OOP Lab-01 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Exercises

Exercise 1

Create a new java project on NetBeans named as “First_Exercise”. Display a text message saying, “this is my first program”. Provide multiline document string for the created program explaining the process in the program

Code:

```java
package com.mycompany.first_exercise;

public class First_Exercise {

    public static void main(String[] args) {
        System.out.println(x: "First Program");

        /*Step1: Open NetBeans IDE and Create a new project
        Step2: Rename this project to first Exercise
        Step3: print "First Program" on console :)
        */
    }
}
```

Output:

```
First Program
------------------------------------------
BUILD SUCCESS
------------------------------------------
Total time:  6.898 s
Finished at: 2023-09-23T07:43:56+05:00
------------------------------------------
```

Exercise 2

Create a java program named “Second_Exercise”. Print the name, enrollment, Subject name, Instructor name and semester of a student and attach proper output screen shot. Provide proper comments.

Code:

```java
package com.mycompany.second_exercise;

public class Second_Exercise {

    public static void main(String[] args) {
        System.out.println(x: "Name: Syed Muhammad Raza Ali");
        //printed my name
        System.out.println(x: "Enrolment: 02-134231-028");
        //printed my enrolment number
        System.out.println(x: "Subject: Object Oriented Programming Lab");
        //printed my subject
        System.out.println(x: "Lab Instructor: Miss Hafsa Munawar");
        //printed my teacher's name
        System.out.println(x: "Semester: BSCS-2 (fall-2023)");
        //printed my semester
    }

}
```

Output:

```
Name: Syed Muhammad Raza Ali
Enrolment: 02-134231-028
Subject: Object Oriented Programming Lab
Lab Instructor: Miss Hafsa Munawar
Semester: BSCS-2 (fall-2023)
------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------
Total time:  1.627 s
```

OOP Lab-02 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Exercise 1 *(Mosque.java)*

Write a program that prints a mosque, similar to the following:

```
      ^              ^              ^
     /|\            /|\            /|\
    ((&))          .((^)).        ((&))
     |.|          ((((^))))        |.|
     |.|         ((((((|)))))      |.|
     |.|        ((((((|))))))      |.|
     |.|         (((((|)))))       |.|
     |.|          (((((|)))))      |.|
   {"""""""""""""""""""""""""""""""""""""}
    " |"""""""""""""""""""""""""""""""""| "
      |                                  |
      |                _                 |
      |              _{#}_               |
      |   ____      {#####}      ____    |
      |  |###|     {#######}    |###|    |
      |  |###|     {#######}    |###|    |
      |  =====     {#######}    =====    |
      |            {#######}             |
      |            {#######}             |
      |           _{#######}_            |
    (@@@@@@@@@@@@@==========@@@@@@@@@@@@)
   (++++++++++/=============\++++++++++)
   [%%%%%%%%%/===============\%%%%%%%%%]
   """""""""/=================\"""""""""
```

Code:

```
package com.mycompany.mavenproject1;
public class Mavenproject1 {
public static void main(String[] args) {
        System.out.println("      ^           ^           ^      ");
        System.out.println("     //|\\        //|\\        //|\\     ");
        System.out.println("    (((&)))      .(((^))).      (((&)))    ");
        System.out.println("      |.|          |.|          |.|      ");
        System.out.println("      |.|          |.|          |.|      ");
        System.out.println("      |.|          |.|          |.|      ");
        System.out.println("      |.|          |.|          |.|      ");
        System.out.println("      |.|          |.|          |.|      ");
        System.out.println(" {""""""""""""""""""""""""""""} ");
        System.out.println("  '||""""""""""""""""""""""""||'  ");
        System.out.println("   ||                          ||   ");
        System.out.println("   ||              _           ||   ");
        System.out.println("   ||             {#}          ||   ");
        System.out.println("   ||            {#####}         ||   ");
        System.out.println("   ||           {#######}        ||   ");
        System.out.println("   ||           {#######}        ||   ");
        System.out.println("   ||           {#######}        ||   ");
        System.out.println("   ||           {#######}        ||   ");
        System.out.println("   ||          {#######}        ||   ");
        System.out.println("
(@@@@@@@@@@@@@@@@==========@@@@@@@@@@@@@@@@) ");
```

```
        System.out.println("(++++++++++++//=============\\++++++++++++) ");
        System.out.println("[%%%%%%%%%%%%//===============\\%%%%%%%%%%%%]");
        System.out.println("""""""""//=================\\""""""""");
        System.out.println("         //===================\\         ");
    }
}
```

Output:

```
--- exec:3.1.0:exec (default-cli) @ mavenproject1 ---
        ^              ^              ^
      //|\          //|\          //|\
     (((&)))       .(((^))).       (((&)))
       |.|            |.|            |.|
       |.|            |.|            |.|
       |.|            |.|            |.|
       |.|            |.|            |.|
       |.|            |.|            |.|
  {''''''''''''''''''''''''''''''''''''''''''''}
   '||''''''''''''''''''''''''''''''''''''''||'
    ||                                      ||
    ||                                      ||
    ||                 _                    ||
    ||               {#}                  ||
    ||              {#####}                 ||
    ||             {#######}                ||
    ||             {#######}                ||
    ||             {#######}                ||
    ||             {#######}                ||
    ||            {#######}                ||
   (@@@@@@@@@@@@@@@@==========@@@@@@@@@@@@@@@@)
  (++++++++++++//=============\++++++++++++)
  [%%%%%%%%%%%%//===============\%%%%%%%%%%%%]
  '''''''''''//=================\'''''''''''
             //===================\
  ------------------------------------------------------------
  BUILD SUCCESS
  ------------------------------------------------------------
```

Exercise 2 **(Equations.java)**

Wrtite a java program that calculates the following equation. Where x = 6, y = 20, z=13

- $2x^2 + y^2$
- $3x + y - 3z^2$
- $2x - 2y + 5z^2$

Code:

```
package javaapplication15;
import java.util.Scanner;


public class JavaApplication15 {
    static int equation1(int x, int y){
        return (2*(x*x) + (y*y));
    }
    static int equation2(int x,int y,int z){
        return (3*x + y -3*(z*z));
    }
    static int equation3(int x,int y,int z){
        return (2*x -2*y + 5*(z*z));
    }
public static void main(String[] args) {

            Scanner input = new Scanner(System.in);
            System.out.println("Enter the value of x: ");
        int x = input.nextInt();
        System.out.println("Enter the value of y: ");
        int y = input.nextInt();
        System.out.println("Enter the value of z: ");
        int z = input.nextInt();

        System.out.println(equation1(x,y));
        System.out.println(equation2(x,y,z));
        System.out.println(equation3(x,y,z));
}
}
```

Output:

```
Enter the value of x:
6
Enter the value of y:
20
Enter the value of z:
13
472
-469
- 817
--------------------------------
BUILD SUCCESS
--------------------------------
```

Exercise 3 *(Arithmatic.java)*

Type-in the following example, which receives the input of two integer numbers and compute the sum, difference and product. Compile and run this program.

Code:

```
package javaapplication15;
import java.util.Scanner;



public class JavaApplication15 {
   static int sum(int a,int b){
      return a+b;
   }
    static int sub(int a,int b){
      return a-b;
   }
     static int mul(int a,int b){
      return a*b;
   }
     static int div(int a,int b){
      return a/b;
   }
 public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
```

```
        System.out.println("Enter the value of a:");
        int a = input.nextInt();
        System.out.println("Enter the value of b:");
        int b = input.nextInt();

        System.out.println("Sum of "+a+" "+" and "+b+" is "+sum(a,b));
        System.out.println("Sum of "+a+" "+" and "+b+" is "+sub(a,b));
        System.out.println("Sum of "+a+" "+" and "+b+" is "+mul(a,b));
        System.out.println("Sum of "+a+" "+" and "+b+" is "+div(a,b));
}
}
```

Output:

```
Enter the value of a:
12
Enter the value of b:
3
Sum of 12  and 3 is 15
Sum of 12  and 3 is 9
Sum of 12  and 3 is 36
Sum of 12  and 3 is 4
----------------------------
BUILD SUCCESS
----------------------------
```

Exercise 4 *(Temperature.java)*

Celsius to Fahrenhite temperature: **F**=(**C** × 9/5) + 32

C = temperature in celsius.

F = temperature in fahrenhite

Calculate the temperature for the following degrees

- 289 °C
- 400 °C
- -36 °C
- -180 °C

Code:

```
package javaapplication15;
import java.util.Scanner;


public class JavaApplication15 {
   static int convertTemp(int C){
      int F = ((C*9/5)+32);
      return F;
   }   public static void main(String[] args) {
      Scanner temp = new Scanner(System.in);
      System.out.println("Enter the temperature in Celcius");
      int tempInC = temp.nextInt();
      System.out.println("Temperature in Celsius =  "+tempInC);
      System.out.println("Temperature in Fahrenheit"+convertTemp(tempInC));
      }
}
```

Output (for 289C):

```
Enter the temperature in Celcius
 289
Temperature in Celsius =  289
Temperature in Fahrenheit552
------------------------------------------------
BUILD SUCCESS
------------------------------------------------
Total time:  7.895 s
Finished at: 2023-09-30T10:23:58+05:00
------------------------------------------------
```

Output (for 400C):

```
Enter the temperature in Celcius
400
Temperature in Celsius =  400
Temperature in Fahrenheit752
------------------------------------
BUILD SUCCESS
------------------------------------
```

Output (for-36C):

```
Enter the temperature in Celcius
-36
Temperature in Celsius =  -36
Temperature in Fahrenheit-32
---------------------------------
BUILD SUCCESS
---------------------------------
Total time:  4.732 s
```

Output (for-180C):

```
Enter the temperature in Celcius
-180
Temperature in Celsius =  -180
Temperature in Fahrenheit-292
---------------------------------
BUILD SUCCESS
---------------------------------
Total time:  5.025 s
```

Exercise 5 *(Cookies.java)*

There are 12 cookies per box (sold at $1.14) and 24 boxes per carton. Left over boxes are sold for 57¢. Remaining cookies are given away free. Given the number of cookies produced, determine the number of boxes, cartons, left over boxes and the total money made.

