**Concepts of OOP**

1. Write a program to demonstrate encapsulation in Java.
2. Create a program showing the use of inheritance and polymorphism.
3. Explain and implement the concept of abstraction in Java using interfaces.
4. Write a program to demonstrate method overloading and method overriding.
5. Create a class hierarchy for animals that demonstrates polymorphism.
6. Develop a program to implement multiple inheritance using interfaces.
7. Write a Java program to showcase the use of this and super keywords.
8. Demonstrate the concept of constructors in OOP with a program.
9. Explain and implement the concept of access modifiers in Java.
10. Show an example of the final keyword for variables, methods, and classes.
11. Write a program that uses Java's StringBuilder for efficient string operations.
12. Write a program to demonstrate the immutability of the String class.

**Data Types, Control Statements, and Identifiers**

1. Write a program to declare variables of all primitive data types in Java and print their default values.
2. Implement a program to demonstrate the use of if-else, switch, and for loops.
3. Write a program to check if a number is prime using a while loop.
4. Create a program to calculate the factorial of a number using recursion.
5. Write a program to identify valid and invalid identifiers in Java.

**Arrays**

1. Write a program to find the largest and smallest numbers in an array.
2. Write a program to check if a given number is odd or even.
3. Write a program to find the largest of three numbers entered by the user.
4. Write a program to calculate the factorial of a given number using recursion.
5. Write a program to check if a given string or number is a palindrome.
6. Write a program to generate the first n terms of the Fibonacci series.
7. Write a program to check whether a given number is prime.
8. Write a program to find the sum of all elements in an array.
9. Implement a program to reverse the elements of an array.
10. Write a Java program to perform matrix addition and multiplication.
11. Create a program to sort an array using the bubble sort algorithm.
12. Write a program to demonstrate a 2D array and print its elements.
13. Write a program to search for an element in a sorted array using the binary search algorithm.
14. Write a program to remove duplicate elements from an array.

**Operators**

1. Write a program to demonstrate the use of arithmetic, relational, and logical operators.
2. Create a program to show the difference between == and equals() for string comparison.
3. Write a program to illustrate the use of the ternary operator.
4. Implement a program to perform bitwise operations in Java.
5. Write a program to demonstrate operator precedence in Java.

**Classes and Constructors**

1. Write a program to create a class with multiple constructors (constructor overloading).
2. Implement a program to demonstrate the use of a copy constructor in Java.
3. Create a program that initializes class fields using a parameterized constructor.
4. Write a program to demonstrate the use of static and non-static methods.
5. Implement a singleton class in Java.

**Inheritance and Method Overriding**

1. Write a program to demonstrate multilevel inheritance in Java.
2. Create a program to show method overriding and the use of super to call the parent class method.
3. Implement an abstract class and override its methods in a subclass.
4. Write a program to demonstrate final classes and methods.
5. Create a program to show run-time polymorphism using dynamic method dispatch.

**String Class and Operations**

1. Write a program to reverse a string without using built-in methods.
2. Implement a program to count the frequency of characters in a string.
3. Write a program to demonstrate the immutability of the String class.
4. Create a program to check if a given string is a palindrome.
5. Implement a program to split a string into words and print each word on a new line.