Problem 1

Write a program to input marks of students in 5 subjects, calculate total, average, and grade using methods and handle invalid marks using exception handling.

```
import java.util.*;

class GradeCalculator {
    static int total = 0;

    static void inputMarks() throws Exception {
        Scanner sc = new Scanner(System.in);
        for (int i = 1; i <= 5; i++) {
            System.out.print("Enter mark " + i + ": ");
            int mark = sc.nextInt();
            if (mark < 0 || mark > 100)
                throw new Exception("Marks must be between 0 and 100");
            total += mark;
        }
    }

    static String grade(double avg) {
        if (avg >= 90) return "A";
        else if (avg >= 80) return "B";
        else if (avg >= 70) return "C";
        else if (avg >= 60) return "D";
        else return "F";
    }

    public static void main(String[] args) {
        try {
            inputMarks();
            double avg = total / 5.0;
            System.out.println("Total: " + total);
            System.out.println("Average: " + avg);
            System.out.println("Grade: " + grade(avg));
        } catch (Exception e) {
            System.out.println("Invalid input: " + e.getMessage());
        }
    }
}
```

Problem 2

Accept item names, price, and quantity. Calculate total, apply a discount if total ¿ 2000, and display formatted bill using methods.

```
import java.util.*;

public class Bill {
    static double discount(double total) {
        return (total > 2000) ? total * 0.1 : 0;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of items: ");
        int n = sc.nextInt();
        String[] name = new String[n];
        double[] price = new double[n];
        int[] qty = new int[n];
        double total = 0;

        for (int i = 0; i < n; i++) {
            System.out.print("Enter item name, price, quantity: ");
            name[i] = sc.next();
            price[i] = sc.nextDouble();
            qty[i] = sc.nextInt();
            total += price[i] * qty[i];
        }
        double dis = discount(total);
        System.out.println("Total: " + total);
        System.out.println("Discount: " + dis);
        System.out.println("Net Amount: " + (total - dis));
    }
}
```

Problem 3

Take a sentence and count the number of words and occurrences of a specific word using arrays and string methods.

```
import java.util.*;

public class WordCount {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter a sentence:");
        String sentence = sc.nextLine();
        System.out.println("Enter a word to count:");
        String word = sc.next();

        String[] words = sentence.split("\\s+");
        int count = 0;
        for (String w : words)
            if (w.equalsIgnoreCase(word))
                count++;

        System.out.println("Total words: " + words.length);
        System.out.println("Occurrences of ’" + word + "’: " + count);
    }
}
```

Problem 4

Check password strength: Length 8, contains uppercase, lowercase, digit, and symbol, throw exception if invalid.

```
import java.util.*;

public class PasswordCheck {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter password: ");
        String p = sc.next();

        try {
            if (p.length() < 8) throw new Exception("Password too short");
            if (!p.matches(".*[A-Z].*")) throw new Exception("No uppercase letter");
            if (!p.matches(".*[a-z].*")) throw new Exception("No lowercase letter");
            if (!p.matches(".*\\d.*")) throw new Exception("No digit");
            if (!p.matches(".*[^a-zA-Z0-9].*")) throw new Exception("No special symbo
            System.out.println("Password is strong!");
        } catch (Exception e) {
            System.out.println("Invalid password: " + e.getMessage());
        }
    }
}
```

Problem 5

Simulate ATM operations like deposit, withdraw, and check balance. Use methods for each operation and handle insufficient balance with exception handling.

```
import java.util.*;

public class ATM {
    static double balance = 0;

    static void deposit(double amt) {
        balance += amt;
        System.out.println("Deposited: " + amt);
    }

    static void withdraw(double amt) throws Exception {
        if (amt > balance)
```

```
            throw new Exception("Insufficient balance");
        balance -= amt;
        System.out.println("Withdrawn: " + amt);
    }

    static void checkBalance() {
        System.out.println("Balance: " + balance);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (true) {
            System.out.println("1. Deposit 2. Withdraw 3. Check Balance 4. Exit");
            int ch = sc.nextInt();

            try {
                switch (ch) {
                    case 1 -> deposit(sc.nextDouble());
                    case 2 -> withdraw(sc.nextDouble());
                    case 3 -> checkBalance();
                    case 4 -> System.exit(0);
                    default -> System.out.println("Invalid choice!");
                }
            } catch (Exception e) {
                System.out.println(e.getMessage());
            }
        }
    }
}
```

Problem 6

Accept basic salary and compute HRA, DA, PF, and gross salary. Display results using methods and handle invalid inputs with exceptions.

```
import java.util.*;

public class Salary {
    static double hra(double s) { return s * 0.1; }
    static double da(double s) { return s * 0.08; }
    static double pf(double s) { return s * 0.05; }
    static double gross(double s) { return s + hra(s) + da(s) - pf(s); }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter basic salary: ");
        try {
            double sal = sc.nextDouble();
            if (sal < 0) throw new Exception("Invalid salary");
            System.out.println("HRA: " + hra(sal));
            System.out.println("DA: " + da(sal));
            System.out.println("PF: " + pf(sal));
            System.out.println("Gross Salary: " + gross(sal));
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Problem 7

Accept total bill and membership type (Silver/Gold/Platinum) and apply discounts accordingly using if-else and methods.

```
import java.util.*;

public class Discount {
    static double discount(double bill, String type) {
        if (type.equalsIgnoreCase("Silver")) return bill * 0.05;
        else if (type.equalsIgnoreCase("Gold")) return bill * 0.10;
        else if (type.equalsIgnoreCase("Platinum")) return bill * 0.15;
        else return 0;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter bill amount: ");
        double bill = sc.nextDouble();
        sc.nextLine();
        System.out.print("Enter membership type: ");
        String type = sc.nextLine();

        double d = discount(bill, type);
        System.out.println("Discount: " + d);
        System.out.println("Net Amount: " + (bill - d));
    }
}
```

Problem 8

For ‘n’ products, store product name, price, and quantity in arrays. Calculate total stock value and handle out-of-stock errors via exception handling.

```
import java.util.*;

public class Stock {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of items: ");
        int n = sc.nextInt();

        String[] name = new String[n];
        double[] price = new double[n];
        int[] qty = new int[n];
        double total = 0;

        for (int i = 0; i < n; i++) {
            System.out.print("Enter item name, price, quantity: ");
            name[i] = sc.next();
            price[i] = sc.nextDouble();
            qty[i] = sc.nextInt();
        }

        for (int i = 0; i < n; i++) {
            try {
                if (qty[i] == 0)
                    throw new Exception(name[i] + " is out of stock");
                total += price[i] * qty[i];
            } catch (Exception e) {
                System.out.println("Error: " + e.getMessage());
            }
        }

        System.out.println("Total stock value: " + total);
    }
}
```