Code:

```
package com.mycompany.task05;
import java.util.Scanner;

public class Task05 {

    public static void main(String[] args) {



            Scanner sc =  new Scanner (System.in);
        System.out.println("Enter number of cookies produced");
        int numbers = sc.nextInt();

        int boxes=numbers/12;

        System.out.println("no of boxes = "+boxes);
        int cartons = boxes/24;

      System.out.println("Number of cartoons : "+cartons);
```

```
        int leftover = cartons%24;
        System.out.println("Number of leftovers:"+leftover);
        double totalmoney = (cartons*1.14);
        double leftovermoney = (leftover * 57);
        System.out.println("Totalmoney : "+totalmoney);
        System.out.println("leftover : "+leftover);
        System.out.println("Leftover money: " + leftovermoney);




    }

}
```

Output:

```
Enter number of cookies produced
2000
no of boxes = 166
Number of cartoons : 6
Number of leftovers:6
Totalmoney : 6.84
leftover : 6
Leftover money: 342.0
--------------------------------
BUILD SUCCESS
--------------------------------
```

Exercise 6 *(PullyFormulas.java)*

Pulley formulas

a) calculate the speed of one pulley if there are 2 pulleys connected with a belt:
   RPM2 = diameter1/diameter2 * RPM1

b) calculate the amount of weight that can be lifted with a multiple pulley system:
   weight lifted = force exerted * number of up ropes

Code:

```
package javaapplication15;
import java.util.Scanner;


public class JavaApplication15 {




    static int rpm2(int rpm1,int diameter1, int diameter2){
        int rpm2 = (diameter1/diameter2 )* rpm1;
        return rpm2;
    }
    static int weightLifted(int forceExerted, int ropes){
        return forceExerted * ropes;
    }
    public static void main(String[] args) {





        Scanner input = new Scanner(System.in);
        System.out.println("Ente diameter 1 : ");
        int diameter1 = input.nextInt();
         System.out.println("Ente diameter 2 : ");
        int diameter2 = input.nextInt();
         System.out.println("Ente RPM 1 : ");
        int rpm1 = input.nextInt();


        System.out.println("The Speed of pulley(RPM2) = "+rpm2(rpm1,diameter1,diameter2));
        System.out.println("Enter the Force Exerted : ");
        int forceExerted = input.nextInt();
          System.out.println("Enter the number of  up ropes: ");
        int ropes = input.nextInt();

        System.out.println("The Amount of lifted weight  = "+weightLifted(forceExerted,ropes));
```

```
  }

}
```

Output:

```
Ente diameter 1 :
20
Ente diameter 2 :
10
Ente RPM 1 :
333
The Speed of pulley(RPM2) = 666
Enter the Force Exerted :
120
Enter the number of  up ropes:
30
The Amount of lifted weight  = 3600
---------------------------------------------
BUILD SUCCESS
---------------------------------------------
```

OOP Lab-03 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Exercises

Exercise 1 *(StringParser.java)*

Write a java program that have this string “Hello! I am string in java. I have several function and I am very “Important”# string_is _importnat” and split it as follows.

```
run:
Original : Hello! I am string in Java.I have several Functions and I am very "Important" #string_is_importnat

-------------- Splitting Strings ------------------

Splitting at ! : Hello!
After ! till Important :  I am string in Java
After . till # : I have several Functions and I am very "Important"
After # till end : #string_is_importnat
BUILD SUCCESSFUL (total time: 0 seconds)
```

Code:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

 class Tasks{
    //Task---01
 public void Task1(){
    String myStr = "Hello! I am a String in java. I have several functions and I am very 'important'
#String_is_important";
    System.out.println("-------------- Spliting Strings -------------- ");
    System.out.println(myStr);
    System.out.println("Spliting at ! :"+myStr.substring(0,myStr.indexOf("!")+1));
    System.out.println("Splitting from ! to Important : "+myStr.substring( (myStr.indexOf("!")+1),
          (myStr.lastIndexOf("""")+1)));
    System.out.println("Splitting from Important to # : "+myStr.substring((myStr.lastIndexOf("""")+1)
          ,myStr.lastIndexOf("t")+1) );
 }
}
public class Mavenproject2 {



   public static void main(String[] args) {
      Tasks taskObj = new Tasks();
      taskObj.Task1();
}
}
```

Output:

```
-------------- Spliting Strings --------------
Hello! I am a String in java. I have several functions and I am very 'important' #String_is_important
Spliting at ! :Hello!
Splitting from ! to Important :  I am a String in java. I have several functions and I am very 'important'
Splitting from Important to # :  #String_is_important
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```

Exercise 2 *(Weekdays.java)*

Write a Java program that keeps a number from the user and generates an integer between 1 and 7 and displays the name of the weekday as follows.

Code:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

class Tasks{

public void Task2(){
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter a number between 1 to 7 : ");
    int day = sc.nextInt();
    if(day == 1)
        System.out.println("It is sunday");
    else if(day == 2)
        System.out.println("It is Monday");
    else if(day == 3)
            System.out.println("It is Tuesday");
    else if(day == 4)
        System.out.println("It is Wednesday");
    else if(day == 5)
        System.out.println("It it Thursday");
    else if(day == 6)
        System.out.println("It is Friday");
    else if(day == 7)
        System.out.println("It ts Saturday");
    else
        System.out.println("Invalid Input!!");
}
```

```
}
public class Mavenproject2 {


    public static void main(String[] args) {
        Tasks taskObj = new Tasks();
        taskObj.Task2();
    }

}
```

Output:

```
Enter a number between 1 to 7 :
4
It is Wednesday
-------------------------------------------------------
BUILD SUCCESS
-------------------------------------------------------
Total time:  5.441 s
```

Exercise 3 *(Alphabets.java)*

Write a Java program that takes the user to provide a single character from the alphabet. Print Vowel or Consonant, depending on the user input. If the user input is not a letter (between a & z or A & Z), or is a string of length > 1, print an error message.

Code:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

 class Tasks{
 public void Task3(){
     Scanner sc = new Scanner(System.in);
        System.out.println("Enter an alphabet : ");
        String alphabet = sc.nextLine();
        char myChar = alphabet.charAt(0);

        if(myChar == 'a' ||myChar == 'A' || myChar == 'e' || myChar == 'E' || myChar == 'i' || myChar ==
'I' || myChar == 'o' ||
                myChar == 'O' ||myChar == 'u' ||myChar == 'U'){
            System.out.println(myChar+" is a Vowel");
```

```
        }
        else{
          System.out.println(myChar+" is a consonant");
        }
}
public class Mavenproject2 {


    public static void main(String[] args) {
        Tasks taskObj = new Tasks();
        taskObj.Task3();
    }

}
```

Output:

```
Enter an alphabet :
o
o is a Vowel
---------------------------
BUILD SUCCESS
---------------------------
Total time:  7.556 s
```

Exercise 4 *(Occurrances.java)*

Write a java program that gets input from a user into an array (size defined by user) then find the max and min numbers found in array as well as the index at which they are found at. Then calculate the difference between two values and the difference between indexes as well. See screen shot for reference. HINT: use java.lang.Math.abs package to print absolute value between index differences to avoid negative value.

Code:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

 class Tasks{

 //Task---04
 public void Task4(){
    int size = 0;
```

```
Scanner sc = new Scanner(System.in);
System.out.println("Enter the size of Array: ");
size = sc.nextInt();

int[] arr = new int[size];
//arr input
for(int i =0;i<size;i++){
    System.out.println("Enter the value of index of: "+i);
     arr[i] =sc.nextInt();
}
//arr output
for(int i = 0;i<size;i++){
    System.out.println("The value ofindex no "+i+" is "+arr[i]);
}

int maxNumber = arr[0];
int minNumber = arr[0];
int indexOfMin = 0;
int indexOfMax = 0;
//for maxNumber
for(int i = 0;i<size;i++){
    if(arr[i]>maxNumber){
        maxNumber = arr[i];
        indexOfMax =i;
    }
}

  //for minNumber
for(int i = 0;i<size;i++){
    if(arr[i]<minNumber){
        minNumber = arr[i];
        indexOfMin =i;
    }
}


System.out.println("Size of Array : "+size);
System.out.println("Max value of array : "+maxNumber);
System.out.println("Index of Max value : "+indexOfMax);
System.out.println("Min value of array : "+minNumber);
```

```
    System.out.println("Index of Min value : "+indexOfMin);



 }
}
public class Mavenproject2 {


   public static void main(String[] args) {
      Tasks taskObj = new Tasks();
      taskObj.Task4();
   }

}
```

Output:

```
Enter the size of Array:
5
Enter the value of index of: 0
12
Enter the value of index of: 1
31
Enter the value of index of: 2
76
Enter the value of index of: 3
5
Enter the value of index of: 4
89
The value ofindex no 0 is 12
The value ofindex no 1 is 31
The value ofindex no 2 is 76
The value ofindex no 3 is 5
The value ofindex no 4 is 89
Size of Array : 5
Max value of array : 89
Index of Max value : 4
Min value of array : 5
Index of Min value : 3
---------------------------------
BUILD SUCCESS
---------------------------------
```

Exercise 6

(StringReplacement.java)

Write a Java program to replace each substring of a given string that matches the given regular expression with the given replacement.
Sample string: "this is the sample exercise of OOP basics."
Replace OOP withn ICT.

Code:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

 class Tasks{

 //Task ---06
 public void Task6(){
    String myStr = "This is the exercise of OOP basics";
    System.out.println("String before replacing : ");
    System.out.println(myStr);
    System.out.println("String after replacing : ");
    String newStr = myStr.replace("OOP", "IICT");
    System.out.println(newStr);
 }
}

public class Mavenproject2 {


   public static void main(String[] args) {
      Tasks taskObj = new Tasks();
      taskObj.Task6();
   }

}
```

Output:

```
String before replacing :
This is the exercise of OOP basics
String after replacing :
This is the exercise of IICT basics
------------------------------------
BUILD SUCCESS
------------------------------------
```

OOP Lab-04 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Designing and implementing Java programs that deal with:

1. **Static Methods**
2. **Recursion**

Exercises

Exercise 1 *(PatientInfo.java)*

Consider you are a receptionist at hospital and whenever the patient comes you're to take his following info P_number, P_Name, P_age , P_email, P_contact, P_Complain and P_bill then print the receipt for customer so method responsible for taking customer's info is called as Take_Patient_data() and method responsible print receipt is called as Print_Receipt()

Hint: Create global variable that is outside of the main method and use them in both methods for taking and printing customer's details

NOTE: These functions must not be static

Code:

Patient Class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

public class Patient {
    String p_name,p_email,p_complain;
    int p_number,p_age,p_bill,p_contact;

    public void takePatientRecord(){
        Scanner p_input = new Scanner(System.in);
        System.out.print("Enter your'e Name : ");
        p_name = p_input.nextLine();
         System.out.print("Enter your'e Email : ");
        p_email = p_input.nextLine();
         System.out.print("Enter your'e Complain : ");
        p_complain = p_input.nextLine();
         System.out.print("Enter your'e Patient ID : ");
        p_number = p_input.nextInt();
         System.out.print("Enter your'e Age : ");
```

```
        p_age = p_input.nextInt();
         System.out.print("Enter your'e Contact no : ");
        p_contact = p_input.nextInt();
        p_input.nextLine();
         System.out.print("Enter your'e Bill : ");
        p_bill = p_input.nextInt();
    }

    public void printReceipt(){
        System.out.println("===================== Reciept =====================");
        System.out.println("Patient's Name : "+ p_name);
        System.out.println("Patient's Email : "+ p_email);
        System.out.println("Patient's Complain : "+ p_complain);
        System.out.println("Patient's ID : "+ p_number);
        System.out.println("Patient's Age : "+ p_age);
        System.out.println("Patient's Contact : "+ p_contact);
        System.out.println("Patient's Bill : "+ p_bill);

System.out.println("====================================================");


    }
}
```

Application Class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

public class Mavenproject2 {

    public static void main(String[] args) {
//        Task--01
        Patient obj = new Patient();
        obj.takePatientRecord();
        obj.printReceipt();
}
}
```

Output:

```
Enter your'e Name : Syed Raza Ali
Enter your'e Email : asyedraza85632
Enter your'e Complain : fever
Enter your'e Patient ID : 134231
Enter your'e Age : 19
Enter your'e Contact no : 127182
Enter your'e Bill : 20000
===================== Reciept =====================
Patient's Name : Syed Raza Ali
Patient's Email : asyedraza85632
Patient's Complain : fever
Patient's ID : 134231
Patient's Age : 19
Patient's Contact : 127182
Patient's Bill : 20000
====================================================
----------------------------------------------------
BUILD SUCCESS
```

Exercise 2 *(Sum.java)*

Write the following 2 static methods:

`public static int ComputeOddSum(int input)`

`public static int ComputeEvenSum(int input)`

The method **`ComputeOddSum`** find the sum of all odd numbers less than input (should be recursive function).

The method **`ComputeEvenSum`** find the sum of all even numbers less than input.

