

الاسم: إياد امين احمد احمد الفار

ID: 202000172

Sheet 2

**5.4**

(*Conversion from miles to kilometers*) Write a program that displays the following table (note **1** mile is **1.609** kilometers):

Sol:

public class aaa {

    public static void main(String args[]) {

        double km;

        int m=0;

        System.out.println("Miles\t\tKilometers");

        for (m=1; m<=10;m++){

        km = m\*(1.609);

        System.out.println( m + "\t\t" + km);

        }}}

**\*\*5.7**

(*Financial application: compute future tuition*) Suppose the tuition for a university is $10,000 this year and increases 5% every year. In one year, the tuition will be $10,500. Write a program that displays the

tuition in 10 years, and the total cost of four years’ worth of tuition starting after the tenth year.

Sol:

public class aaa {

    public static void main(String args[]) {

        int fouryr = 0;

        int tuition = 10000;

        int inTenYears=0;

        for (int year = 1; year <= 14; year++)

        {   tuition += (tuition \* 0.05);

            if (year == 10)

                inTenYears = tuition;

            if (year > 10)

                fouryr += tuition;}

        System.out.println("Tuition in 10 years is: $" + inTenYears);

        System.out.println("Total cost for four years' worth of tuition after the tenth year is : $" + fouryr);}}

**\*5.9**

(*Find the two highest scores*) Write a program that prompts the user to enter the

number of students and each student’s name and score, and finally displays the

student with the highest score and the student with the second-highest score. Use

the **next()** method in the **Scanner** class to read a name rather than using the

**nextLine()** method. Assume that the number of students is at least 2.

Sol:

import java.util.Scanner;

public class aaa {

    public static void main(String args[]) {

        String name="",max1n="",max2n="";

        int score=0,max1=0,max2=0;

        Scanner Students = new Scanner(System.in);

        System.out.println("Enter number of students");

        int no\_students=Students.nextInt();

        if(no\_students>=2){

            System.out.println("Enter a student name followed by his score then enter "+ no\_students +" times:");

        for(int i=0;i<no\_students;i++){

            name= Students.next();

            score=Students.nextInt();

            if(score > max1)

            {max2=max1;

            max1=score;

            max1n=name;}}

            if(score>max2)

            {max2=score;

            max2n=name;}

        System.out.println("1st highest score is : "+max1n+" with "+ max1 + " points");

        System.out.println("2nd highest score is : "+max2n+" with "+ max2 + " points");}

        else

            System.out.println("Please enter at least two students.");}}

**\*6.2**

(*Sum the digits in an integer*) Write a method that computes the sum of the digits

in an integer. Use the following method header:

**public static int** sumDigits(**long** n)

For example, **sumDigits(234)** returns **9** (= 2 + 3 + 4). (*Hint*: Use the **%** op

erator to extract digits and the **/** operator to remove the extracted digit. For in

stance, to extract 4 from 234, use **234 % 10** (= 4). To remove 4 from 234, use

**234 / 10** (= 23). Use a loop to repeatedly extract and remove the digit until

all the digits are extracted. Write a test program that prompts the user to enter an

integer then displays the sum of all its digits.

Sol:

public class aaa

{

    public static int sumDigits(long n){

        long sum=0;

        int length = String.valueOf(n).length(); //why log 10 of n didn’t work ?

        for(int i=0;i<=length;i++){

            sum += n % 10;

            n /= 10;

        }

        return (int)sum;

    }

    public static void main(String args[]) {

        System.out.println(sumDigits(9839932));

    }}

**\*\*6.3**

(*Palindrome integer*) Write the methods with the following headers:

// Return the reversal of an integer, e.g., reverse(456) re

turns 654

**public static int** reverse(**int** number)

// Return true if number is a palindrome

**public static boolean** isPalindrome(**int** number)

Use the **reverse** method to implement **isPalindrome**. A number is a palin

drome if its reversal is the same as itself. Write a test program that prompts the

user to enter an integer and reports whether the integer is a palindrome.

Sol:

import java.util.Scanner;

public class aaa

{public static int reverse(int number){

int reversedNum=0;

int length = String.valueOf(number).length();

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

reversedNum = reversedNum \* 10 + number % 10;

 number /= 10;}

 return reversedNum;}

 public static boolean isPalindrome(int number)

{

 if(number == reverse(number))

 return true;

else

 return false;}

public static void main(String args[]) {

    int palnum=0;

    Scanner num = new Scanner(System.in);

    System.out.println("Enter a number to check its palindromic status: ");

    palnum=num.nextInt();

    System.out.println("You entered: "+ palnum+ " , is palindrome ? : " +isPalindrome(palnum)); }}

**\*6.5**

(*Sort three numbers*) Write a method with the following header to display three numbers in increasing order: **public static void** displaySortedNumbers( **double** num1, **double** num2, **double** num3)

