Jatiya Kabi Kazi Nazrul Islam University

Trishal, Mymensingh

Lab Report on:

Object Oriented Programming Lab

Course Code: CSE-202

Submitted to:

Selim Al Mamun
Associate Professor,
Department of Computer Science & Engineering.
Jatiya Kabi Kazi Nazrul Islam University

Submitted by:

Arpa Paul
Roll No: 20102006
Reg. No: 8864
Session: 2019-20
2nd year, 1st semester
Department of Computer Science and Engineering
Jatiya Kabi Kazi Nazrul Islam University

Date of submission: **19/09/2022**

Index

Experiment No:01
Experiment Name: Java Program to Find average.
Program code:

```
import java.util.Scanner;
class Averaging{
   public static void main(String[] args) {
      double count=0,sum=0;
      Scanner sc=new Scanner(System.in);
      System.out.println("Enter positive numbers one per line.");
      System.out.println("Indicate end of list with a negative number.");
      while(true){
         double number=sc.nextDouble();
         if(number<0) break;
         sum+=number;
         count++;
      }
      System.out.println("Averge is: "+(sum/count));
   }
}
```

Output:

```
Enter positive numbers one per line.
Indicate end of list with a negative number.
5
0.5
1.3
-1
Average is: 2.1
```

Experiment No:02
Experiment Name: Java Program to Find Grade.
Program code:

```
import java.util.Scanner;
public class Grade {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      while (true) {
         System.out.println("Enter totalMark mark:");
         int totalMark= sc.nextInt();
         System.out.println("Enter Ontained mark: ");
         double obtainedMark = sc.nextDouble();
         if (totalMark < 0 || obtainedMark < 0) {
            System.out.println("Negative value is not acceptable");
            break;
         } else if (obtainedMark > totalMark) {
            System.out.println("Invalid input");
         } else {
            double percentage= (obtainedMark / totalMark) * 100;
            if (percentage>= 0 && percentage< 50) {
               System.out.println("Grade is: F");
            } else if (percentage>= 50 && percentage< 60) {
               System.out.println("Grade is: D");
            } else if (percentage>= 60 && percentage< 70) {
               System.out.println("Grade is: C");
            } else if (percentage>= 70 && percentage< 80) {
               System.out.println("Grade is: B");
            } else {
               System.out.println("Grade is: A");
            }
         }
      }
   }
```

output:

```
Enter totalMark mark:
100
Enter Ontained mark:
90
Grade is: A
Enter totalMark mark:
80
Enter Ontained mark:
50
Grade is: C
Enter totalMark mark:
100
Enter Ontained mark:
101
Invalid input
Enter totalMark mark:
-100
Enter Ontained mark:
50
Negative value is not acceptable.
```

Experiment No:03

Experiment Name: Java program to calculate area, circumference of circle ,and area, perimeter and length of Diagonal and volume and surface area of sphere.

Program code:

```
import java.util.Scanner;

public class Circle {

  public static void main(String[]args) {

   Scanner sc=new Scanner(System.in);

   System.out.println("Enter a real number: ");

   double number=sc.nextInt();

   System.out.println("Area of Circle is: "+(Math.PI*number*number));

   System.out.println("Circumference of Circle is: "+(2*Math.PI*number));

   System.out.println("Area of Square is: "+(number*number));

   System.out.println("Perimeter of Square is: "+(4*number));

   System.out.println("length of diagonal of the Square is :"+(Math.sqrt(2)*number));

   System.out.println("Volume of Sphere is: "+((4/3)*Math.PI*Math.pow(number,3)));

   System.out.println("Surface area of Sphere is :"+(4*Math.PI*number*number));
```

```
    }
}
```

output:

```
Enter a real number:
10
Area of Circle is: 314.1592653589793
Circumference of Circle is: 62.83185307179586
Area of Square is: 100.0
Perimeter of Square is: 40.0
length of diagonal of the Square is :14.142135623730951
Volume of Sphere is: 3141.592653589793
Surface area of Sphere is :1256.6370614359173
```

Experiment No:04

Experiment Name: Java Program to find Character Frequency.