Code:

Task2 Class:

```
package com.mycompany.mavenproject2;
public class Task2 {
    static int i = 0;
    static int sum = 0;

    //for Odd Numbers
    public static int computeOddSum(int number){
        if(i<number){
            if(i%2 != 0){
                sum+=i;
            }
            i+=1;
            computeOddSum(number);
        }
        i = 0;
        return sum;
    }

    //For Even Numbers
     public static int computeEvenSum(int number){
        if(i<number){
            if(i%2 == 0){
                sum+=i;
            }
            i+=1;
            computeEvenSum(number);
        }
        i = 0;
        return sum;
    }
}
```

Application class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

public class Mavenproject2 {

    public static void main(String[] args) {
        //task---02
        Scanner input = new Scanner(System.in);
        System.out.println("Enter an integar ");
        int number = input.nextInt();
        System.out.println("The sum of all Odd numbers = "+Task2.computeOddSum(number));
        System.out.println("The sum of all Even numbers = "+Task2.computeEvenSum(number));
}
}
```

Output:

```
Enter an integar
20
The sum of all Odd numbers = 100
The sum of all Even numbers = 190
------------------------------------
BUILD SUCCESS
------------------------------------
```

Exercise 3 *(MatrixTest.java)*

Create a Matrix named as Mat_1 of size 3x3 and ask user to insert values take another matrix named as Mat_2 of size 3x3 again and then implement following equations

1. (Mat_1*3)+(Mat_2)*2
2. (Mat_2 -3) *2
3. (Mat_2*5) – (Mat_1-2)

Code:

Task3 Class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

    class Task3{
    int[][] mat_1 = new int[3][3];
    int[][] mat_2 = new int[3][3];
    Scanner input = new Scanner(System.in);

    //for tasking inputs in arrays
    void inputInArrays(){
        //For mat_1
        System.out.println("Enter the elements of mat_1");
        for(int i = 0;i<mat_1.length;i++){
            for(int j = 0;j<mat_1.length;j++){
                System.out.print("Enter the value of position "+i+""+j+" : ");
                mat_1[i][j] = input.nextInt();
            }
        }

         //For mat_2
                 System.out.println("Enter the elements of mat_2");

        for(int i = 0;i<mat_2.length;i++){
            for(int j = 0;j<mat_2.length;j++){
                System.out.print("Enter the value of position "+i+""+j+" : ");
                mat_2[i][j] = input.nextInt();
            }
        }

        //printing arrays
        System.out.println("================== Arrays After Taking Input
==================");
```

```
    System.out.println("mat_1 is given as : ");
    for(int i = 0; i<mat_1.length; i++){
        for(int j = 0; j<mat_1.length; j++){
            System.out.print(mat_1[i][j] + "\t");
        }
        System.out.print("\n");
    }

    System.out.println("mat_2 is given as : ");
      for(int i = 0; i<mat_2.length; i++){
        for(int j = 0; j<mat_2.length; j++){
            System.out.print(mat_2[i][j] + "\t");
        }
        System.out.print("\n");
    }

}

//For Equation1
void equation1(){
    //mat_1 * 3
    for(int i = 0; i<mat_1.length; i++){
        for(int j = 0; j<mat_1.length; j++){
            mat_1[i][j] = (mat_1[i][j]) * 3;
        }
    }

    //mat_2 * 2
       for(int i = 0; i<mat_2.length; i++){
        for(int j = 0; j<mat_2.length; j++){
           mat_2[i][j] = (mat_2[i][j]) * 2;
        }
    }
```

```
        //adding both arrys
        int[][] mat_ans = new int[3][3];
        for(int i = 0; i<mat_ans.length; i++){
            for(int j = 0; j<mat_ans.length; j++){
                mat_ans[i][j] = mat_1[i][j] + mat_ans[i][j];
            }
        }

        //printing result
        System.out.println("================== Arrays After Performing Eq1
==================");
        System.out.println("(Mat_1*3) + (Mat_2*2)");
        for(int i = 0; i<mat_ans.length; i++){
            for(int j = 0; j<mat_ans.length; j++){
                System.out.print(mat_ans[i][j] +"\t");
            }
            System.out.print("\n");
        }
    }

    //for equation 2
    void equation2(){
        //mat_2 - 3
        for(int i = 0; i<mat_2.length; i++){
            for(int j = 0; j<mat_2.length; j++){
                mat_2[i][j] = (mat_2[i][j]) - 3;
            }
        }

        //mat_2-3 *2
        int[][] mat_ans = new int[3][3];
        for(int i = 0; i<mat_ans.length; i++){
            for(int j = 0; j<mat_ans.length; j++){
                mat_ans[i][j] = mat_2[i][j] * 2;
            }
        }
```

```
        //printing final array
        System.out.println("================== Arrays After Performing Eq2
==================");
        System.out.println("(Mat_2-3)*2");
        for(int i = 0; i<mat_ans.length; i++){
            for(int j = 0; j<mat_ans.length; j++){
                System.out.print(mat_ans[i][j] +"\t");
            }
            System.out.print("\n");
        }
    }



    //for equation3
      void equation3(){
          //mat_2*5
          for(int i = 0; i<mat_2.length; i++){
              for(int j = 0; j<mat_2.length; j++){
              mat_2[i][j] = (mat_2[i][j]) * 5;
          }
          }

          //mat_1-2
          for(int i = 0; i<mat_1.length; i++){
              for(int j = 0; j<mat_1.length; j++){
                  mat_1[i][j] = (mat_1[i][j]) - 2;
              }
          }

          //for final array
          int[][] mat_ans = new int[3][3];
          for(int i = 0; i<mat_ans.length; i++){
              for(int j = 0; j<mat_ans.length; j++){
                  mat_ans[i][j] = mat_2[i][j] - mat_1[i][j];
              }
```

```
        }


        //printing final array
        System.out.println("================== Arrays After Performing Eq3
==================");
      System.out.println("(Mat_2*5)-(Mat_1*2)");
      for(int i = 0; i<mat_ans.length; i++){
         for(int j = 0; j<mat_ans.length; j++){
            System.out.print(mat_ans[i][j] +"\t");
         }
         System.out.print("\n");
      }
     }

}
```

Application class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

public class Mavenproject2 {

   public static void main(String[] args) {
      //task---03
      Task3 obj = new Task3();
      obj.inputInArrays();
      obj.equation1();
      obj.equation2();
      obj.equation3();
}
}
```

Output (For eq1):

```
Enter the elements of mat_1
Enter the value of position 00 : 12
Enter the value of position 01 : 34
Enter the value of position 02 : 65
Enter the value of position 10 : 3
Enter the value of position 11 : 78
Enter the value of position 12 : 4
Enter the value of position 20 : 2
Enter the value of position 21 : 98
Enter the value of position 22 : 12
Enter the elements of mat_2
Enter the value of position 00 : 65
Enter the value of position 01 : 45
Enter the value of position 02 : 123
Enter the value of position 10 : 90
Enter the value of position 11 : 435
Enter the value of position 12 : 23
Enter the value of position 20 : 23
Enter the value of position 21 : 87
Enter the value of position 22 : 45
================== Arrays After Taking Input =====
mat_1 is given as :
12      34      65
3       78      4
2       98      12
mat_2 is given as :
65      45      123
90      435     23
23      87      45
================== Arrays After Performing Eq1 ===
(Mat_1*3) + (Mat_2*2)
36      102     195
9       234     12
6       294     36
-------------------------------------------------
BUILD SUCCESS
```

Output (For eq2):

```
Enter the elements of mat_1
Enter the value of position 00 : 2
Enter the value of position 01 : 8
Enter the value of position 02 : 4
Enter the value of position 10 : 0
Enter the value of position 11 : 12
Enter the value of position 12 : 5
Enter the value of position 20 : 87
Enter the value of position 21 : 32
Enter the value of position 22 : 6
Enter the elements of mat_2
Enter the value of position 00 : 76
Enter the value of position 01 : 12
Enter the value of position 02 : 9
Enter the value of position 10 : 4
Enter the value of position 11 : 69
Enter the value of position 12 : 15
Enter the value of position 20 : 34
Enter the value of position 21 : 1
Enter the value of position 22 : 1
================== Arrays After Taking Input ==============
mat_1 is given as :
2       8       4
0       12      5
87      32      6
mat_2 is given as :
76      12      9
4       69      15
34      1       1
================== Arrays After Performing Eq2 ============
(Mat_2-3)*2
146     18      12
2       132     24
62      -4      -4
------------------------------------------------------------
BUILD SUCCESS
```

Output (For eq3):

```
Enter the elements of mat_1
Enter the value of position 00 : 1
Enter the value of position 01 : 2
Enter the value of position 02 : 4
Enter the value of position 10 : 2
Enter the value of position 11 : 6
Enter the value of position 12 : -1
Enter the value of position 20 : 56
Enter the value of position 21 : 0
Enter the value of position 22 : 4
Enter the elements of mat_2
Enter the value of position 00 : -11
Enter the value of position 01 : 3
Enter the value of position 02 : 2
Enter the value of position 10 : 67
Enter the value of position 11 : 4
Enter the value of position 12 : 36
Enter the value of position 20 : -34
Enter the value of position 21 : 5
Enter the value of position 22 : 7
================== Arrays After Taking Input ===============
mat_1 is given as :
1       2       4
2       6       -1
56      0       4
mat_2 is given as :
-11     3       2
67      4       36
-34     5       7
================== Arrays After Performing Eq3 =============
(Mat_2*5)-(Mat_1*2)
-54     15      8
335     16      183
-224    27      33
------------------------------------------------------------
BUILD SUCCESS
```

Exercise 4 (Recursion) *(prodcut.java)*

Write a recursive method to get multiply of all number from 1 up to given number. E.g. Number = 5 Result must be sum (1*2*3*4*5)

Code:

Task4 Class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;
class Task4 {
    int i = 1;
    int product = 1;
    int number = 0;
    public void input(){
        System.out.print("Enter an integar : ");
        Scanner input = new Scanner(System.in);
         number = input.nextInt();
    }
    public int computeProduct() {
        if (number != 0) {
            if (i <= number) {
                product = product * i;
                i = i + 1;
                computeProduct();
            }
        }
        return product;
    }
}
```

Application class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

public class Mavenproject2 {

    public static void main(String[] args) {
        Task4 obj = new Task4();
        obj.input();
        System.out.println("The Final product is : "+obj.computeProduct());
}
}
```

Output:

```
Enter an integar : 12
The Final product is : 479001600
------------------------------------
BUILD SUCCESS
------------------------------------
```

Exercise 5 (Recursion) *(NumberSum.java)*

Write a recursive function that takes two int as arguments and compute the sum of all number between provided two positive integers for example

If 1 and 20 are passed to the function answer should be 210.

Code:

Task5 Class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;
class Task5{
    int a,b;
    int i = 0,sum = 0;
    void input(){
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a positive integar : ");
        a = input.nextInt();
        System.out.print("Enter another positive integar : ");
        b = input.nextInt();
    }

    int computeSum(){
        if(a>=0 && b>=0 && a<=b){
            if(i<=b){
                sum+=a;
                a+=1;
                computeSum();
            }
        }

        return sum;
    }
}
```

Application class:

```
package com.mycompany.mavenproject2;
import java.util.Scanner;

public class Mavenproject2 {

    public static void main(String[] args) {
```

```
        Task5 obj = new Task5();
         obj.input();
         System.out.print("The sum of all positive numbers between "+obj.a+" and "+obj.b+" is : "+obj.computeSum());
}
}
```

Output:

```
Enter a positive integar : 1
Enter another positive integar : 20
The sum of all positive numbers between 1 and 20 is : 210
------------------------------------------------------------
BUILD SUCCESS
```

OOP Lab-05 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Exercise 1 *(Computer.java)*

