1. Hello World Program

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

2. Simple Calculator

```
import java.util.Scanner;


public class Calculator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter first number: ");
        double a = sc.nextDouble();
        System.out.print("Enter second number: ");
        double b = sc.nextDouble();
        System.out.print("Enter operation (+, -, *, /): ");
        char op = sc.next().charAt(0);
        switch(op) {
            case '+': System.out.println("Result: " + (a + b)); break;
            case '-': System.out.println("Result: " + (a - b)); break;
            case '*': System.out.println("Result: " + (a * b)); break;
            case '/': System.out.println("Result: " + (a / b)); break;
            default: System.out.println("Invalid operator");
        }
        sc.close();
```

```
    }
}
```

3. Even or Odd Checker

```
import java.util.Scanner;


public class EvenOdd {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int num = sc.nextInt();
        if (num % 2 == 0)
            System.out.println("Even");
        else
            System.out.println("Odd");
        sc.close();
    }
}
```

4. Leap Year Checker

```
import java.util.Scanner;


public class LeapYear {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a year: ");
        int year = sc.nextInt();
        if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0))
```

```
            System.out.println("Leap Year");
        else
            System.out.println("Not a Leap Year");
        sc.close();
    }
}
```

5. Multiplication Table

```
import java.util.Scanner;


public class MultiplicationTable {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        for (int i = 1; i <= 10; i++) {
            System.out.println(num + " * " + i + " = " + (num * i));
        }
        sc.close();
    }
}
```

6. Data Type Demonstration

```
public class DataTypesDemo {
    public static void main(String[] args) {
        int i = 42;
        float f = 3.14f;
        double d = 123.456;
```

```
        char c = 'X';
        boolean b = true;
        System.out.println("int: " + i);
        System.out.println("float: " + f);
        System.out.println("double: " + d);
        System.out.println("char: " + c);
        System.out.println("boolean: " + b);
    }
}
```

7. Type Casting Example

```
public class TypeCasting {
    public static void main(String[] args) {
        double d = 9.78;
        int i = (int) d;
        int j = 10;
        double dj = j;
        System.out.println("Double to Int: " + i);
        System.out.println("Int to Double: " + dj);
    }
}
```

8. Operator Precedence

```
public class OperatorPrecedence {
    public static void main(String[] args) {
        int result = 10 + 5 * 2;
        System.out.println("Result: " + result); // Output: 20
    }
```

```
}
```

9. Grade Calculator

```
import java.util.Scanner;


public class GradeCalculator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter marks (0-100): ");
        int marks = sc.nextInt();
        if (marks >= 90) System.out.println("Grade A");
        else if (marks >= 80) System.out.println("Grade B");
        else if (marks >= 70) System.out.println("Grade C");
        else if (marks >= 60) System.out.println("Grade D");
        else System.out.println("Grade F");
        sc.close();
    }
}
```

10. Number Guessing Game

```
import java.util.Scanner;


public class NumberGuessingGame {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int number = (int)(Math.random() * 100 + 1);
        int guess;
        do {
```

```
            System.out.print("Guess the number (1-100): ");
            guess = sc.nextInt();
            if (guess > number) System.out.println("Too high");
            else if (guess < number) System.out.println("Too low");
            else System.out.println("Correct!");
        } while (guess != number);
        sc.close();
    }
}
```

11. Factorial Calculator

```
import java.util.Scanner;

public class FactorialCalculator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a non-negative integer: ");
        int n = sc.nextInt();
        long factorial = 1;
        for (int i = 1; i <= n; i++) {
            factorial *= i;
        }
        System.out.println("Factorial: " + factorial);
        sc.close();
    }
}
```

12. Method Overloading

```
public class MethodOverloading {
    static int add(int a, int b) {
        return a + b;
    }

    static double add(double a, double b) {
        return a + b;
    }

    static int add(int a, int b, int c) {
        return a + b + c;
    }

    public static void main(String[] args) {
        System.out.println(add(2, 3));
        System.out.println(add(2.5, 3.5));
        System.out.println(add(1, 2, 3));
    }
}
```

