affect other parts.

Disadvantages of OOP:

1. Complexity: More complex to design than POP.
2. Memory Consumption: Requires more memory to run.
3. Learning Curve: Difficult for beginners.

———

Procedure Oriented Programming (POP):

- POP is a programming style based on functions and sequential flow of control.
- It focuses on functions rather than objects.
- Example: C, COBOL, FORTRAN.

Features of POP:

1. Top-down Approach: Starts from main function and breaks down into sub-functions.
2. Global Data: Data is available to all functions.
3. Sequential Execution: One function calls another.

Advantages of POP:

1. Simple and Easy: Easy to understand and implement.
2. Memory Efficient: Requires less memory.
3. Less Development Time: Faster development.

Disadvantages of POP:

1. No Security: Data is exposed globally.
2. No Code Reusability: Functions cannot be reused.
3. Difficult to Modify: Making changes affects the entire code.

2. Explain the features and benefits of Object Oriented Development.

Features of Object Oriented Development:

1. Encapsulation: It is the mechanism of wrapping the data and code together in a single unit (class).
2. Inheritance: The process of acquiring properties from one class to another class.
3. Polymorphism: The ability of a function to perform different operations based on input.
4. Abstraction: Hiding complex implementation and showing only the necessary features.
5. Modularity: Dividing a large program into small modules for easier development.

Benefits of Object Oriented Development:

1. Code Reusability: Reusing the code reduces development time and cost.
2. Security: Encapsulation ensures that sensitive data is hidden from unauthorized access.
3. Scalability: Easy to scale the application when needed.
4. Maintainability: Easy to maintain and modify the code.

3. Explain Java Buzzwords or Features of Java.

Java is a powerful programming language introduced by Sun Microsystems. It has several key features known as Java Buzzwords:

1. Simple: Java is easy to learn, write, and understand.
2. Object-Oriented: Java supports OOP concepts like classes, objects, inheritance, etc.
3. Platform-Independent: Java code can run on any operating system without modification.
4. Robust: Java has strong memory management and exception handling features.
5. Secure: Java provides a secure runtime environment using a security manager.
6. Multithreading: Java supports the execution of multiple tasks simultaneously.
7. High Performance: Java uses Just-In-Time (JIT) compilers for faster execution.
8. Portable: Java applications can easily move from one platform to another.
9. Dynamic: Java supports dynamic loading of classes and objects during runtime.

4. Write a program to demonstrate Dynamic Method Dispatch and explain.

```
class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
    }
}

class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }
}

class Cat extends Animal {
    void sound() {
        System.out.println("Cat meows");
    }
}

public class Main {
    public static void main(String[] args) {
        Animal a;
        a = new Dog();
        a.sound();
        a = new Cat();
        a.sound();
    }
}
```

Explanation:

- Dynamic Method Dispatch (Runtime Polymorphism): It is the process of resolving a method call at runtime.
- The reference of the parent class (Animal) is used to hold the object of the child class (Dog or Cat).
- The method is determined at runtime based on the object type.

5. Explain the control statements in Java with suitable examples.

Control Statements:

1. Decision Making Statements:

- if statement: Executes a block of code if the condition is true.

```
if(a > b) {
    System.out.println("A is greater");
}
```

- switch statement: Executes one of many blocks based on a condition.

```
switch(choice) {
   case 1: System.out.println("One"); break;
   case 2: System.out.println("Two"); break;
}
```

2. Looping Statements:
    - for loop: Executes code repeatedly until condition is false.

```
for(int i=0; i<5; i++) {
   System.out.println(i);
}
```

while loop: Executes code as long as the condition is true.

```
while(x < 5) {
   System.out.println(x);
   x++;
}
```

6. What are Nested and Inner Classes in Java?

Nested Class:

- A class declared inside another class.
- Can be static or non-static.

Inner Class:

- A type of Nested Class that is non-static.
- Can access members of the outer class.

```
class Outer {
   class Inner {
      void msg() {
         System.out.println("Inner class method");
      }
   }
}
```

7. Write a program in Java using String class and its methods.

```
public class StringExample {
   public static void main(String[] args) {
      String str = "Hello World";
      System.out.println(str.toUpperCase());
      System.out.println(str.length());
      System.out.println(str.charAt(1));
   }
}
```

Methods Used:

1. toUpperCase() - Converts string to uppercase.

2. length() - Returns the length of the string.
3. charAt() - Returns a character at a specific index.

8. What are Wrapper Classes in Java?

- Wrapper classes are used to convert primitive data types into objects.
- Example:
- int → Integer
- double → Double
- char → Character

```
int a = 10;
Integer obj = a;
```

9. Write a program to read an integer and check whether it is prime or not.

```
import java.util.Scanner;
public class Prime {
   public static void main(String args[]) {
      Scanner sc = new Scanner(System.in);
      int num = sc.nextInt();
      boolean isPrime = true;
      for(int i=2; i<num/2; i++) {
         if(num % i == 0) {
            isPrime = false;
            break;
         }
      }
      if(isPrime) System.out.println("Prime");
      else System.out.println("Not Prime");
   }
}
```

10. Define a string. List and explain any 6 string handling methods with examples

What is a String in Java?