Write a test program that prompts the user to enter three numbers and invokes the method to display them in increasing order.

import java.util.Scanner;

public class aaa

{public static void displaySortedNumbers(double num1, double num2, double num3)

    {   double temp;

        System.out.println("Before sorting: " +""+ num1+" , "+num2+" , "+num3 );

        if(num2<num1 || num2 < num3)

            {   if(num2<num1) {temp=num1;num1=num2;num2=temp;}

                if(num2<num3) {temp=num3;num3=num2;num2=temp;} }

        if(num3<num1 || num3 < num2)

            { if(num3<num1){temp=num1;num1=num3;num3=temp;}

                if(num3<num2){temp=num2;num2=num3;num3=temp;} }

    System.out.println("After sorting: " +""+ num1+" , "+num2+" , "+num3 );}

public static void main(String args[]){double num1,num2,num3;

    Scanner nums = new Scanner(System.in);

    System.out.println("Please enter three numbers to be sorted :");

    num1= nums.nextDouble();num2= nums.nextDouble();num3= nums.nextDouble();

    displaySortedNumbers(num1,num2,num3);}}

**7.2**

(*Reverse the numbers entered*) Write a program that reads 10 integers then displays them in the reverse of the order in which they were read.

Sol:

import java.util.Scanner;

public class aaa{

public static void reverse(int A[]){

System.out.println("the reverse is :");

    for(int i=9;i>=0;i--)

        System.out.println(A[i]);}

public static void main(String args[]) {

    int Arr[] = new int[10];

    Scanner Arry = new Scanner(System.in);

    for(int i=0;i<10;i++)

        Arr[i]=Arry.nextInt();

    reverse(Arr);}}

**\*7.7**

(*Count single digits*) Write a program that generates 100 random integers between

0 and 9 and displays the count for each number. (*Hint*: Use an array of 10 integers,

say **counts**, to store the counts for the number of 0s, 1s, . . . , 9s.)

Sol:

public class aaa{

public static void main(String args[]) {

    int[] counts = new int[10];

        for (int i = 1; i <= 100; i++)

            counts[(int)(Math.random() \* 10)]++;

    System.out.println("Count for each number between 0 and 9:");

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

            System.out.println(i + "s: " + counts[i]);}}

**7.9**

(*Find the smallest element*) Write a method that finds the smallest element in an

array of double values using the following header:

**public static double** min(**double**[] array)

Write a test program that prompts the user to enter 10 numbers, invokes this

method to return the minimum value, and displays the minimum value.

import java.util.Scanner;

public class aaa{

public static double min(double[] array){

    double minimum=array[0];

    for (int i = 1; i < 10; i++)

        if(minimum>array[i])

            minimum=array[i];

    return minimum;}

public static void main(String args[]){

    Scanner input = new Scanner(System.in);

    double[] nums = new double[10];

    System.out.println("Enter 10 numbers :");

    for (int i = 0; i < 10; i++)

        nums[i]=input.nextDouble();

    System.out.println("The minimun value is : "+ min(nums));}}

**\*8.2**

(*Sum the major diagonal in a matrix*) Write a method that sums all the numbers

in the major diagonal in an *n* \* *n* matrix of **double** values using the following

header:

**public static double** sumMajorDiagonal(**double**[][] m)

Write a test program that reads a 4-by-4 matrix and displays the sum of all its

elements on the major diagonal.

Sol:

import java.util.Scanner;

public class aaa{

    public static double sumMajorDiagonal(double[][] m){

    double sum=0;

    for (int row = 0; row < 4; row++)

        for(int column=0;column<4;column++)

            if(row == column)

                sum+=m[row][column];

    return sum;}

public static void main(String args[]){

    Scanner input = new Scanner(System.in);

    double[][] nums = new double[4][4]; //RozCarry[2][2]={{carry0,carry1,carry2}(roz0)

    System.out.println("Enter a (4x4) matrix row by row:");//,{carry0,carry1,carry2}(roz1),{carry0,carry1,carry2}(roz2)}

    for (int row = 0; row < 4; row++){

        for(int column=0;column<4;column++)

            nums[row][column]=input.nextDouble();

        System.out.println();}

System.out.println(sumMajorDiagonal(nums));}}

**\*8.3**

(*Sort students on grades*) Rewrite Listing 8.2, GradeExam.java, to display the

students in increasing order of the number of correct answers.