13. Recursive Fibonacci

```
import java.util.Scanner;

public class RecursiveFibonacci {
    public static int fibonacci(int n) {
        if (n <= 1)
            return n;
```

```
        return fibonacci(n - 1) + fibonacci(n - 2);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter n: ");
        int n = sc.nextInt();
        System.out.println("Fibonacci number: " + fibonacci(n));
        sc.close();
    }
}
```

14. Array Sum and Average

```
import java.util.Scanner;

public class ArraySumAverage {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        int sum = 0;
        for (int i = 0; i < n; i++) {
            System.out.print("Enter element " + (i + 1) + ": ");
            arr[i] = sc.nextInt();
            sum += arr[i];
        }
```

```
        double avg = (double) sum / n;
        System.out.println("Sum: " + sum);
        System.out.println("Average: " + avg);
        sc.close();
    }
}
```

15. String Reversal

```
import java.util.Scanner;

public class StringReversal {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String input = sc.nextLine();
        String reversed = new StringBuilder(input).reverse().toString();
        System.out.println("Reversed: " + reversed);
        sc.close();
    }
}
```

16. Palindrome Checker

```
import java.util.Scanner;

public class PalindromeChecker {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
```

```
        String input = sc.nextLine().replaceAll("[^a-zA-Z0-9]", "").toLowerCase();
        String reversed = new StringBuilder(input).reverse().toString();
        if (input.equals(reversed))
            System.out.println("Palindrome");
        else
            System.out.println("Not a palindrome");
        sc.close();
    }
}
```

17. Class and Object Creation

```
public class Car {
    String make, model;
    int year;

    Car(String make, String model, int year) {
        this.make = make;
        this.model = model;
        this.year = year;
    }

    void displayDetails() {
        System.out.println("Make: " + make + ", Model: " + model + ", Year: " + year);
    }

    public static void main(String[] args) {
        Car car1 = new Car("Toyota", "Camry", 2020);
```

```
        car1.displayDetails();
    }
}
```

18. Inheritance Example

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

class Dog extends Animal {
    void makeSound() {
        System.out.println("Bark");
    }

    public static void main(String[] args) {
        Animal a = new Animal();
        Dog d = new Dog();
        a.makeSound();
        d.makeSound();
    }
}
```

19. Interface Implementation

```
interface Playable {
    void play();
}
```

```
class Guitar implements Playable {
    public void play() {
        System.out.println("Playing guitar");
    }
}

class Piano implements Playable {
    public void play() {
        System.out.println("Playing piano");
    }

    public static void main(String[] args) {
        Playable g = new Guitar();
        Playable p = new Piano();
        g.play();
        p.play();
    }
}
```

20. Try-Catch Example

```
import java.util.Scanner;

public class TryCatchExample {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        try {
```

```
            System.out.print("Enter numerator: ");
            int a = sc.nextInt();
            System.out.print("Enter denominator: ");
            int b = sc.nextInt();
            int result = a / b;
            System.out.println("Result: " + result);
        } catch (ArithmeticException e) {
            System.out.println("Cannot divide by zero.");
        } finally {
            sc.close();
        }
    }
}
```

21. Custom Exception

```
class InvalidAgeException extends Exception {
    public InvalidAgeException(String message) {
        super(message);
    }
}


public class CustomException {
    public static void main(String[] args) {
        int age = 16;
        try {
            if (age < 18)
                throw new InvalidAgeException("Age must be 18 or older.");
```

```
            System.out.println("Valid age.");
        } catch (InvalidAgeException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

22. File Writing

```
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class FileWriting {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter text to write to file: ");
        String text = sc.nextLine();

        try (FileWriter writer = new FileWriter("output.txt")) {
            writer.write(text);
            System.out.println("Data written to output.txt");
        } catch (IOException e) {
            e.printStackTrace();
        }

        sc.close();
    }
```

```
}
```

23. File Reading

```
import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;


public class FileReading {

    public static void main(String[] args) {

        try (BufferedReader reader = new BufferedReader(new FileReader("output.txt"))) {

            String line;

            while ((line = reader.readLine()) != null)

                System.out.println(line);

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}
```