- In Java, a String is a sequence of characters.
- It is a predefined class in Java.lang package that represents a string of text.
- Strings in Java are immutable, meaning their value cannot be changed once created.
- A String can be created using double quotes (” “) or using the String class.

String Handling Methods with Examples:

1. length() Method

- Purpose: This method returns the length of the string (number of characters).

```
public class Example {
   public static void main(String[] args) {
```

```
        String str = "Hello World";
        System.out.println("Length of string: " + str.length());
    }
}
```

2. toUpperCase() Method

- Purpose: Converts all the characters of a string to uppercase letters.

Example:

```
public class Example {
    public static void main(String[] args) {
        String str = "hello world";
        System.out.println(str.toUpperCase());
    }
}
```

3. toLowerCase() Method

- Purpose: Converts all the characters of a string to lowercase letters.

Example:

```
public class Example {
    public static void main(String[] args) {
        String str = "HELLO WORLD";
        System.out.println(str.toLowerCase());
    }
}
```

4. charAt() Method

- Purpose: Returns the character at a specific index in the string.
- Indexing starts from 0 (zero).

Example:

```
public class Example {
    public static void main(String[] args) {
        String str = "Java";
        System.out.println("Character at index 1: " + str.charAt(1));
    }
}
```

5. substring() Method

- Purpose: Extracts a part (substring) from the string based on the starting index and optional ending index.

Example:

```
public class Example {
    public static void main(String[] args) {
        String str = "Hello World";
        System.out.println(str.substring(0, 5));
    }
}
```

6. replace() Method

- Purpose: Replaces all occurrences of a character or substring with another character or substring.

Example:

```
public class Example {
   public static void main(String[] args) {
      String str = "Java is Fun";
      System.out.println(str.replace("Java", "Python"));
   }
}
```

SAQ

1. What is the difference between == operator and equals() method of Object class?

- == Operator: It is used to compare the memory location (reference) of two objects, not their content.
- equals() Method: It is used to compare the content (data) of two objects.
- Example

```
String str1 = new String("Hello");
String str2 = new String("Hello");
System.out.println(str1 == str2);    // Output: false (Different memory)
System.out.println(str1.equals(str2)); // Output: true (Same content)
```

2. What are THIS, Static Variables, and Instance Variables?

- THIS: Refers to the current object of a class. It is used to resolve variable shadowing or pass the current object.
- Static Variable: Belongs to the class and shared among all objects. It is declared using the static keyword.
- Instance Variable: Belongs to the object and has a unique value for each object.

3. What is the purpose of Finalize() method or Explain Java Garbage Collection?

- Finalize() Method: It is a predefined method in Java called by the garbage collector before destroying an object.
- Purpose: To clean up resources like closing files, database connections, etc., before the object is destroyed.

- Garbage Collection: It is a process in Java that automatically deallocates memory occupied by unused objects.

4. Explain the significance of public static void main(String args[]).

- public: Makes the method accessible from anywhere.
- static: Allows the JVM to call the method without creating an object.
- void: The main method does not return any value.
- main(): Entry point of the Java program.
- String args[]: Accepts command-line arguments.

5. Define Class, Object, Constructor.

- Class: It is a blueprint or template that defines properties (variables) and behavior (methods).
- Object: An instance of a class that can access its members.
- Constructor: A special method that initializes an object. It has the same name as the class and has no return type.

```
class Car {
    String color;
    Car() {
        color = "Red";
    }
}
```

6. What are the uses of Final Keyword in Java?

- Final Class: Cannot be inherited.
- Final Method: Cannot be overridden.

- Final Variable: Value cannot be changed.

```
final int a = 10;

a = 20;  // Error: Cannot change value of final variable.
```

7. Differentiate between Constructor and Method.

Feature	Constructor	Method
Purpose	Initializes an object.	Performs operations.
Name	Same as class name.	Any valid name.
Return Type	No return type.	Must have a return type.
Call	Automatically when object is created.	Manually using object.

8. Explain Bytecode and Java Virtual Machine (JVM).

- Bytecode: It is the intermediate code generated by the Java compiler (.class file).
- Java Virtual Machine (JVM): It is responsible for executing the bytecode on any platform, making Java platform-independent.

9. List all Java Operators.

1. Arithmetic Operators: +, -, *, /, %
2. Relational Operators: ==, !=, <, >, <=, >=
3. Logical Operators: &&, ||, !
4. Assignment Operators: =, +=, -=, *=
5. Bitwise Operators: &, |, ^, ~