LAB 4 – Assignment

Name: Krish Singh

Roll: AID25072

Class: AID-G sem 1

PROBLEM 1 - Daily Sales Analysis (2D Array)

Pseudo code:

START

Declare a 2D array with 3 rows(products) and 4 columns(days)

FOR each product i from 0 to product-1 Do

SET sum=0

FOR each day j from 0 to day-1 Do

sum=sum+sales[i][j]

END FOR

PRINT"Product i total sales=sum"

END FOR

END

Explanation:

- each row represents one product
- each column represents sales for a day
- outer loop -> moves product wise
- inner loop -> accumulates daily sales for that product

Code:

```
class SalesAnalysis{
   int[][] sales={
         {10,12,15,11},
         {8,9,10,12},
         {14,13,16,15}
   };
   void compute(){
      for(int i=0;i<sales.length;i++){
```

```
        int sum=0;
        for(int j=0;j<sales[i].length;j++){
            sum+=sales[i][j];
        }
        System.out.println("Product "+i+" total sales: "+sum);
    }
  }

  public static void main(String[] args){

      new SalesAnalysis().compute();

  }
}
```

Output:

```
/usr/java/jdk-17-oracle-x64/bin/java -javaagent:/h
Product 0 total sales: 48
Product 1 total sales: 39
Product 2 total sales: 58

Process finished with exit code 0
```

PROBLEM 2 - Student Attendance Tracking (2D Array)

Pseudo code:

START

Declare a 2D array attendance with 3 rows(Students) and 5 columns(days)

(1=present, 0=absent)

FOR each student i from 0 to number of students-1 DO

SET sum=0

FOR each day j from 0 to number of days-1 DO

sum=sum+attendance[i][j]

END FOR

PRINT"Student i attendance=sum"

END FOR

END

Explanation:

- each row represents a student
- each column represents days that student was present or absent
- present is represented by 1 and absent by 0 on each day
- adding ones directly gives total presence count
- structure is row wise traversal same as matrix logic

Code:

```
class AttendanceAnalysis {
   int[][] attendance = {
         {1, 1, 0, 1, 1},
         {1, 0, 1, 1, 0},
         {1, 1, 1, 1, 1}
   };

   void compute() {
      for (int i = 0; i < attendance.length; i++) {
```

```
        int sum=0;
        for (int j = 0; j < attendance[i].length; j++) {
            sum+=attendance[i][j];
        }
        System.out.println("Student "+i+" attendence="+sum);
    }
  }
  public static void main(String[] args){
    new AttendanceAnalysis().compute();
  }
}
```

Output:

```
/usr/java/jdk-17-oracle-x64/bin/java -javaagent:/
Student 0 attendence=4
Student 1 attendence=3
Student 2 attendence=5

Process finished with exit code 0
```

PROBLEM 3 - Classroom Height Analysis (2D Array)

Pseudo code:

START

Declare a 2D array with 3 rows(Class) and 3 columns(heights)

SET max=height[0][0]

SET sum=0

SET count=0

FOR class i from 0 to number of classes-1 DO

FOR height j from 0 to number of heights-1 DO

sum=sum+heights[i][j]

count=count+1

IF heights[i][j] is greater than max

SET max=heights[i][j]

END IF

END FOR

END FOR

PRINT"Maximum height=max"

PRINT"Average height=sum/count"

END

Explanation:

- each row represents a class
- each column represents height of students in that class
- max height is initially set to the first height at index (0,0)
- outer loop moves class wise
- inner loop accumulates heights of student in that class, and keeps increasing the count.
- if condition is set up in the inner loop that checks if the height at current index is greater than the max height, if yes max height is set to the height at current index else the program continues without change
- average is given by sum of all heights divided by count (total number of heights)

Code:

```
class HeightAnalysis{
   int[][] heights={
         {150,152,148},
         {160,158,155},
         {165,170,168}
   };
   void compute(){
      int max=heights[0][0];
      int sum=0;
      int count=0;
      for(int i=0;i<heights.length;i++){
         for(int j=0;j<heights[i].length;j++){
            sum+=heights[i][j];
            count++;
            if(heights[i][j]>max){
               max=heights[i][j];
            }
         }
      }
      System.out.println("Maximum height="+max);
      System.out.println("Average height="+(double)sum/count);
   }
   public static void main(String[] args){
      new HeightAnalysis().compute();
   }
}
```