24. ArrayList Example

```
import java.util.ArrayList;

import java.util.Scanner;


public class ArrayListExample {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        ArrayList<String> names = new ArrayList<>();

        System.out.println("Enter names (type 'end' to finish):");
```

```
        while (true) {
            String name = sc.nextLine();
            if (name.equalsIgnoreCase("end")) break;
            names.add(name);
        }

        System.out.println("Student Names:");
        for (String name : names)
            System.out.println(name);

        sc.close();
    }
}
```

25. HashMap Example

```
import java.util.HashMap;
import java.util.Scanner;

public class HashMapExample {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        HashMap<Integer, String> students = new HashMap<>();
        System.out.println("Enter ID and name (type -1 to stop):");
        while (true) {
            int id = sc.nextInt();
            if (id == -1) break;
            sc.nextLine();
```

```
            String name = sc.nextLine();
            students.put(id, name);
        }

        System.out.print("Enter ID to search: ");
        int searchId = sc.nextInt();
        System.out.println("Name: " + students.getOrDefault(searchId, "Not found"));

        sc.close();
    }
}
```

26. Thread Creation

```
class MyThread extends Thread {
    public void run() {
        for (int i = 0; i < 5; i++)
            System.out.println("Thread running: " + i);
    }

    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        Thread t2 = new Thread(() -> {
            for (int i = 0; i < 5; i++)
                System.out.println("Runnable thread: " + i);
        });

        t1.start();
```

```
        t2.start();
    }
}
```

27. Lambda Expressions

```
import java.util.*;


public class LambdaSort {
    public static void main(String[] args) {
        List<String> list = Arrays.asList("Banana", "Apple", "Orange");
        list.sort((a, b) -> a.compareToIgnoreCase(b));
        list.forEach(System.out::println);
    }
}
```

28. Stream API

```
import java.util.*;
import java.util.stream.Collectors;


public class StreamEvenNumbers {
    public static void main(String[] args) {
        List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5, 6);
        List<Integer> evens = numbers.stream()
            .filter(n -> n % 2 == 0)
            .collect(Collectors.toList());
        System.out.println("Even numbers: " + evens);
    }
}
```

29. Records

```
import java.util.List;


public class RecordExample {
    record Person(String name, int age) {}


    public static void main(String[] args) {
        List<Person> people = List.of(new Person("Alice", 30), new Person("Bob", 20));
        people.stream().filter(p -> p.age() >= 25).forEach(System.out::println);
    }
}
```

30. Pattern Matching for switch

```
public class PatternMatching {
    public static void printType(Object obj) {
        switch (obj) {
            case Integer i -> System.out.println("Integer: " + i);
            case String s -> System.out.println("String: " + s);
            case Double d -> System.out.println("Double: " + d);
            default -> System.out.println("Unknown type");
        }
    }


    public static void main(String[] args) {
        printType(10);
        printType("Hello");
        printType(5.5);
```

```
    }
}
```

31. Basic JDBC Connection

```
import java.sql.*;


public class JDBCBasic {
    public static void main(String[] args) {
        try (Connection conn = DriverManager.getConnection("jdbc:sqlite:students.db");
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery("SELECT * FROM students")) {
            while (rs.next()) {
                        System.out.println("ID: " + rs.getInt("id") + ", Name: " +
rs.getString("name"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

32. Insert and Update Operations in JDBC

```
import java.sql.*;


public class StudentDAO {
    public void insertStudent(int id, String name) throws SQLException {
        try (Connection conn = DriverManager.getConnection("jdbc:sqlite:students.db")) {
            String sql = "INSERT INTO students(id, name) VALUES (?, ?)";
```

```
            try (PreparedStatement stmt = conn.prepareStatement(sql)) {
                stmt.setInt(1, id);
                stmt.setString(2, name);
                stmt.executeUpdate();
            }
        }
    }

    public void updateStudent(int id, String name) throws SQLException {
        try (Connection conn = DriverManager.getConnection("jdbc:sqlite:students.db")) {
            String sql = "UPDATE students SET name = ? WHERE id = ?";
            try (PreparedStatement stmt = conn.prepareStatement(sql)) {
                stmt.setString(1, name);
                stmt.setInt(2, id);
                stmt.executeUpdate();
            }
        }
    }
}
```

33. Transaction Handling in JDBC

