**Experiment No.06**

Name : Sanskruti Shinde. (139)

**Aim :** Analysis of multilevel memory hierarchy.

**Code :**

import java.util.Scanner;

import java.text.DecimalFormat;

public class MemoryHierarchyAnalysis {

    public static double averageCostPerBit(double[] cost, double[] size) {

        double num = 0.0;

        double den = 0.0;

        for (int i = 0; i < cost.length; i++) {

            num += cost[i] \* size[i];

            den += size[i];

        }

        if (den == 0) return 0.0;

        return num / den;

    }

    public static double averageAccessTime(double[] hitRate, double[] accessTime) {

        double missProb = 1.0;

        double tav = 0.0;

        for (int i = 0; i < hitRate.length; i++) {

            double serveProb = missProb \* hitRate[i];

            tav += serveProb \* accessTime[i];

            missProb = missProb \* (1.0 - hitRate[i]);

        }

        return tav;

    }

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        DecimalFormat dfCost = new DecimalFormat("#.######");

        DecimalFormat dfTime = new DecimalFormat("#.####");

        System.out.println("Analysis of Multilevel Memory Hierarchy (n = 2 or 3)");

        int n;

        while (true) {

            System.out.print("Enter number of levels n (2 or 3): ");

            n = sc.nextInt();

            if (n >= 2 && n <= 3) break;

            System.out.println("Invalid n. Please enter 2 or 3.");

        }

        double[] cost = new double[n];

        double[] size = new double[n];

        double[] hitRate = new double[n];

        double[] accessTime = new double[n];

        for (int i = 0; i < n; i++) {

            int level = i + 1;

            System.out.printf("---- Level %d ----%n", level);

            System.out.printf("Enter cost/bit C%d (INR): ", level);

            cost[i] = sc.nextDouble();

            System.out.printf("Enter size S%d (bits): ", level);

            size[i] = sc.nextDouble();

            System.out.printf("Enter hit rate H%d (0 to 1) : ", level);

            hitRate[i] = sc.nextDouble();

            if (hitRate[i] < 0.0) hitRate[i] = 0.0;

            if (hitRate[i] > 1.0) hitRate[i] = 1.0;

            System.out.printf("Enter access time ta%d (microseconds): ", level);

            accessTime[i] = sc.nextDouble();

        }

        sc.close();

        double cav = averageCostPerBit(cost, size);

        double tav = averageAccessTime(hitRate, accessTime);

        System.out.println("\n===== Results =====");

        System.out.println("Average cost per bit (INR) = " + dfCost.format(cav));

        System.out.println("Average access time (microseconds) = " + dfTime.format(tav));

    }

}

**Output :**