Write a program using the concepts of a default constructor. Consider a computer system whose name, type, processor specification, ram, hard disk drives, mother board, optical drive etc, in a constructor, desired values are entered by the user in a get method (that takes information from the user) and the displays the inputted information via display method. The user shall be asked to change any of the provided information

Code:

(ComputerSystem Class)

```
package com.mycompany.mavenproject2;

public class ComputerSystem {
    private String name,type,processor,motherBoard,opticalDrive;
    private int ram,hardDisk;

    //setters
    public void setName(String name){
        this.name = name;
    }
     public void setType(String type){
        this.type = type;
    }
      public void setProcessor(String processor){
        this.processor = processor;
    }
       public void setMotherBoard(String motherBoard){
        this.motherBoard = motherBoard;
    }
        public void setOpticalDrive(String opticalDrive){
        this.opticalDrive = opticalDrive;
    }
         public void setRAM(int ram){
        this.ram = ram;
    }
          public void sethardDisk(int hardDisk){
```

```
        this.hardDisk = hardDisk;
    }

    //getters
    public String getName(){
        return name;
    }
     public String getType(){
        return type;
    }
      public String getProcessor(){
        return processor;
    }
       public String getMotherBoard(){
        return motherBoard;
    }
        public String getOpticalDrive(){
        return opticalDrive;
    }
         public int getRAM(){
        return ram;
    }
          public int getHardDisk(){
        return hardDisk;
    }

     //constructor
      ComputerSystem(){
          name = "Lenovo";
          type = "Desktop";
          processor = "i7 4th gen";
          motherBoard = "MSI";
          opticalDrive = "GigaByte";
          ram = 12;
          hardDisk = 1000;
      }
      //printAll
      void printSpecs(){
          System.out.println("Name : "+name);
          System.out.println("Type : "+type);
```

```
        System.out.println("Processor : "+processor);
        System.out.println("Motherboard : "+motherBoard);
        System.out.println("OpticalDrive : "+opticalDrive);
        System.out.println("RAM : "+ram+" GB");
        System.out.println("Hard drive : "+hardDisk+" GB");
    }

}
```

Main Class

```
package com.mycompany.mavenproject2;
public class Mavenproject2 {

   public static void main(String[] args) {
      //Task---01
            ComputerSystem obj1 = new ComputerSystem();
            obj1.printSpecs();
        ComputerSystem obj = new ComputerSystem();
        obj.setName("DELL");
        obj.setType("Laptop");
        obj.setProcessor("i5 12th gen");
        obj.setOpticalDrive("HP");
        obj.setRAM(16);
        obj.sethardDisk(20000);
        System.out.println("Name : "+obj.getName());
        System.out.println("Name : "+obj.getType());

        System.out.println("Name : "+obj.getProcessor());

         System.out.println("Name : "+obj.getMotherBoard());

         System.out.println("Name : "+obj.getOpticalDrive());
          System.out.println("Name : "+obj.getRAM());
          System.out.println("Name : "+obj.getHardDisk());
)
```

Output:

By using Default Constructor

```
Name : Lenovo
Type : Desktop
Processor : i7 4th gen
Motherboard : MSI
OpticalDrive : GigaByte
RAM : 12 GB
Hard drive : 1000 GB
--------------------------
BUILD SUCCESS
```

By using Getters and Setters

```
Name : DELL
Name : Laptop
Name : i5 12th gen
Name : MSI
Name : HP
Name : 16
Name : 20000
---------------------
BUILD SUCCESS
---------------------
```

Exercise 2 Time.java)

Create a program that determines the current time and date. The program must incorporate several Methods out of which three Methods should be constructors, the first one shall be a default constructor, the second and third one shall be an overloaded constructors, from which one method deals with YEAR, MONTH AND DAY, whereas the second method deals with YEAR, MONTH,DAY,HOUR,MINUTES AND SECONDS. The other methods may include the set methods and get methods which sets and gets the described values.

Code:

Task2 Class

```
package com.mycompany.mavenproject2;

import java.time.LocalDateTime;

public class Task2 {
LocalDateTime now = LocalDateTime.now();

    private int year, day, hour, minutes, seconds;
    private String month;
    private final String[] months = {"January","February","March","April","May","June","July",
"August","September","October","November","December"};

//    Constructors
    Task2() {
        year = now.getYear();
        day = now.getDayOfMonth();
        month = now.getMonth().toString();
        month = month.substring(0, 1) + month.substring(1).toLowerCase();
        hour = now.getHour();
        minutes = now.getMinute();
        seconds = now.getSecond();
    }
        // parameterized constructor for yr, month, day
    Task2(int year, int month, int day, int hour, int minutes, int seconds) {
         this.year = year;
        this.month = months[month - 1];
        this.day = day;
        this.hour = hour;
        this.minutes = minutes;
        this.seconds = seconds;
    } // parameterized function for yr, month, day, hour, minute, second

    // Getters
    public int getYear() {
```

```
        return year;
    }
    public String getMonth() {
        return month;
    }
    public int getDay() {
        return day;
    }
    public int getHour() {
        return hour;
    }
    public int getMinutes() {
        return minutes;
    }
    public int getSeconds() {
        return seconds;
    }

    // Setters
    public void setYear(int year) {
        this.year = year;
    }
    public void setMonth(int month) {
        this.month = months[month - 1];
    }
    public void setDay(int day) {
        this.day = day;
    }
    public void setHour(int hour) {
        this.hour = hour;
    }
    public void setMinute(int minutes) {
        this.minutes = minutes;
    }
    public void setSecond(int seconds) {
        this.seconds = seconds;
    }

    public void print() {
        System.out.println(month + " " + day + " " + year + " " + hour + ":" + minutes + ":" + seconds);
```

```
    }
}
```

Application Class

```
package com.mycompany.mavenproject2;
public class Mavenproject2 {
    public static void main(String[] args) {
        //Values can be taken from the user and can be validated by conditioning but giving hardcoded
        Task2 now1 = new Task2(); // default
        now1.print(); // Whole date

        Task2 now2 = new Task2(); // 3 parameterized
        now2.print(); // Whole date

        Task2 now3 = new Task2(2003, 10, 21, 11, 30, 55); // 6 parametized

//      Same setting can be done with year, hour, ,day, hour, minutes, seconds
        String month = now3.getMonth(); // getting
        System.out.println("The current month is " + month);
        Scanner input = new Scanner(System.in);
        System.out.print("Please enter a number that indicates month: "); // setting
        int monthInput = input.nextInt();
        now3.setMonth(monthInput);
        month = now3.getMonth(); // getting again
        System.out.println("Now the month is " + month);

        System.out.println("And the whole date looks like:");
        now3.print(); // Whole date
}
```

Output:

```
October 21 2023 7:24:58
October 21 2023 7:24:58
The current month is October
Please enter a number that indicates month: 9
Now the month is September
And the whole date looks like:
September 21 2003 11:30:55
------------------------------------------------------------
BUILD SUCCESS
```

Exercise 3 (Book.java)

Write a Java class Book with following features:

- Instance variables :
  - **title** for the title of book of type String.
  - **author** for the author's name of type String.
  - **price** for the book price of type double.
- Constructor:
  - **public Book (String title, Author name, double price)**: A constructor with parameters, it creates the Author object by setting the the fields to the passed values.
- Instance methods:
  - **public void setTitle(String title)**: Used to set the title of book.
  - **public void setAuthor(String author)**: Used to set the name of author of book.
  - **public void setPrice(double price)**: Used to set the price of book.
  - **public double getTitle()**: This method returns the title of book.
  - **public double getAuthor()**: This method returns the author's name of book.
  - **public String toString()**: This method printed out book's details to the screen

Write a separate class **BookDemo** with a main() method creates a Book titled "Developing Java Software" with authors Russel Winderand price 79.75. Prints the Book's string representation to standard output (using System.out.println).

Code:

Book Class

```
package com.mycompany.mavenproject2;

class Book{
    private String title, author;
    private double price;

    //setters
    public void setTitle(String title){
        this.title = title;
    }
    public void setAuthor(String author){
        this.author = author;
    }
    public void setPrice(double price){
```

```
        this.price = price;
    }

    //getters
    public String getTitle(){
        return title;
    }
        public String getAuthor(){
        return author;
    }
            public double getPrice(){
        return price;
    }
            //constructor
            Book(){
                title = "Developing Java Software";
                author = "Russel Winderand";
                price = 79.75;
            }

    @Override
            public String toString(){
                return ("Name : "+title+"\n"+"Author : "+author+"\n"+"Price : "+price);
            }
}
```

Application Class

```
package com.mycompany.mavenproject2;
public class Mavenproject2 {
    public static void main(String[] args) {
        Book book1 = new Book();
        System.out.println("=============== By using Getters and Setters ===============");
        book1.setTitle("Fundamentals of Physics");
        book1.setAuthor("Halliday Resnick");
        book1.setPrice(100.32);
        System.out.println("Title : "+book1.getTitle());
        System.out.println("Title : "+book1.getAuthor());
        System.out.println("Title : "+book1.getPrice());

        System.out.println("=============== By using Default constructor ===============");
        Book book2 = new Book();
        System.out.println(book2);
    }
}
```

Output:

```
=============== By using Getters and Setters ===============
Title : Fundamentals of Physics
Title : Halliday Resnick
Title : 100.32
=============== By using Default constructor ===============
Name : Developing Java Software
Author : Russel Winderand
Price : 79.75
------------------------------------------------------------------
BUILD SUCCESS
```

OOP Lab-06 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

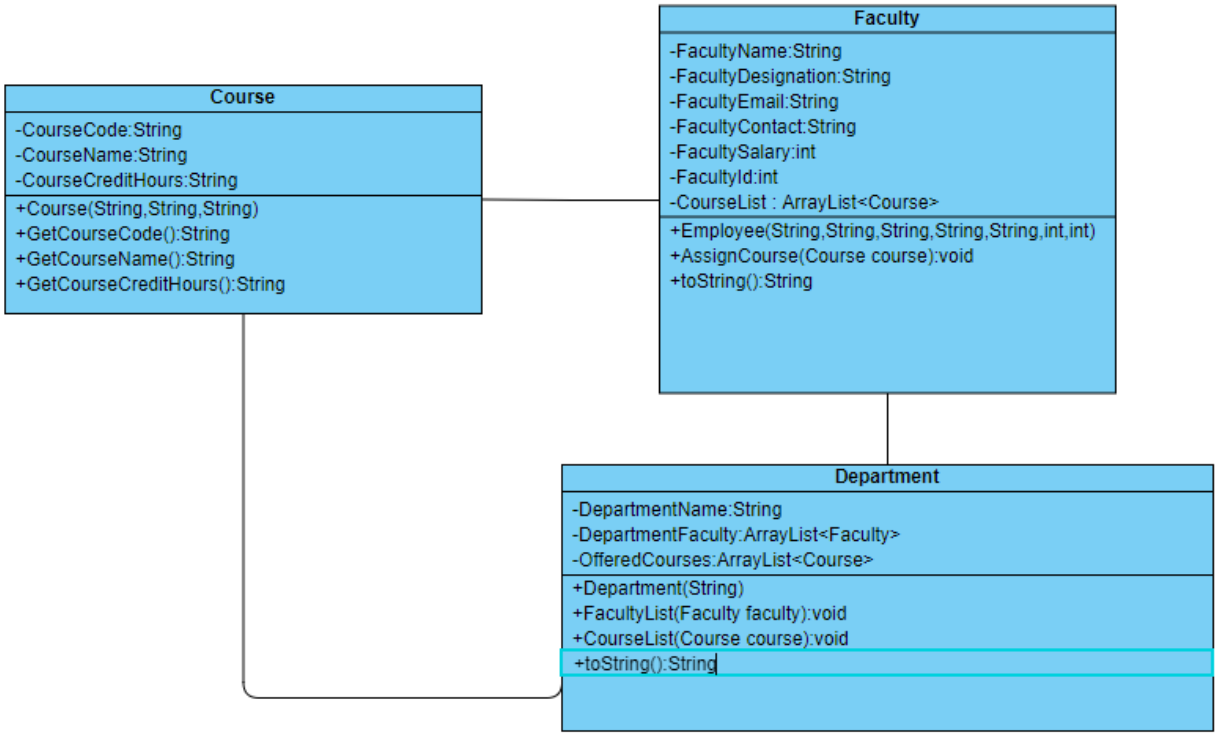
Designing and implementing Java programs that deal with:

1. **Association and**
2. **Aggregation**

Exercises

Exercise 1(a)

Create a jav program based on the guivrn UML diagram and implement the relation between the classes as shown in the diagram



Exercise 1(b)

Create Driver class named as **Association_aggregation_1**. Create proper Objects of all classes as follows

- 2 Objects of Course class
- 3 Objects of Faculty class
- 3 Objects of Departement Class

And properly display all the information

```
package com.mycompany.mavenproject2;
import java.util.ArrayList;
class Course{
    private String courseCode,courseName,courseCreditHours;

    //constructor
    Course(String courseCode,String courseName, String courseCreditHours){
        this.courseCode = courseCode;
        this.courseName = courseName;
        this.courseCreditHours = courseCreditHours;
    }
    //methods
    public String getCourseCode(){
        return courseCode;
    }
    public String getCourseName(){
        return courseName;
    }
    public String getCourseCreditHours(){
        return courseCreditHours;
    }

}

class Faculty{
    private String facultyName,facultyDesignation,facultyEmail,facultyContact;
    private int facultySalary, facultyId;
    private ArrayList<Course> courseList = new ArrayList<Course>();

    //constructor
    Faculty(String facultyName, String facultyDesignation, String facultyEmail, String
facultyContact, int facultySalary, int facultyId){
        this.facultyName = facultyName;
        this.facultyDesignation = facultyDesignation;
        this.facultyEmail = facultyEmail;
        this.facultyContact = facultyContact;
        this.facultySalary = facultySalary;
        this.facultyId = facultyId;
    }
    //methods
    public void assignCourse(Course course){
        courseList.add(course);
    }
    public String getFacultyName(){
        return facultyName;
    }
    public String getFacultyDesgination(){
        return facultyDesignation;
    }
    public String getFacultyEmail(){
```

```
        return facultyEmail;
    }
    public String getFacultyContact(){
        return facultyContact;
    }
    public int getFacultySalary(){
        return facultySalary;
    }
    public int getFacultyId(){
        return facultyId;
    }
    public void printAll(){
        System.out.println("Faculty Name : "+facultyName+"\n"+
                "Faculty Designation : "+facultyDesignation + "\n"+
                "Faculty Email : "+ facultyEmail+ "\n"+
                "Faculty Contact : "+facultyContact+"\n"+
                "Faculty Salary : "+facultySalary+"\n"+
                "Faculty Id : "+facultyId+"\n=============== Faculty Courses ===============
\n");
                for(int i = 0;i<courseList.size();i++){
                    System.out.println( "* Course "+(i+1)+" : ");
                    System.out.println("Course code : "+courseList.get(i).getCourseCode());
                    System.out.println("CourseName : "+courseList.get(i).getCourseName());
                    System.out.println("CourseCreditHours : "+courseList.get(i).getCourseCreditHours());

                }
    }
}

class Department{
    private String departmentName;
    private ArrayList<Faculty> departmentFaculty = new ArrayList<Faculty>();
    private ArrayList<Course> offeredCourses = new ArrayList<Course>();

    //methods
    Department(String departmentName){
        this.departmentName = departmentName;
    }

    public void facultyList(Faculty faculty){ //assigning faculty
        departmentFaculty.add(faculty);
    }
    public void courseList(Course course){
        offeredCourses.add(course);
    }
    public void printAll(){
        System.out.println("Department Name : "+departmentName);
        System.out.println("============= Faculty List =============");
        for (int i = 0; i <departmentFaculty.size() ; i++) {
            System.out.println("* Faculty "+(i+1));
            System.out.println("Faculty Name : "+departmentFaculty.get(i).getFacultyName()+"\n"+
                "Faculty Designation : "+departmentFaculty.get(i).getFacultyDesgination()+ "\n"+
```

```
            "Faculty Email : "+ departmentFaculty.get(i).getFacultyEmail()+ "\n"+
            "Faculty Contact : "+departmentFaculty.get(i).getFacultyContact()+"\n"+
            "Faculty Salary : "+departmentFaculty.get(i).getFacultySalary()+"\n"+
            "Faculty Id : "+departmentFaculty.get(i).getFacultyId());
        }
        System.out.println("============= Course List =============");

        for (int i = 0; i <offeredCourses.size() ; i++) {
                        System.out.println("* Course "+(i+1));
                System.out.println("Course code : "+offeredCourses.get(i).getCourseCode());
                System.out.println("CourseName : "+offeredCourses.get(i).getCourseName());
                System.out.println("CourseCreditHours :
"+offeredCourses.get(i).getCourseCreditHours());
        }
    }
}
public class Mavenproject2 {

    public static void main(String[] args) {
        //Two objects of course
        Course c1 = new Course("0000","oop","12");
        Course c2 = new Course("001","CP","18");
        //Three objs of Faculty
        Faculty f1 = new Faculty("Syed
Raza","Lecturer","asyedraza85632@gmail.com","121212121",24000,1234);
        Faculty f2 = new Faculty("Aimen","TA","aimen@gmail.com","232323232",10000,1248);
        Faculty f3 = new
Faculty("Muskan","Lecturer","muskan123@gmail.com","98989898",15000,1233);
        //Three objs of Department
        Department d1 = new Department("Computer Science");
        Department d2 = new Department("IPP");
        Department d3 = new Department("Maritime");

        //Assigning faculty to depart
        d1.facultyList(f1);
        d1.facultyList(f2);
        d1.facultyList(f3);
        //Assigning Courses to depart
        d1.courseList(c1);
        d1.courseList(c2);

        //assigning courses to faculty
        f1.assignCourse(c1);
        f1.assignCourse(c2);
        f1.printAll();
       //printing faculty and courses in a depart
        d1.printAll();

    }
}
```

Output:

Creating two course objects and assigning them to a faculty obj

```
Faculty Name : Syed Raza
Faculty Designation : Lecturer
Faculty Email : asyedraza85632@gmail.com
Faculty Contact : 121212121
Faculty Salary : 24000
Faculty Id : 1234
=============== Faculty Courses ===============

* Course 1 :
Course code : 0000
CourseName : oop
CourseCreditHours : 12
* Course 2 :
Course code : 001
CourseName : CP
CourseCreditHours : 18
```

Creating Two objects of courses and three objects of faculty and assigning them to an obj of department, then printing all the details

```
Department Name : Computer Science
============= Faculty List =============
*. Faculty 1
Faculty Name : Syed Raza
Faculty Designation : Lecturer
Faculty Email : asyedraza85632@gmail.com
Faculty Contact : 121212121
Faculty Salary : 24000
Faculty Id : 1234
* Faculty 2
Faculty Name : Aimen
Faculty Designation : TA
Faculty Email : aimen@gmail.com
Faculty Contact : 232323232
Faculty Salary : 10000
Faculty Id : 1248
* Faculty 3
Faculty Name : Muskan
Faculty Designation : Lecturer
Faculty Email : muskan123@gmail.com
Faculty Contact : 98989898
Faculty Salary : 15000
Faculty Id : 1233
============= Course List =============
* Course 1
Course code : 0000
CourseName : oop
CourseCreditHours : 12
* Course 2
Course code : 001
CourseName : CP
CourseCreditHours : 18
```

OOP Lab-07 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

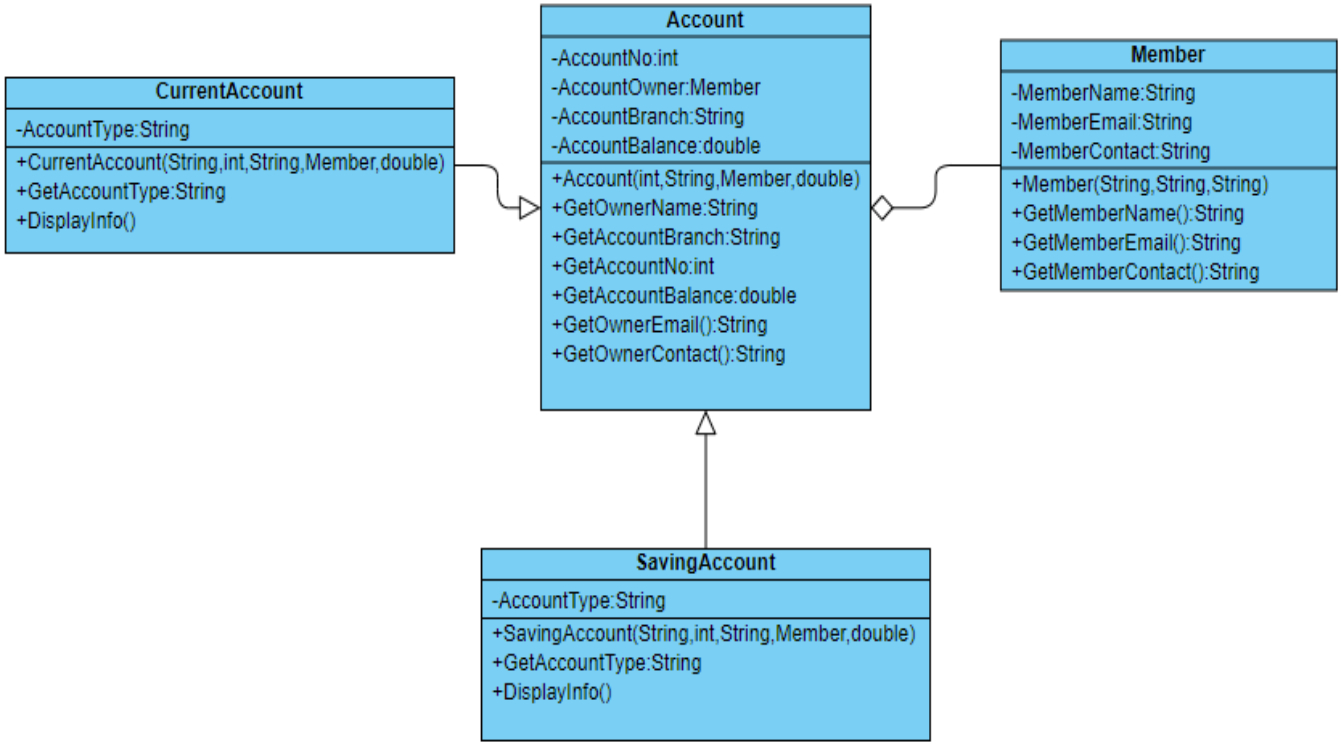
In this lab, the following topics will be covered:

1. **Inheritacen (in Java)**
2. **Exercise for practice**

Exercises

Exercise 1

The UML diagram of a Bank Account system that contains the following classes.Write Java code for these classes.



Imlement all classes given in UML diagram.
Create Driver class named as **Inheritance_2**. Create proper Objects of all classes as follows

- Create three objects of **Member** class. Two of which have **current account** and one has **saving account.** Properly display all the information.