```
import java.sql.*;

public class TransactionExample {
       public static void transferMoney(int fromId, int toId, double amount) throws
SQLException {
        try (Connection conn = DriverManager.getConnection("jdbc:sqlite:bank.db")) {
```

```
            conn.setAutoCommit(false);
            try {
                    PreparedStatement debit = conn.prepareStatement("UPDATE accounts SET
balance = balance - ? WHERE id = ?");
                debit.setDouble(1, amount);
                debit.setInt(2, fromId);
                debit.executeUpdate();

                   PreparedStatement credit = conn.prepareStatement("UPDATE accounts SET
balance = balance + ? WHERE id = ?");
                credit.setDouble(1, amount);
                credit.setInt(2, toId);
                credit.executeUpdate();

                conn.commit();
                System.out.println("Transfer successful.");
            } catch (SQLException e) {
                conn.rollback();
                System.out.println("Transfer failed. Rolled back.");
            }
        }
    }
}
```

34. Create and Use Java Modules

```
// com/utils/Utils.java
package com.utils;
```

```
public class Utils {
    public static String getMessage() {
        return "Hello from Utils";
    }
}

// com/greetings/Main.java
package com.greetings;

import com.utils.Utils;

public class Main {
    public static void main(String[] args) {
        System.out.println(Utils.getMessage());
    }
}
```

35. TCP Client-Server Chat

```
import java.io.*;
import java.net.*;

public class TCPServer {
    public static void main(String[] args) throws IOException {
        ServerSocket server = new ServerSocket(5000);
        Socket client = server.accept();
                                BufferedReader in = new BufferedReader(new
```

```
InputStreamReader(client.getInputStream()));
        PrintWriter out = new PrintWriter(client.getOutputStream(), true);

        String line;
        while ((line = in.readLine()) != null) {
            System.out.println("Client: " + line);
            out.println("Echo: " + line);
        }

        client.close();
        server.close();
    }
}
```

36. HTTP Client API

```
import java.net.URI;
import java.net.http.*;
import java.io.IOException;

public class HttpClientExample {
    public static void main(String[] args) throws IOException, InterruptedException {
        HttpClient client = HttpClient.newHttpClient();
        HttpRequest request = HttpRequest.newBuilder()
            .uri(URI.create("https://api.github.com"))
            .build();

                            HttpResponse<String>    response    =    client.send(request,
```

```
HttpResponse.BodyHandlers.ofString());

        System.out.println("Status: " + response.statusCode());

        System.out.println("Body: " + response.body());

    }

}
```

37. Using javap to Inspect Bytecode

```
// Compile with: javac Test.java

// Inspect with: javap -c Test

public class Test {

    public static void hello() {

        System.out.println("Hello bytecode!");

    }

}
```

38. Decompile a Class File

```
// Write and compile Test.java

// Open Test.class in JD-GUI or CFR to view decompiled source

public class Test {

    public static void main(String[] args) {

        System.out.println("Decompile me!");

    }

}
```

39. Reflection in Java

```
import java.lang.reflect.Method;


public class ReflectionExample {
```

```
    public static void main(String[] args) throws Exception {
        Class<?> clazz = Class.forName("java.util.ArrayList");
        Method[] methods = clazz.getDeclaredMethods();
        for (Method m : methods) {
            System.out.println(m.getName());
        }
    }
}
```

40. Virtual Threads

```
public class VirtualThreads {
    public static void main(String[] args) {
        for (int i = 0; i < 100000; i++) {
                Thread.startVirtualThread(() -> System.out.println("Hello from virtual
thread"));
        }
    }
}
```

41. Executor Service and Callable

```
import java.util.concurrent.*;

public class ExecutorCallableExample {
    public static void main(String[] args) throws Exception {
        ExecutorService executor = Executors.newFixedThreadPool(3);
        Callable<String> task = () -> "Result from thread";

        Future<String> result = executor.submit(task);
```

```
        System.out.println(result.get());



        executor.shutdown();

    }

}
```