Output:

```
/usr/java/jdk-17-oracle-x64/bin/java -javaagent:/
Maximum height=170
Average height=158.444444444444446


Process finished with exit code 0
```

PROBLEM 4 - Temperature Monitoring (3D Array)

Pseudo code:

START

Declare a 3D array temperatures with 2 matrices(rooms) of 2 rows(days) and 2 columns(temperatures)

START

FOR each room i from 0 to number of rooms-1 DO

SET sum=0

SET count=0

FOR each day j from 0 to number of days-1 DO

FOR each temp k from 0 to number of temp-1 DO

sum=sum+temperature[i][j][k]

count=count+1

END FOR

END FOR

PRINT"Room i average = sum/count"

END FOR

END

Explanation:

- each matrix represents a room
- each row represents a day
- each column represents a temperature reading
- outer loop moves room wise
- sum and count are initially set to 0
- middle loop moves day wise
- inner loop accumulates temperature readings and increases the count by 1
- average room temperature is given by sum/count

Code:

```
class TemperatureMonitor{
   int[][][] temperature={
         { //this is for room 1
               {30,32}, // day 1 temp 1 and 2
               {33,34} // day 2 temp 1 and 2
         },
         { //this is for room 2
               {28,29}, // day 1 temp 1 and 2
               {30,31}, // day 2 temp 1 and 2
         }
   };
   void compute(){
      for(int room=0;room<temperature.length;room++){
         int sum=0;
         int count=0;
         for(int day=0;day<temperature[room].length;day++){
            for(int temp=0;temp<temperature[room][day].length;temp++){
               sum+=temperature[room][day][temp];
               count++;
            }
         }
         System.out.println("Room "+room+" average= "+(double)sum/count);
      }
   }
   public static void main(String[] args){
      new TemperatureMonitor().compute();
   }

}
```

Output:

```
/usr/java/jdk-17-oracle-x64/bin/java -javaagent:/h
Room 0 average= 32.25
Room 1 average= 29.5


Process finished with exit code 0
```

PROBLEM 5 - Student Marks Analysis (3D Array)

Pseudo code:

START

Declare a 3D array marks with 2 matrices(class) of 2 rows(student) and 2 columns(subject)

FOR class i from 0 to number of class-1 DO

FOR student j from 0 to number of students-1 DO

SET total=0

FOR subject k from 0 to number of subjects-1 DO

total=total+marks[i][j][k]

END FOR

PRINT"Class i student j total=total"

END FOR

END FOR

END

Explanation:

- each matrix represents a class
- each row represent a student
- each column represents a subject
- outer loop moves class wise
- middle loop moves student wise
- inner loop accumulates marks of the student

Code:

```
class MarksAnalysis{
  int[][][] marks={
        { //class 1
              {80,85}, //student 1 subject 1 and 2
              {70,75} // student 2 subject 1 and 2
        },
        { // class 2
              {90,95}, // student 1 subject 1 and 2
              {88,92} // student 2 subject 1 and 2
        }
  };
  void compute(){
```

```
    for(int clss=0;clss<marks.length;clss++){
        for(int student=0;student<marks[clss].length;student++){
            int total=0;
            for(int subject=0;subject<marks[clss][student].length;subject++){
                total+=marks[clss][student][subject];
            }
            System.out.println("Class "+clss+" Student "+student+" total= "+total);
        }
    }
  }
  public static void main(String[] args){
    new MarksAnalysis().compute();
  }
}
```

Output:

```
/usr/java/jdk-17-oracle-x64/bin/java -javaagent:/h
Class 0 Student 0 total= 165
Class 0 Student 1 total= 145
Class 1 Student 0 total= 185
Class 1 Student 1 total= 180

Process finished with exit code 0
```

PROBLEM 6 - Hospital Bed Occupancy Analysis (3D Array)

Pseudo code:

START

Declare a 3D array with 3 matrices(wards) of 3 rows(days) and 3 columns(Time Slots)

SET max=occupancy[0][0][0]

SET ovward=0, ovday=0, ovtime=0

FOR ward i from 0 to number of wards-1 DO

SET sum=0

SET count=0

FOR day j from 0 to number of days-1 DO

FOR timeSlot k from 0 to number of time slots-1 DO

sum=sum+occupancy[i][j][k]

count=count+1

IF occupancy[i][j][k] is greater than max

SET max=occupancy[i][j][k]

SET ovward=ward

SET ovday=day

SET ovtime=time

END IF

END FOR

END FOR

SET average=sum/count

IF average is greater than or equal to 45

SET condition=true

END IF

PRINT"Ward i overloaded = condition"

END FOR

SET stress=false

FOR day i from 0 to number of days-1 DO

SET sum=0

SET count=0

FOR ward j from 0 to number of wards-1 Do

```
FOR time slots k from 0 to number of time slots-1 DO
sum=sum+occupancy[i][j][k]
count=count+1
END FOR
END FOR
SET dayaverage=sum/count
IF dayaverage is greater than 40
SET stress=true
break the loop i
END IF
END FOR
PRINT"Hospital under stress=stress"
PRINT"Maximum occupancy=max ward=ovward day=ovday time=ovtime"
END
```

Explanation:

- each matrix represents a ward
- each row represents a day
- each column represents a time slot
- The first outer loop moves ward wise
- The first middle loop moves day wise
- The first inner loop accumulates sum of patients across time slots, increments count by 1 and checks if the number of patients is greater than max or not.
- if number of patients is greater than max then max is set to the number of patients at that index
- average is calculated by sum divide by count
- if average is greater than or equal to 45 then ward is printed as overloaded, else not.

-The second loop starts from days->ward->time slots to check average patients across all wards and time slots in a day.

- sum and count are re-initialized
- The inner loop accumulates sum and increments count
- average is calculated and if day average is greater than 40 hospital is marked a s under stress

Code:

```
class Hospital{
   int[][][] occupancy={
         {
               {46,50,45},
               {45,46,48},
               {45,48,44}
         },
         {
               {39,35,38},
               {40,29,40},
               {40,40,39}
         },
         {
               {55,44,45},
               {45,48,47},
               {45,46,49}
         }
   };
   void compute(){
      int max=occupancy[0][0][0];
      int ovward=0, ovday=0, ovtime=0;
      boolean stress=false;
      for(int ward=0;ward<occupancy.length;ward++){
         int sum=0;
         int count=0;
         for(int day=0;day<occupancy[ward].length;day++) {
            for (int time = 0; time < occupancy[ward][day].length; time++) {
               sum += occupancy[ward][day][time];
               count++;
               if (occupancy[ward][day][time] > max) {
                  max = occupancy[ward][day][time];
                  ovward = ward;
                  ovday = day;
                  ovtime = time;
               }
            }
         }

         double average=(double)sum/count;
         System.out.println("Ward "+ward+" overloaded= "+(average>=45));
      }
      for(int day=0;day<occupancy[0].length;day++){
         int sum=0;
         int count=0;
         for(int ward=0;ward<occupancy.length;ward++){
            for(int time=0;time<occupancy[ward][day].length;time++) {
               sum+=occupancy[ward][day][time];
               count++;
            }
         }
         double dayavg=(double)sum/count;
```

```
        if(dayavg>40) {
            stress = true;
            break;
        }

    }
    System.out.println("Hospital under stress= "+stress);
    System.out.println("Maximum Occupancy= "+max+" Ward "+ovward+" Day "+ovday+" Time
"+ovtime);
  }
  public static void main(String[] args){
    new Hospital().compute();
  }


}
```

Output:

```
/usr/java/jdk-17-oracle-x64/bin/java -javaagent:/hom
Ward 0 overloaded= true
Ward 1 overloaded= false
Ward 2 overloaded= true
Hospital under stress= true
Maximum Occupancy= 55 Ward 2 Day 0 Time 0


Process finished with exit code 0
```