Program code:

```
import java.util.*;
class CharacterFrequency{
   public static int findMaxValue(int[] arr){
      int mx = 0;
      for(int i=0;i<arr.length;i++)
      {
         if(arr[i]>mx)mx = arr[i];
      }
      return mx;
   }
   public static void main(String args[])
   {
      Scanner sc = new Scanner(System.in);
      String s;
      int[] Frequency = new int[26];
      Arrays.fill(Frequency, 0);
      System.out.println("Please enter the sequence of letters ending with `#': ");
      while(true)
```

```
{
   s = sc.nextLine();
   if(s.equals("#"))break;
   for(int i=0;i<s.length();i++)
   {
      char ch = s.charAt(i);
      boolean isAlphabet = Character.isLetter(ch);
      if(ch=='#'){
        boolean  f = false;
          break;
      }
      if(!isAlphabet)continue;
      if(ch>='A' && ch<='Z'){
         ch-='A';
         ch+='a';
      }
      Frequency[ch-'a']++;
   }
}
int mx = findMaxValue(Frequency);
if(mx>0)
{
   System.out.println("The most frequent letters are: ");
   for(int i=0;i<26;i++){
      if(Frequency[i]==mx)
      {
         System.out.printf("%c ",i+'a');
      }
   }
   System.out.println("They occured "+mx+" times");
}
for(int i=0;i<26;i++)
{
   if(Frequency[i]>0)
   {
```

```
                char ch ='a';
                ch+=i;
                System.out.println("'"+ch+"'" + "-"+Frequency[i]+" occurrences.");
            }
        }

    }
}
```

output:

```
Please enter the sequence of letters ending with `#':
c
Med
a
C
A
;Ace .!$$
#
The most frequent letters are:
a c They occured 3 times
'a'-3 occurrences.
'c'-3 occurrences.
'd'-1 occurrences.
'e'-2 occurrences.
'm'-1 occurrences.
```

Experiment No:5

Experiment Name: Java Program to calculate Grading II.

Program code :

```
public class Student {

double tls;

String lstnm, fstnm;

student(){

tls=0;

}

public String toString()

{

return this.tls + " " + this.fstnm + " "+this.lstnm;

}

void totalscore(double mid, double fnl, double hw1, double hw2, double hw3, double hw4, double hw5, String fnme,String lnme) {

tls = (mid * 0.2) + (fnl * 0.3) + ((hw1 + hw2 + hw3 + hw4 + hw5) * 0.1);

fstnm=fnme;

lstnm=lnme;

}

}

import java.io.*;

import java.util.*;

public class test {

public static void main(String[] args) throws Exception {

File stdb = new File("student.txt");

Scanner sc = new Scanner(stdb);

int stnm = sc.nextInt();

student[] sdt = new student[stnm + 1];

for (int i = 0; i < stnm; i++) {
```

```
double mid = sc.nextDouble();

double fnl = sc.nextDouble();

double hw1 = sc.nextDouble();

double hw2 = sc.nextDouble();

double hw3 = sc.nextDouble();

double hw4 = sc.nextDouble();

double hw5 = sc.nextDouble();

String lnme = sc.next();

String fnme = sc.next();

sdt[i]=new student();

sdt[i].totalscore(mid, fnl, hw1, hw2, hw3, hw4, hw5, fnme, lnme);

}

student tmp = new student();

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

for (int j = i+1; j < stnm; j++) {

if(sdt[i].tls < sdt[j].tls){

tmp = sdt[i];

sdt[i] = sdt[j];

sdt[j] = tmp;

}

}

}

System.out.println("INFORMATION read from students.txt\n"+
"=====================================\n");

System.out.println("NUMERICAL LIST\n"+stnm +"\n==========\n"+ "Final
Score\tName\n"+ "---------- --------------------");

for (int i = 0; i < stnm; i++)

System.out.println(sdt[i]);

sc.close();
```