**Listing 8.2 GradeExam.java**

1 **public class** GradeExam {

3 **public static void** main(String[] args) {

5 **char**[][] answers = {

6 {**'A'**, **'B'**, **'A'**, **'C'**, **'C'**, **'D'**, **'E'**, **'E'**, **'A'**, **'D'**},

7 {**'D'**, **'B'**, **'A'**, **'B'**, **'C'**, **'A'**, **'E'**, **'E'**, **'A'**, **'D'**},

8 {**'E'**, **'D'**, **'D'**, **'A'**, **'C'**, **'B'**, **'E'**, **'E'**, **'A'**, **'D'**},

9 {**'C'**, **'B'**, **'A'**, **'E'**, **'D'**, **'C'**, **'E'**, **'E'**, **'A'**, **'D'**},

10 {**'A'**, **'B'**, **'D'**, **'C'**, **'C'**, **'D'**, **'E'**, **'E'**, **'A'**, **'D'**},

11 {**'B'**, **'B'**, **'E'**, **'C'**, **'C'**, **'D'**, **'E'**, **'E'**, **'A'**, **'D'**},

12 {**'B'**, **'B'**, **'A'**, **'C'**, **'C'**, **'D'**, **'E'**, **'E'**, **'A'**, **'D'**},

13 {**'E'**, **'B'**, **'E'**, **'C'**, **'C'**, **'D'**, **'E'**, **'E'**, **'A'**, **'D'**}};

16 **char**[] keys = {**'D'**, **'B'**, **'D'**, **'C'**, **'C'**, **'D'**, **'A'**, **'E'**, **'A'**, **'D'**};

19 **for** (**int** i = **0**; i < answers.length; i++) {

21 **int** correctCount = **0**;

22 **for** (**int** j = **0**; j < answers[i].length; j++) {

23 **if** (answers[i][j] == keys[j])

24 correctCount++; }

26 System.out.println(**"Student "** + i + **"'s correct count is "** +

27 correctCount); }}}

Sol:

import java.util.Arrays;

public class aaa{

public static void main(String[] args){

char[][] answers = {

{'A', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D'},

{'D', 'B', 'A', 'B', 'C', 'A', 'E', 'E', 'A', 'D'},

{'E', 'D', 'D', 'A', 'C', 'B', 'E', 'E', 'A', 'D'},

{'C', 'B', 'A', 'E', 'D', 'C', 'E', 'E', 'A', 'D'},

{'A', 'B', 'D', 'C', 'C', 'D', 'E', 'E', 'A', 'D'},

{'B', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D'},

{'B', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D'},

{'E', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D'}};

char[] keys = {'D', 'B', 'D', 'C', 'C', 'D', 'A', 'E', 'A', 'D'};

int correctCount[] = new int[answers[0].length-2];

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

    correctCount[i] =0;

        for (int j = 0; j < answers[i].length; j++)

            if (answers[i][j] == keys[j])

            correctCount[i]++;}

    System.out.println("Before Sort: ");

for(int x=0;x<correctCount.length;x++){

    System.out.println("Student " + (x+1) + "'s correct count is " + correctCount[x]);}

    System.out.println("After Sort: ");

for(int x=0;x<correctCount.length;x++){

    Arrays.sort(correctCount);

    System.out.println("Student " + (x+1) + "'s correct count is " + correctCount[x]);}}}

**\*8.7**

(*Points nearest to each other*) Listing 8.3 gives a program that finds two points in

a two-dimensional space nearest to each other. Revise the program so it finds two

points in a three-dimensional space nearest to each other. Use a two-dimensional

array to represent the points. Test the program using the following points:

**double**[][] points ={{**−1**, **0**, **3**}, {**−1**, **−1**, **−1**}, {**4**, **1**, **1**},{**2**, **0.5**, **9**}, {**3.5**, **2**, **−1**}, {**3**, **1.5**, **3**}, {**−1.5**, **4**, **2**}

The formula for computing the distance between two points

**(x1, y1, z1)** and **(x2, y2, z2)** is

Sol:

public class aaa {

public static void main(String[] args) {

double[][] points = {{-1, 0, 3}, {-1, -1, -1}, {4, 1, 1},

            {2, 0.5, 9}, {3.5, 2, -1}, {3, 1.5, 3}, {-1.5, 4, 2},

            {5.5, 4, -0.5}};

int p1 = 0, p2 = 1, p3 = 3;

double shortestDistance = distance(points[p1][0], points[p1][1], points[p1][2],

points[p2][0], points[p2][p1], points[p3][p2]);

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

    for (int j = i + 1; j < points.length; j++) {

        double distance = distance(points[i][0], points[i][1], points[i][2],

            points[j][0], points[j][1], points[j][2]);

        if (shortestDistance > distance) {

            p1 = i;

            p2 = j;

            shortestDistance = distance;}}

System.out.println("The closest two points are " +

            "(" + points[p1][0] + ", " + points[p1][1] + ") and (" +

                points[p2][0] + ", " + points[p2][1] + ")");}

public static double distance(

    double x1, double y1, double z1, double x2, double y2, double z2) {

    return Math.sqrt(Math.pow(x2 - x1, 2) + Math.pow(y2 - y1, 2) + Math.pow(y2 - y1, 2));}}