**Java Lab Manual**

**Instruction: 1**. Need to install intellJ IDEA By visiting to [www.jetbrains.com](http://www.jetbrains.com)

2. Select Classic mode for download

3. Create New Project

4. Go to source code and select main then editor enables you to write the

Code.

Program 1. Write a Program to find sum of the series 1+1/(2\*2)+1/(3\*3)+1/(4\*4)....+1/(n\*n).

Code:

import java.io.\*;

class GFG {

    // Function to calculate the following series

    static int Series(int n)

    {

        int i;

        int sums = 0;

        for (i = 1; i <= n; i++)

            sums += 1/(i \* i);

        return sums;

    }

    // Driver Code

    public static void main(String[] args)

    {

        int n = 3;

        int res = Series(n);

        System.out.println(res);

    }

}

Out put: 1.36

Program 2. Write a Program for printing Pascal’s Triangle (5Rows) Using Nested Loop

Theory: Pascal’s triangle is a pattern of the triangle which is based on nCr. below is the pictorial representation of Pascal’s triangle.

Example:

Input: N = 5

Output:

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

**Algorithm:**

* Take a number of rows to be printed, assume it to be n
* Make outer iteration i from 0 to n times to print the rows.
* Make inner iteration for j from 0 to (N – 1).
* Print single blank space ” “
* Close inner loop (j loop) //it’s needed for left spacing
* Make inner iteration for j from 0 to i.
* Print nCr of i and j.
* Close inner loop.
* Print newline character (\n) after each inner iteration.

Code:

// Print Pascal's Triangle in Java

import java.io.\*;

class GFG {

    public int factorial(int i)

    {

        if (i == 0)

            return 1;

        return i \* factorial(i - 1);

    }

    public static void main(String[] args)

    {

        int n = 4, i, j;

        GFG g = new GFG();

        for (i = 0; i <= n; i++) {

            for (j = 0; j <= n - i; j++) {

                // for left spacing

                System.out.print(" ");

            }

            for (j = 0; j <= i; j++) {

                // nCr formula

                System.out.print(

                    " "

                    + g.factorial(i)

                          / (g.factorial(i - j)

                             \* g.factorial(j)));

            }

            // for newline

            System.out.println();

        }

    }

}

**Output**

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

Lab Program: 3

Write a Program That accept three numbers from user and prints “increasing “ if the numbers from the user are in increasing order, “decreasing” if the numbers are in increasing order, and “neither” increasing or decreasing order” otherwise.

Code:

import java.util.Scanner;

class Exercise31 {

public static void main(String[] args)

{

Scanner in = new Scanner(System.in);

System.out.print("Input first number: ");

double x = in.nextDouble();

System.out.print("Input second number: ");

double y = in.nextDouble();

System.out.print("Input third number: ");

double z = in.nextDouble();

if (x < y && y < z)

{

System.out.println("Increasing order");

}

else if (x > y && y > z)

{

System.out.println("Decreasing order");

}

else

{

System.out.println("Neither increasing or decreasing order");

}

}

}

Output:

Input first number: 1524

Input second number: 2345

Input third number: 3321

Increasing order

Lab Program: 4

Create Java class called complex with the following details and variables within it as (i) Real (ii) Imaginary.

Develop a Java program to perform addition and subtraction of two complex numbers to perform addition and subtraction of two complex numbers by using add() and subtract() methods by passing object as parameter and display result using method display(). Initialize the real and imaginary values of the complex numbers using parameterized constructer.

Code:

import java.util.\*;

// User Defined Complex class

class Complex {

    // Declaring variables

    int real, imaginary;

    // Empty Constructor

    Complex()

    {

    }

    // Constructor to accept

    // real and imaginary part

    Complex(int tempReal, int tempImaginary)

    {

        real = tempReal;

        imaginary = tempImaginary;

    }

    // Defining addComp() method

    // for adding two complex number

    Complex addComp(Complex C1, Complex C2)

    {

        // creating temporary variable

        Complex temp = new Complex();

        // adding real part of complex numbers

        temp.real = C1.real + C2.real;

        // adding Imaginary part of complex numbers

        temp.imaginary = C1.imaginary + C2.imaginary;

        // returning the sum

        return temp;

    }

    // Defining subtractComp() method

    // for subtracting two complex number

    Complex subtractComp(Complex C1, Complex C2)

    {

        // creating temporary variable

        Complex temp = new Complex();

        // subtracting real part of complex numbers

        temp.real = C1.real - C2.real;

        // subtracting Imaginary part of complex numbers

        temp.imaginary = C1.imaginary - C2.imaginary;

        // returning the difference

        return temp;

    }

    // Function for printing complex number

    void printComplexNumber()

    {

        System.out.println("Complex number: "

                           + real + " + "

                           + imaginary + "i");

    }

}

// Main Class

class GFG {

    // Main function

    public static void main(String[] args)

    {

        // First Complex number

        Complex C1 = new Complex(3, 2);

        // printing first complex number

        C1.printComplexNumber();

        // Second Complex number

        Complex C2 = new Complex(9, 5);

        // printing second complex number

        C2.printComplexNumber();

        // for Storing the sum

        Complex C3 = new Complex();

        // calling addComp() method

        C3 = C3.addComp(C1, C2);

        // printing the sum

        System.out.print("Sum of ");

        C3.printComplexNumber();

        // calling subtractComp() method

        C3 = C3.subtractComp(C1, C2);

        // printing the difference

        System.out.print("Difference of ");

        C3.printComplexNumber();

    }

}

Out Put:

Complex number: 3 + 2i

Complex number: 9 + 5i

Sum of Complex number: 12 + 7i

Difference of Complex number: -6 + -3i

Lab Program 5

|  |
| --- |
| /\* |
|  | \* /\* |
|  | \* Exercise 1.9 |
|  | \* |
|  | \* A class called MyTime, which models a time instance, is designed as shown in the class diagram. |
|  | It contains the following private instance variables: |
|  | hour: between 0 to 23. |
|  | minute: between 0 to 59. |
|  | Second: between 0 to 59. |
|  | The constructor shall invoke the setTime() method (to be described later) to set the instance variable. |
|  | It contains the following public methods: |
|  | setTime(int hour, int minute, int second): It shall check if the given hour, minute and second are valid before setting the instance variables. |
|  | (Advanced: Otherwise, it shall throw an IllegalArgumentException with the message "Invalid hour, minute, or second!".) |
|  | Setters setHour(int hour), setMinute(int minute), setSecond(int second): It shall check if the parameters are valid, similar to the above. |
|  | Getters getHour(), getMinute(), getSecond(). |
|  | toString(): returns "HH:MM:SS". |
|  | nextSecond(): Update this instance to the next second and return this instance. Take note that the nextSecond() of 23:59:59 is 00:00:00. |
|  | nextMinute(), nextHour(), previousSecond(), previousMinute(), previousHour(): similar to the above. |
|  | Write the code for the MyTime class. Also write a test program (called TestMyTime) to test all the methods defined in the MyTime class. |
|  | \*/ |
|  | package oneseven; |
|  |  |
|  | public class TestMyTime { |
|  | public static void main(String[] args) { |
|  | MyTime a = new MyTime(10,59,34); |
|  | System.out.println(a.toString() + " one hour later is " + (a.nextHour()).toString()); |
|  | System.out.println(a.toString() + " one minute later is " + (a.nextMinute()).toString()); |
|  | MyTime b = new MyTime(0,0,0); |
|  | System.out.println("hour of " + b.toString() + " is " + b.getHour()); |
|  | System.out.println(b.toString() + " one second before is " + (b.previousSecond()).toString()); |
|  | } |
|  | } |

Lab Program -6

Assume that a bank maintains two kinds of accounts for customers, one called as savings account and the other as current account. The saving account provides compound interest ad withdrawal facilities but not cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintains a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and opening balance. From this derive the classes Current and Saving to make them more specific to their requirements. Include necessary member functions in order to achieve the following tasks:

(i) deposit an amount for a customer and update the balance

(ii) display the account details

(iii) compute and deposit interest

(iv) withdraw amount for a customer after checking the balance and update the balance.

(v) check for the minimum balance (for current account holders), impose penalty, if necessary, and update the balance.

Implement these without using any constructor

Code:

import java.util.\*;

import java.lang.\*;

class Account

{

public String acc\_name;

public double acc\_no;

public int acc\_type;

public double balance;

public void getdata(String name,double no,int type,double bal)

{

acc\_name=name;

acc\_no=no;

acc\_type=type;

balance=bal;

}

}

class Savings extends Account

{

public void deposit(double amt)

{

balance=balance+amt;

System.out.println(balance);

}

public void withdraw(double amt)

{

balance=balance-amt;

System.out.println(balance);

}

public void interest(int time,int no)

{

double intr=balance\*(1+6/no);

intr=Math.pow(intr,(time\*no));

System.out.println("Intertest calculated is"+intr);

balance=balance+intr;

System.out.println("The new balance is"+balance);

}

}

class Current extends Account

{

public void deposit(double amt)

{

balance=balance+amt;

System.out.println(balance);

}

public void withdraw(double amt)

{

balance=balance-amt;

System.out.println(balance);

check(balance);

}

public void check(double amt)

{

if(amt<10000)

{

balance =balance-500;

System.out.println("Insufficient Balance"+balance);

}

}

}

class Main

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int temp=1;

while(temp==1)

{

double amt=0;

System.out.println("Enter name");

sc.next();

String name=sc.nextLine();

System.out.println("Enter acc\_no");

double no=sc.nextDouble();

System.out.println("Enter acc\_type\n0 for Savings\n1 for Current");

int type=sc.nextInt();

do

{

System.out.println("Enter balance");

amt=sc.nextDouble();

}while(type==1 && amt<10000);

if(type==0)

{

Savings s=new Savings();

s.getdata(name,no,type,amt);

System.out.println("\n1.Deposit\n\2.Withdraw\n3.Interest");

int temp3=sc.nextInt();

if(temp3==1)

{

System.out.println("Enter Amoumt");

double amt1=sc.nextDouble();

s.deposit(amt1);

}

else if(temp3==2)

{

System.out.println("Enter Amoumt");

double amt1=sc.nextDouble();

s.withdraw(amt1);

}

else if(temp3==3)

{

System.out.println("Enter time period");

int tp=sc.nextInt();

System.out.println("Enter no of times");

int nof=sc.nextInt();

s.interest(tp,nof);

}

}

else if(type==1)

{

Current c=new Current();

c.getdata(name,no,type,amt);

System.out.println("\n1.Deposit\n\2.Withdraw");

int temp3=sc.nextInt();

if(temp3==1)

{

System.out.println("Enter Amoumt");

double amt1=sc.nextDouble();

c.deposit(amt1);

}

else if(temp3==2)

{

System.out.println("Enter Amoumt");

double amt1=sc.nextDouble();

c.withdraw(amt1);

}

}

System.out.println("To continue 1 else 0");

temp=sc.nextInt();

}

}

}