}

}

Input :

5

10 78 12 31 11 34 1 Jennifer Johnson

12 23 45 6 7 7 77 Natalie Robinson
10 20 30 45 34 34 44 hasib hasan

22 33 11 11 22 21 32 rakib hasan

33 45 34 33 11 1 2 Elijah Fisher

Output:

INFORMATION read from students.txt

==========

NUMERICAL LIST

5

==========

Final Score Name

---------- ----------

34.3 Elijah Fisher

28.200000000000003 *Natalie Robinso*

26.7 hasan hasib

24.0 Jennifer Johnson

23.5 kobir hasan

Experiment No: **06**

Experiment Name: Java Program to computes the distance an object will fall in Earth's gravity.

Part-1

Program Code:

```
class GravityCalculator {
public static void main(String[] arguments) {
double gravity = -9.81; // Earth's gravity in m/s^2
double initialVelocity = 0.0;
double fallingTime = 10.0;
double initialPosition = 0.0;
double finalPosition = 0.0;
System.out.println("The object's position after " + fallingTime +
" seconds is " + finalPosition + " m.");
    }
}
/* output of unmodified programe.
The object's position after 10.0 seconds is 0.0 m.
*/
```

Part-2

Program code:

```
class GravityCalculator {
   public static void main(String[] arguments) {
   double gravity = -9.81; // Earth's gravity in m/s^2
   double initialVelocity = 0.0;
   double fallingTime = 10.0;
   double initialPosition = 0.0;
   double finalPosition =
(0.5*gravity*(fallingTime*fallingTime)+(initialVelocity*fallingTime)+initialPosition);
   System.out.println("The object's position after " + fallingTime +
   " seconds is " + finalPosition + " m.");
   }
   }
```

Output:

The object's position after 10.0 seconds is -490.5 m.

Experiment No:07

Experiment name: Java Program to check **Parentheses Balance**

program code:

```
import java.util.*;
public class ParenthesesBalance{
   public static boolean isValid(String str){
      Stack<Character> stack = new Stack<>();
      for(int i=0;i<str.length();i++){
         char c=str.charAt(i);
         if(c=='('||c=='{'||c=='['){
            stack.push(c);
            continue;}
            if(stack.isEmpty())
               return false;
            char check;
         switch (c) {
            case ')':
               check=stack.pop();
               if(check=='{'|| check=='[') return false;
               break;
            case '}':
               check=stack.pop();
               if(check=='('||check=='[') return false;
               break;
            case ']':
               check=stack.pop();
               if(check=='{'||check=='(') return false;
               break;
         }     }
      return (stack.isEmpty());
}
public static void main(String[] args) {
   int testCase;
   Scanner input = new Scanner(System.in);
   testCase = input.nextInt();
   while (testCase!=0){
```

```
            String c = input.next();
            if(isValid(c)) System.out.println("Yes");
            else System.out.println("No");
            testCase--;
        }
    }
}
```

Output:

Sample Input	Sample Output
3	
([])	Yes
(([()]))	No
([()[]()])()	Yes

Experiment No:08

Experiment name: Java Program to **Calculate BMI**

program code:

```
import java.util.*;
class CalculatingBMI{
public static void main(String[] args) {
Scanner sc=new Scanner(System.in);
System.out.print("Enter height in meter: ");
float height=sc.nextFloat();
System.out.print("Enter weight in kg: ");
float weight=sc.nextFloat();
String indicator;
float BMI=weight/(height*height);
if(BMI<=16.00){
System.out.println("starvation");
}
else if(BMI<=16.99){
System.out.println("emaciation");
}
else if(BMI>=17.00&&BMI<=16.99){
System.out.println("underweight");
}
else if(BMI>=18.50&&BMI<=22.99){
System.out.println("normal, low range");
}
else if(BMI>=23.00&&BMI<=24.99){
System.out.println("normal high range");
}
```

```
else if(BMI>=25.00&&BMI<=27.49){
System.out.println("overweight low range");
}
else if(BMI>=27.50&&BMI<=29.99){
System.out.println("overweight high range");
}
else if(BMI>=30.00&&BMI<=34.90){
System.out.println("1st degree obesity");
}
else if(BMI>=35.00&&BMI<=39.99){
System.out.println("2nd degree obesity");
}
else{
System.out.println("3rd degree obesity");
}
}
}
```

Output:

```
Enter height in meter: 1.6
Enter weight in kg: 70
overweight low range
```

Experiment No:9

Experiment name: Java Program to Find the Number of Containers.

program code:

```
import java.util.*;

class Container{

   public static void main(String[] args) {

      System.out.print("Enter a odd number between 50 to 100 :");

      Scanner sc=new Scanner(System.in);

      int odd=sc.nextInt();

      System.out.print("Enter an even number between 5 to 10 : ");

      int evn=sc.nextInt();

      System.out.println("Number of containers : "+(odd/evn));

      System.out.println("1 container is not full.");

      System.out.println("Number of bricks which are in the incomplete container :
"+(odd%evn));
```

```
    }
}
```

Output:

```
Enter a odd number between 50 to 100 :37
Enter an even number between 5 to 10 : 8
Number of containers : 4
1 container is not full.
Number of bricks which are in the incomplete container : 5
```

Experiment No:10

Experiment name: Java Program to Calculate Taxes

program code:

```
import java.util.*;
import java.math.BigDecimal;
class Container{
   public static void main(String[] args) {
      double netValue = 9.99;
      double VAT = 23.0;
      double grossValue = netValue + (VAT*netValue/100);
      System.out.println("The gross value is: "+grossValue);
      double grossValue10000 = grossValue * 10000;
      System.out.println("The gross value for 10000 units is: "+grossValue10000);
      double excludingVAT = grossValue10000 - (VAT*grossValue10000/100);
      System.out.println("The value for 10000 units excluding VAT is: "+excludingVAT);
       System.out.println("\n----Using BigDecimal instead of double----\n");
       BigDecimal netValue_big = new BigDecimal("9.99");
       BigDecimal VAT_big = new BigDecimal("23.0");
       BigDecimal HUNDRED = new BigDecimal("100");
       BigDecimal TenThousand = new BigDecimal("10000");
```

```
        BigDecimal grossValue_big =
netValue_big.add(VAT_big.multiply(netValue_big.divide(HUNDRED)));
        System.out.println("The gross value is: "+grossValue_big);
        BigDecimal grossValue10000_big = grossValue_big.multiply(TenThousand);
        System.out.println("The gross value for 10000 units is: "+grossValue10000_big);
        BigDecimal excludingVAT_big =
grossValue10000_big.subtract(VAT_big.multiply(grossValue10000_big.divide(HUNDRED)));
        System.out.println("The value for 10000 units excluding VAT is: "+excludingVAT_big);
       System.out.println("\nThe accuracy is higher when we use BigDecimal instead of
double");
    }
}
```

Output:

```
The gross value is: 12.287700000000001
The gross value for 10000 units is: 122877.00000000001
The value for 10000 units excluding VAT is: 94615.29000000001
----Using BigDecimal instead of double----
The gross value is: 12.28770
The gross value for 10000 units is: 122877.00000
The value for 10000 units excluding VAT is: 94615.290000
The accuracy is higher when we use BigDecimal instead of double
```

Experiment No:11

Experiment name: Java Program to implements stack without java libraray function.

program code:

```
import java.util.*;
class Stack{
    int stack_array[]=new int[10];
    int top=-1;
   void push(int value){
      top++;
      if(top<10){
```

```
            stack_array[top]=value;
        }
        else{
            System.out.println("Stack Overflow");
        }
    }
    int pop(){
        if(top==-1){
            System.out.println("Stack underflow");
            return 0;
        }
        else
            return stack_array[top--];
    }
}
    public class Test{
        public static void main(String[] args) {
            Stack st=new Stack();
            System.out.println("Stack elements are : ");
            st.push(10);
            st.push(20);
            st.push(30);
            st.push(40);
        System.out.println(st.pop());
        System.out.println(st.pop());
        System.out.println(st.pop());
        System.out.println(st.pop());
        }
    }
```

Output:

```
Stack elements are :
40
30
20
10
```

Experiment No:12

Experiment name: Java Program to implements stack with java libraray function.

program code:

```
import java.util.Stack;

class Stack2{

   public static void main(String[] args) {

      Stack<String>subjects=new Stack<>();

      subjects.push("Physics");

      subjects.push("Chemistry");

      subjects.push("Math");

      subjects.push("Biology");

      System.out.println("Stack Elemets is: "+subjects);

      subjects.pop();

      System.out.println("Stack Elemets after pop: "+subjects);

   }

}
```

Output:

```
Stack Elemets is: [Physics, Chemistry, Math, Biology]
Stack Elemets after pop: [Physics, Chemistry, Math]
```

Experiment No:13

Experiment name: Java Program to implements QUEUE without java libraray function.

program code:

```
class Queue {
        private static int front, rear, capacity;
        private static int queue[];
        Queue(int c)
        {
                front = rear = 0;
                capacity = c;
                queue = new int[capacity];
        }
        static void queueEnqueue(int data)
        {
                if (capacity == rear) {
                        System.out.printf("\nQueue is full\n");
                        return;
                }

                else {
                        queue[rear] = data;
                        rear++;
                }
                return;
        }
        static void queueDequeue()
        {
                if (front == rear) {
                        System.out.printf("\nQueue is empty\n");
                        return;
                }
                else {
                        for (int i = 0; i < rear - 1; i++) {
                                queue[i] = queue[i + 1];
                        }
                        if (rear < capacity)
```

```
                queue[rear] = 0;
            rear--;
        }
        return;
    }
    static void queueDisplay()
    {
        int i;
        if (front == rear) {
            System.out.printf("\nQueue is Empty\n");
            return;
        }

        for (i = front; i < rear; i++) {
            System.out.printf(" %d <-- ", queue[i]);
        }
        return;
    }
    static void queueFront()
    {
        if (front == rear) {
            System.out.printf("\nQueue is Empty\n");
            return;
        }
        System.out.printf("\n Front Element is: %d", queue[front]);
        return;
    }
}
public class StaticQueueinjava {

    public static void main(String[] args)
    {
        Queue q = new Queue(4);
        q.queueDisplay();
        q.queueEnqueue(20);
```

```
                q.queueEnqueue(30);
                q.queueEnqueue(40);
                q.queueEnqueue(50);
                q.queueDisplay();
                q.queueEnqueue(60);
                q.queueDisplay();
                q.queueDequeue();
                q.queueDequeue();
                System.out.printf("\n\nafter two node deletion\n\n");

                q.queueDisplay();
                q.queueFront();
            }
        }
```

Output:

```
Queue is Empty
 20 <--  30 <--  40 <--  50 <--
Queue is full
 20 <--  30 <--  40 <--  50 <--
after two node deletion
 40 <--  50 <--
 Front Element is: 40
```

Experiment No:14

Experiment name: Java Program to implements QUEUE with java libraray function.

program code:

```
import java.util.LinkedList;
import java.util.Queue;
class QueueWithLibrayFunction
{
   public static void main(String[] args)
   {
      Queue<String> queue = new LinkedList<String>();
      queue.add("A");
      queue.add("B");
      queue.add("C");
      queue.add("D");
      System.out.println("The front element is " + queue.peek());
      queue.remove();
      queue.remove();
      System.out.println("The front element is " + queue.peek());
      System.out.println("The queue size is " + queue.size());
      if (queue.isEmpty()) {
         System.out.println("The queue is empty");
      }
      else {
         System.out.println("The queue is not empty");
      }
   }
}
```

Output:

```
The front element is A
The front element is C
The queue size is 2
The queue is not empty
```