Code:

```
package com.mycompany.mavenproject4;

class Account{
   private int accountNo;
   private Member accountOwner;
   private String accountBranch;
   private double accountBalance;
   //methods
   public String getOwnerEmail(){
      return accountOwner.getMemberEmail();
   }
    public String getOwnerContact(){
      return accountOwner.getMemberContact();
   }
    //getters
   public int getAccountNo() {
      return accountNo;
   }

   public String getAccountBranch() {
      return accountBranch;
   }

   public double getAccountBalance() {
      return accountBalance;
   }
   //constructors
   public Account(int accountNo, Member accountOwner, String accountBranch, double
accountBalance) {
      this.accountNo = accountNo;
      this.accountOwner = accountOwner;
      this.accountBranch = accountBranch;
      this.accountBalance = accountBalance;
   }
```

```
    public Account() {
    }

    @Override
    public String toString() {
        return "Account{" + "accountNo=" + accountNo +
                ", accountOwnerName =" + accountOwner.getMemberName() + ", accountBranch="
+"AccountOwner Email : "+accountOwner.getMemberEmail()+
                "Account Branch : "+accountBranch + ", accountBalance=" + accountBalance + '}';
    }

}
class Member{
private String memberName,memberEmail,memberContact;
    //getters
    public String getMemberName() {
        return memberName;
    }

    public String getMemberEmail() {
        return memberEmail;
    }

    public String getMemberContact() {
        return memberContact;
    }
//constructors
    public Member(String memberName, String memberEmail, String memberContact) {
        this.memberName = memberName;
        this.memberEmail = memberEmail;
        this.memberContact = memberContact;
    }

    public Member() {
    }
```

```
}
class CurrentAccount extends Account{
    private String accountType;

    public String getAccountType() {
        return accountType;
    }

    public CurrentAccount(String accountType) {
        this.accountType = accountType;
    }

    public CurrentAccount() {
    }

    public CurrentAccount(String accountType, int accountNo, Member accountOwner, String
accountBranch, double accountBalance) {
        super(accountNo, accountOwner, accountBranch, accountBalance);
        this.accountType = accountType;
    }

    @Override
    public String toString() {
        return super.toString()+"currentAccount{" + "accountType=" + accountType + '}';
    }




}
class SavingAccount extends Account{
    private String accountType;

    public SavingAccount() {
    }

    public SavingAccount(String accountType, int accountNo, Member accountOwner, String
accountBranch, double accountBalance) {
        super(accountNo, accountOwner, accountBranch, accountBalance);
```

```
        this.accountType = accountType;
    }

    public SavingAccount(String accountType) {
        this.accountType = accountType;
    }

    public String getAccountType() {
        return accountType;
    }

    @Override
    public String toString() {
        return super.toString()+"savingAccount{" + "accountType=" + accountType + '}';
    }

}
public class Mavenproject4 {

    public static void main(String[] args) {
        //Creting 3 members
        Member member1 = new Member("Raza","asyedraza85632@gmail.com","03121218932");
        Member member2 = new Member("Muskan","muskan123@gmail.com","03242423176");
        Member member3 = new Member("Aimen","aimen134@gmail.com","031516183873");
        //assigning 2 members to currentAccount
        CurrentAccount acc1 = new CurrentAccount("Bachat",001,member1,"Malir",12000.00);
        CurrentAccount acc2 = new CurrentAccount("Bachat",002,member2,"Malir",15000.500);
        //assigning 1 member to savingsAccount
        SavingAccount acc3 = new SavingAccount("Mega Bachat",003,member3,"Malir",42000.500);
        System.out.println(acc1);
        System.out.println(acc2);
        System.out.println(acc3);
    }
}
```

Output:

```
--- exec:3.1.0:exec (default-cli) @ mavenproject4 ---
Account{accountNo=1, accountOwnerName =Raza, accountBranch=AccountOwner Email : asyedraza85632@gmail.comAccount Branch : Malir, accountBalance=12000.0}currentAccount{accountType=Bachat}
Account{accountNo=2, accountOwnerName =Muskan, accountBranch=AccountOwner Email : muskan123@gmail.comAccount Branch : Malir, accountBalance=15000.5}currentAccount{accountType=Bachat}
Account{accountNo=3, accountOwnerName =Aimen, accountBranch=AccountOwner Email : aimen134@gmail.comAccount Branch : Malir, accountBalance=42000.5}savingAccount{accountType=Mega Bachat}
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.725 s
Finished at: 2023-11-02T23:38:17+05:00
------------------------------------------------------------------------
```

OOP LAB 9
NetBeans GUI with Class Templates

Name : Syed Muhammad Raza Ali
Enrollment : 02-134231-028

Exercise 1:

Create a GUI based application system that calculates the newton's second law of motion i.e., F=ma where,

F=Force in Newton

m= mass in grams

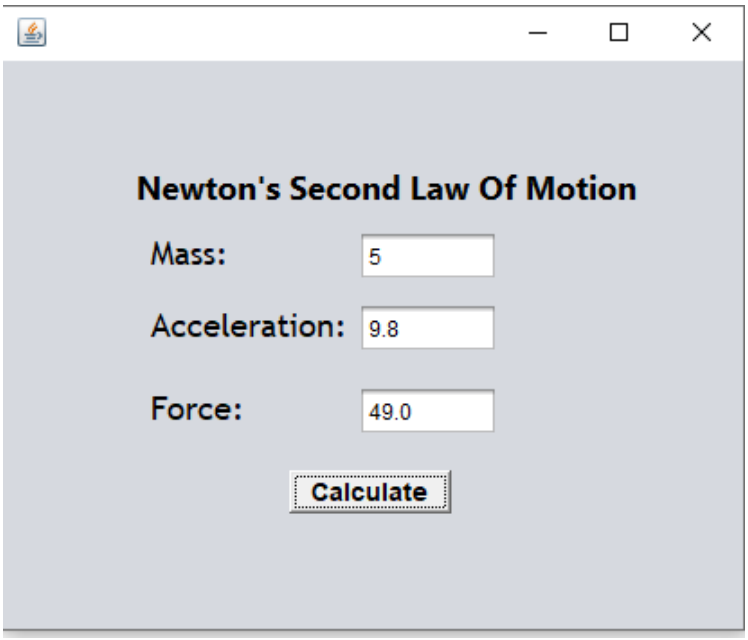
a= acceleration (a=9.8m/s^2 for free falling bodies)

use proper alignment and design suitable to the application being created.

SOURCE CODE:

```
private void calculateActionPerformed(java.awt.event.ActionEvent evt) {

    // TODO add your handling code here:

    int mass = Integer.parseInt(Mass.getText());

    double acc = Double.parseDouble(Acceleration.getText());

    double r = mass * acc;

    Force.setText(String.valueOf(r));

  }
```

OUTPUT:



Exercise 2:

Create a login system interface based on your own choice of environment for example, cafeteria, cyber security site or etc. Make sure you have your own unique scenario and apply proper designing chose appropriate color palette and images. System must have proper labels, text boxes, buttons, and other necessary controls.

SOURCE CODE IN CLASS:

```
package lab9;

public class LAB9 {

  String name,password;
```

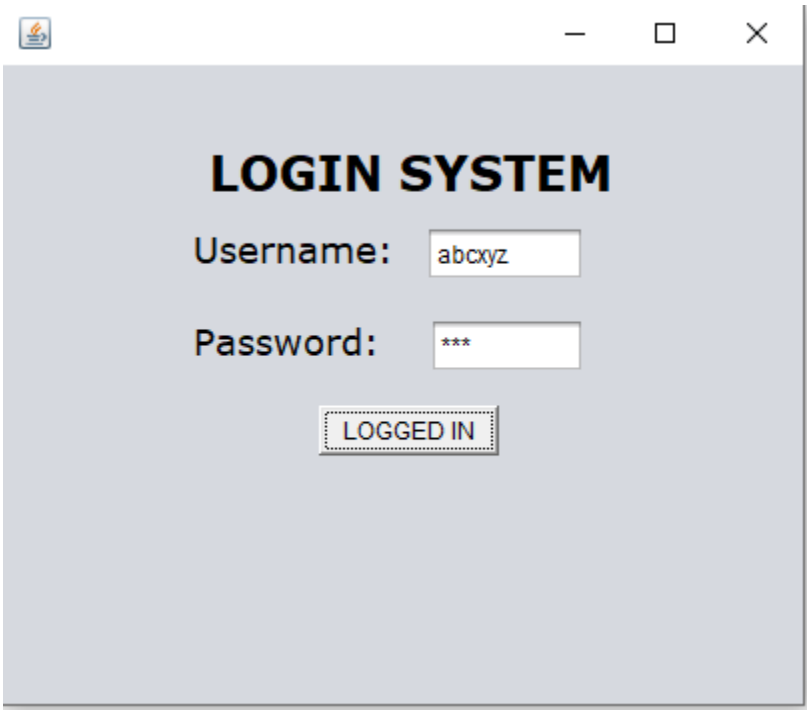
```
    public LAB9(String name, String password) {
        this.name = name;
        this.password = password;
    }

    public String display()
    {
     if (name.contains("abcxyz")&& password.contains("123"))
     {
         return "LOGGED IN";
     }
     else
     {
         return "LOGIN FAIL!!";
     }
    }
```

}jFrame SOURCE CODE:

```
private void clickActionPerformed(java.awt.event.ActionEvent evt) {

        String name= username.getText();
        String pass =password.getName();

        LAB9 obj=new LAB9(name,pass);

String login=obj.display();
```

```
    click.setLabel(String.valueOf(login));
  }
```

OUTPUT:



OOP Lab-10 Task

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Lab10: Abstract Class and Interface

To gain experience with:

1. **Abstract Classes and their purpose**
2. **Interfaces and their purpose**
3. **Exercise for practice**

1. Abstract Classes (in Java)

An abstract class usually defines a concept. Abstract classes have no implementation and they are only used as generic superclasses. An abstract class may contain:

- An abstract method (no implementation)
- A non-abstract method (or concrete method)
- Instance variables

Note that it is not necessary for an abstract class to have an abstract method. An abstract class is always preceded by the keyword **abstract.** As an example we provide the following hierarchy of an abstract class `Shape` and its non-abstract children:

In this lab we'll walk through the following class hierarchy:

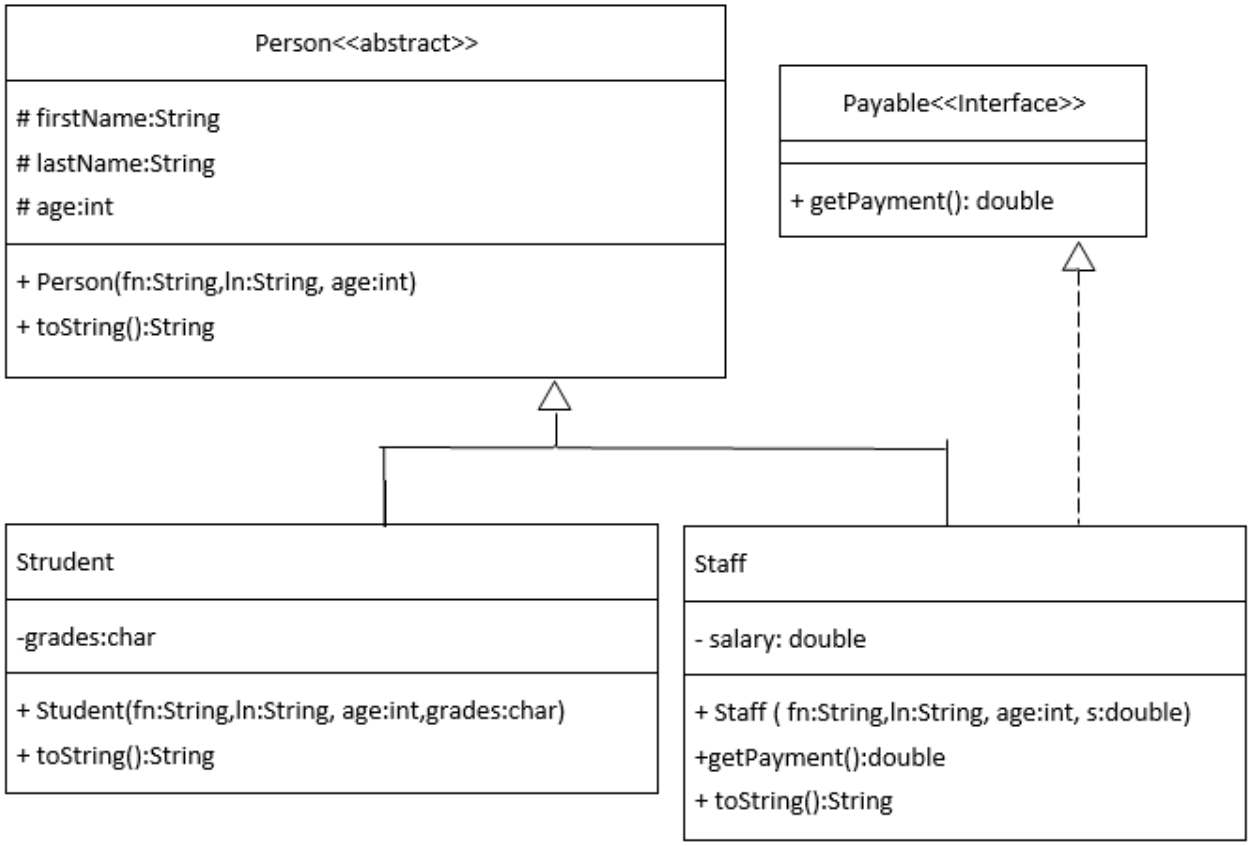
2. Interfaces (in Java)

Interfaces in java are a solution to the multiple-inheritance problem. In instances where a class exhibits a behavior of more than one class, interfaces are used.

Interfaces have only abstract methods and final (constant) variables. All methods in an interface are by default public and abstract.

Take Home Assignment

Consider the following UML class diagram:



- Consider an **abstract class Person** with three private attributes FirstName,LastName and Age. It has one abstract method

  String toString();
- Also, consider an **interface Payable** that contains only one method

  double getPayment().
- Class **Staff** has an additional attribute that is **salary**, which represents the monthly salary.
- Class **Student**has an additional attribute that is **grades**, which represents the grade character like A, B, C, D or F.
- In addition, the class **Staff** implements the interface **Payable**.
- The payment of the staff must return its **annual salary** (salary *12).
- The toString method of the each class must return all attributes of the class in text format. For a student, it returns the following string
  - Student: *FirstName*, *LastName*, *Age*, Grade
- For the Staff, it must return the following string
  - Staff: *FirstName, LastName, Age, Salary*.

Write the application class and create 3 objects of Student and 3 objects of Staff class and print the details.

Code:

```
package com.mycompany.interfacesapplication;
```

Person Class

```
abstract class Person{
    //data members
    protected String firstName;
    protected String lastName;
    protected int age;
    //Constructor
    public Person(String firstName, String lastName, int age) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.age = age;
    }
    //method
    abstract void display();
}
```

Student Class

```
class Student extends Person{
    private char grade;
    //getter

    public char getGrade() {
        return grade;
    }
    //setter

    public void setGrade(char grade) {
        this.grade = grade;
    }
    //constructor

    public Student(String firstName, String lastName, int age, char grade) {
       super(firstName,lastName,age);
        this.grade = grade;
    }
    //method
    @Override
```

```
    void display() {
        System.out.println("======== Student Info ======== \nFirst Name :
"+firstName+"\n"+"Last Name : "+lastName+"\n"+"Age : "+age
        +"\nGrade : "+grade);
    }
}
```

Staff Class

```
class Staff extends Person implements Payable{
    private double salary;
    //constructor
    public Staff(double salary, String firstName, String lastName, int age) {
        super(firstName, lastName, age);
        this.salary = salary;
    }
    //method
    @Override
    void display() {
        System.out.println("======== Staff Info ======== \nFirst Name : "+firstName+"\n"+"Last
Name : "+lastName+"\n"+"Age : "+age
        +"\nSalary : "+salary);
    }
    public double getPayment(){
        return this.salary;
    }
}
```

Interface

```
interface Payable{
    double getPayment();
}
public class InterfacesApplication {

    public static void main(String[] args) {
       //three objs of Student
        Student student1 = new Student("Raza","Ali",19,'A');
        Student student2 = new Student("Muskan","Khan",19,'B');
        Student student3 = new Student("Aimen","Abdullah",20,'C');
        //three objs of Staff
        Staff member1 = new Staff(20000,"Ali","Ahmed",25);
        Staff member2 = new Staff(30000,"Ahmed","Raza",23);
        Staff member3 = new Staff(17000,"Rameel","Ahmmed",26);
```

```
        //printing the details of Students
        student1.display();
        student2.display();
        student3.display();
        //printing the details of Staff
        member1.display();
        member2.display();
        member3.display();
    }
}
```

Output:

```
======== Student Info ========
First Name : Raza
Last Name : Ali
Age : 19
Grade : A
======== Student Info ========
First Name : Muskan
Last Name : Khan
Age : 19
Grade : B
======== Student Info ========
First Name : Aimen
Last Name : Abdullah
Age : 20
Grade : C
======== Staff Info ========
First Name : Ali
Last Name : Ahmed
Age : 25
Salary : 20000.0
======== Staff Info ========
First Name : Ahmed
Last Name : Raza
Age : 23
Salary : 30000.0
======== Staff Info ========
First Name : Rameel
Last Name : Ahmmed
Age : 26
Salary : 17000.0
------------------------------------
BUILD SUCCESS
------------------------------------
```

OOP Lab-11 Tasks

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Lab11: Exception Handling

In Lab following topics will be covered:

1. Exceptions
2. Types of Exceptions

Exercises

Exercise

Create a program that has class of Mathematical Operations on Matrices and Apply all Exception Handling that can occur e.g. IndexOutOfRange or StackOverflow etc.

Program must perform

1. Addition of two matrices (A+B)
2. Subtraction of two matrices (A-B)
3. Scalar Multiplication with Matrix (x*A)
4. Consider A and B as matrices and x as any scalar value other than 0

Code:

Matrices Class

```
package com.mycompany.exceptionhandling;

import java.util.InputMismatchException;
import java.util.Scanner;

public class Matrices{
    protected int[][] A = new int[3][3];
    protected int[][] B = new int[3][3];
    Scanner sc = new Scanner(System.in);
    //methods
    public void takeInputInA(){
        try{
            System.out.println("===== Input In A =====");
            for(int i = 0;i<A.length;i++){
                for(int j = 0;j<A.length;j++){
                    System.out.println("Enter the value of position "+i+j+" : ");
                    A[i][j] = sc.nextInt();
                }
            }
        }
        catch(IndexOutOfBoundsException | NullPointerException | InputMismatchException e){
            System.out.println(e.getMessage());
        }

    }
    public void takeInputInB(){
        try{
            System.out.println("===== Input In B =====");
            for(int i = 0;i<B.length;i++){
                for(int j = 0;j<B.length;j++){
                    System.out.println("Enter the value of position "+i+j+" : ");
```

```
                B[i][j] = sc.nextInt();
            }
        }
    }
    catch(IndexOutOfBoundsException | NullPointerException | InputMismatchException e){
        System.out.println(e.getMessage());
    }
  }

}
```

Equation1 Class

```
package com.mycompany.exceptionhandling;

import java.util.InputMismatchException;

public class Equation1 extends Matrices{
  private int[][] result = new int[3][3];

  public Equation1() {
  }

  public void calculateSum(){
    try{
        for(int i = 0;i<result.length;i++){
        for(int j = 0;j<result.length;j++){
            result[i][j] = A[i][j] + B[i][j];
        }
    }
                System.out.println("===== A+B =====");
        for(int i = 0;i<result.length;i++){
        for(int j = 0;j<result.length;j++){
            System.out.print(result[i][j]+"   ");
        }
        System.out.println("\n");
```

```
        }
        }
         catch(IndexOutOfBoundsException | NullPointerException | InputMismatchException e){
            System.out.println(e.getMessage());
        }
    }
}
```

Equation2 Class

```
package com.mycompany.exceptionhandling;

import java.util.InputMismatchException;

public class Equation2 extends Matrices{
        private int[][] result = new int[3][3];

    public Equation2() {
    }
 public void calculateDifference(){
        try{
            for(int i = 0;i<result.length;i++){
            for(int j = 0;j<result.length;j++){
                result[i][j] = A[i][j] - B[i][j];
            }
        }
                    System.out.println("===== A-B =====");
            for(int i = 0;i<result.length;i++){
            for(int j = 0;j<result.length;j++){
                System.out.print(result[i][j]+"   ");
            }
            System.out.println("\n");
        }
        }
         catch(IndexOutOfBoundsException | NullPointerException | InputMismatchException e){
            System.out.println(e.getMessage());
        }
```

```
    }
}
```

Equation3 Class

```
package com.mycompany.exceptionhandling;

import java.util.InputMismatchException;



public class Equation3 extends Matrices{
    private int scalarValue;

    public Equation3() {
    }
    public void calculateProduct(){

        try{
            System.out.print("Enter a value : ");
        scalarValue = sc.nextInt();
            for(int i = 0;i<A.length;i++){
            for(int j = 0;j<A.length;j++){
                A[i][j] *=scalarValue;
            }
        }
                    System.out.println("===== A*B =====");
            for(int i = 0;i<A.length;i++){
            for(int j = 0;j<A.length;j++){
                System.out.print(A[i][j]+"   ");
            }
            System.out.println("\n");
        }
        }
         catch(IndexOutOfBoundsException | NullPointerException | InputMismatchException e){
            System.out.println(e.getMessage());
        }
    }
```

```
}
```

Equation4 Class

```
package com.mycompany.exceptionhandling;

import java.util.InputMismatchException;

public class Equation4 extends Matrices{
        private int scalarValue;
        private int[][] result = new int[3][3];

    public Equation4() {
    }
 public void calculateProduct(){
        try{

            System.out.println("Enter a value : ");
            scalarValue = sc.nextInt();
            if(scalarValue==0){
                System.out.println("Enter a non zero value....");
            }
            else{
                 for(int i = 0;i<A.length;i++){
            for(int j = 0;j<A.length;j++){
               result[i][j] =A[i][j] * B[i][j]*scalarValue ;
            }
            }

        }
                System.out.println("===== A*B*scalarValue =====");

            for(int i = 0;i<A.length;i++){
            for(int j = 0;j<A.length;j++){
                System.out.print(result[i][j]+"   ");
            }
            System.out.println("\n");
```

```
        }
        }
         catch(IndexOutOfBoundsException | NullPointerException | InputMismatchException e){
            System.out.println(e.getMessage());
        }    }}
```

Application Class

```
package com.mycompany.exceptionhandling;
public class ExceptionHandling {

    public static void main(String[] args) {
        //for equation1
          Equation1 obj = new Equation1();
          obj.takeInputInA();
          obj.takeInputInB();
          obj.calculateSum();
        //for equation2

          Equation2 obj = new Equation2();
          obj.takeInputInA();
          obj.takeInputInB();
          obj.calculateDifference();
                //for equation3

          Equation3 obj = new Equation3();
          obj.takeInputInA();
          obj.takeInputInB();
          obj.calculateProduct();
        //for equation4

          Equation4 obj = new Equation4();
          obj.takeInputInA();
          obj.takeInputInB();
          obj.calculateProduct();
    }
}
```

Output:

For Equation1

```
===== Input In A =====
Enter the value of position 00 :
12
Enter the value of position 01 :
-7
Enter the value of position 02 :
-45
Enter the value of position 10 :
34
Enter the value of position 11 :
398
Enter the value of position 12 :
58
Enter the value of position 20 :
4
Enter the value of position 21 :
7
Enter the value of position 22 :
3
===== Input In B =====
Enter the value of position 00 :
90
Enter the value of position 01 :
56
Enter the value of position 02 :
3
Enter the value of position 10 :
7
Enter the value of position 11 :
-76
Enter the value of position 12 :
-45
Enter the value of position 20 :
34
Enter the value of position 21 :
7
Enter the value of position 22 :
3
===== A+B =====
102   49   -42

41   322   13

38   14   6
```

For Equation2

```
===== Input In A =====
Enter the value of position 00 :
12
Enter the value of position 01 :
76
Enter the value of position 02 :
345
Enter the value of position 10 :
8
Enter the value of position 11 :
95
Enter the value of position 12 :
9
Enter the value of position 20 :
-7
Enter the value of position 21 :
56
Enter the value of position 22 :
-6
===== Input In B =====
Enter the value of position 00 :
45
Enter the value of position 01 :
623
Enter the value of position 02 :
98
Enter the value of position 10 :
-
null
===== A-B =====
-33   -547   247

8   95   9

-7   56   -6
```

For Equation3

```
===== Input In A =====
Enter the value of position 00 :
12
Enter the value of position 01 :
657
Enter the value of position 02 :
4
Enter the value of position 10 :
-45
Enter the value of position 11 :
-454
Enter the value of position 12 :
67
Enter the value of position 20 :
23
Enter the value of position 21 :
6
Enter the value of position 22 :
843
===== Input In B =====
Enter the value of position 00 :
-3
Enter the value of position 01 :
3
Enter the value of position 02 :
456
Enter the value of position 10 :
8
Enter the value of position 11 :
4
Enter the value of position 12 :
6
Enter the value of position 20 :
5
Enter the value of position 21 :
0
Enter the value of position 22 :
6
Enter a value : 3
===== A*B =====
36   1971   12

-135   -1362   201

69   18   2529
```

For Equation4

```
===== Input In A =====
Enter the value of position 00 :
12
Enter the value of position 01 :
67
Enter the value of position 02 :
45
Enter the value of position 10 :
8
Enter the value of position 11 :
9
Enter the value of position 12 :
45
Enter the value of position 20 :
78
Enter the value of position 21 :
56
Enter the value of position 22 :
7
===== Input In B =====
Enter the value of position 00 :
8
Enter the value of position 01 :
3
Enter the value of position 02 :
7
Enter the value of position 10 :
4
Enter the value of position 11 :
-67
Enter the value of position 12 :
34
Enter the value of position 20 :
6
Enter the value of position 21 :
6
Enter the value of position 22 :
string
null
Enter a value :
null
----------------------------------------
BUILD SUCCESS
----------------------------------------
```

OOP Lab-12 Task

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Lab12: Database Connectivity

Designing and implementing Java programs that deal with:

JDBC <-> ODBC

Exercise:

Exercise

Create an application for the student enrollment system where student is enrolled in the university. Create proper dataset to store information for a student an apply CRUD operations over the students enrollment application. Also have the appropriate designing and login system for admin.

Code:

Entering data in Dbms on Submit button

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int id, salary;
        String name;
        database = new Db();
        name = txtEname.getText();
        id = Integer.parseInt(txtEmpid.getText());
        salary = Integer.parseInt(txtSal.getText());
        String query = "Insert into emp(empno, ename, sal) values (" + id + ",'" + name + "'," +
salary + ")";

    } catch (SQLException ex) {
        Logger.getLogger(EmpForm.class.getName()).log(Level.SEVERE, null, ex);
    }
  }
```

Fetching data from Dbms on Display Button:

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            String name, id, salary;
            database = new Db();
            String query = "select * from emp";

            ResultSet reSet = database.runSelect(query);

            DefaultTableModel model = (DefaultTableModel) tableEmp.getModel();
            while (reSet.next()) {
                id = reSet.getString("empno");
                salary = reSet.getString("sal");
                name = reSet.getString("ename");
                String[] obj = {id, name, salary};
                model.addRow(obj);
            }
        } catch (SQLException ex) {
            Logger.getLogger(EmpForm.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
```

Application Class:

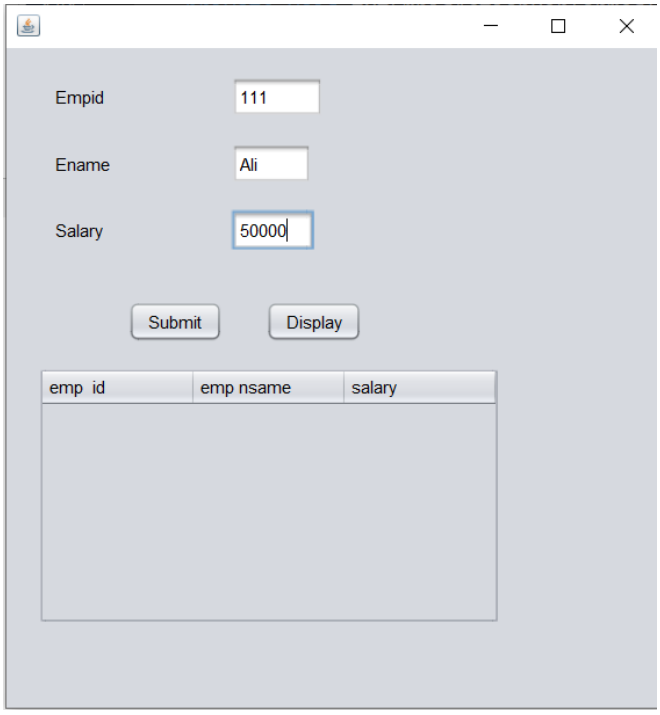
```
package Demodb;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.PreparedStatement;
```

```
import java.sql.SQLException;


public class Db {

    private static final String dLoc = "jdbc:ucanaccess://Demodb.accdb";
    private Connection con;
    private PreparedStatement pState;
    private ResultSet resSet;

    public Db() throws SQLException {
        try {
            con = DriverManager.getConnection(dLoc);
            System.out.println("Connected!");
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
    }

    public void dmlOperation(String sqlQuery) {
        try {
            pState = con.prepareStatement(sqlQuery);
            pState.executeUpdate();
            System.out.println("Update sucessfull");
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
    }
```

```
    public ResultSet runSelect(String sqlQuery) {
        try {
            pState = con.prepareStatement(sqlQuery);
            resSet = pState.executeQuery();
        } catch (SQLException e) {
            System.out.println(e.getMessage());
            return null;
        }
        return resSet;
    }
}
```

Output:

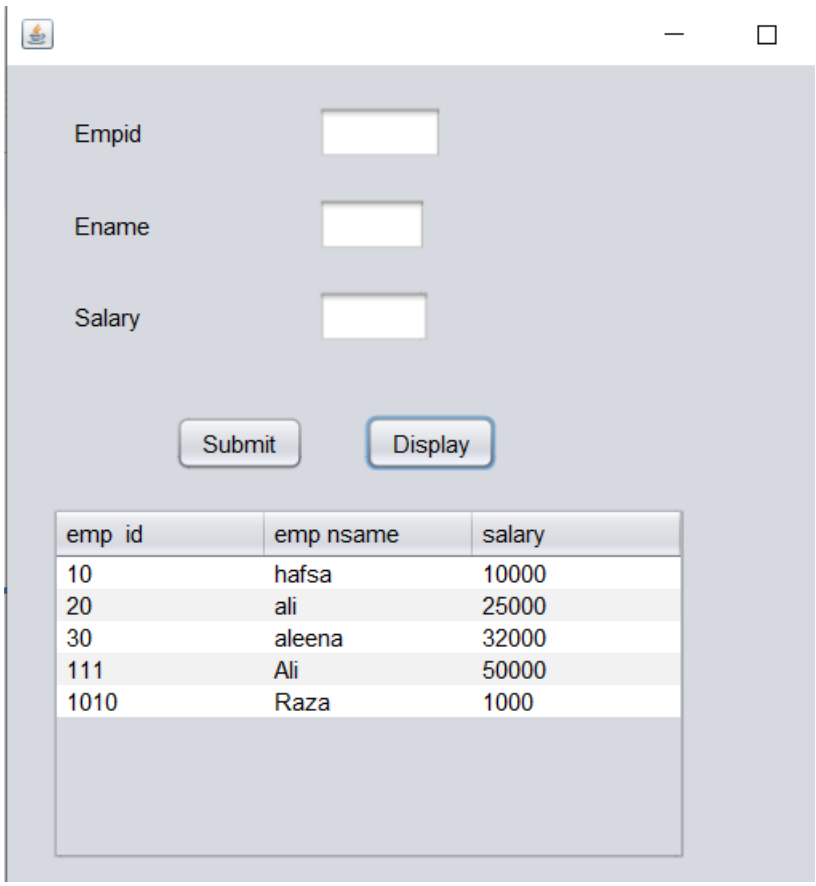
Entering Data Using form:



emp

empno	ename	sal	Click to Add
10	hafsa	10000	
20	ali	25000	
30	aleena	32000	
111	Ali	50000	
1010	Raza	1000	
* 0		0	

Fetching Data:



OOP Lab-13 Task

Name:	Syed Muhammad Raza Ali
Enrolment:	02-134231-028
Course:	OOP Lab
Faculty:	Miss Hafsa Munawar

Lab13: File Handling

Designing and implementing Java programs that deal with:

1. Reading Input From Text Files
2. Writing Output To Text Files

Exercises

Exercise

Create a system that gets following information of the employees

Employee name, Employee department, Employee contact, Employee designation, Employee Salary, Employee status and store it in the file names as **EmplyeeData.txt.** Once the data is inserted properly then fetch the data from the file display the data properly. Update record for sec and third employee and reenter the data in new file named as **UpdatedEmployeeInfo.** Ad display the updated data again.

Code:

Employee Class:

```
class Employee {
    private String name;
    private String department;
    private String contact;
    private String designation;
    private String salary;
    private String status;
    //constructor
    public Employee(String name, String department, String contact, String designation, String
salary, String status) {
        this.name = name;
        this.department = department;
        this.contact = contact;
        this.designation = designation;
        this.salary = salary;
        this.status = status;
    }
    //getters

    public String getContact() {
        return contact;
    }

    public String getDesignation() {
        return designation;
    }

    public String getDepartment() {
        return department;
    }
```

```
public String getName() {
    return name;
}

public String getSalary() {
    return salary;
}

public String getStatus() {
    return status;
}
//setters

public void setContact(String contact) {
    this.contact = contact;
}

public void setDepartment(String department) {
    this.department = department;
}

public void setDesignation(String designation) {
    this.designation = designation;
}

public void setName(String name) {
    this.name = name;
}

public void setSalary(String salary) {
    this.salary = salary;
}

public void setStatus(String status) {
    this.status = status;
```

```
    }

    //toString method
    public String toString() {
        return name + "," + department + "," + contact + "," + designation + "," + salary + "," +
status;
    }
}
```

Application class:

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.List;



public class EmployeeManagement {


    public static void main(String[] args) {
        // Initial employee data
        List<Employee> employees = new ArrayList<>();
        employees.add(new Employee("Muhammad Raza", "CS", "1234567890", "Manager",
"50000", "Active"));
        employees.add(new Employee("Muskan Khan", "CS", "9876543210", "Assistant", "40000",
"Active"));
        employees.add(new Employee("Ali", "IT", "5555555555", "Developer", "60000",
"Active"));
```

```
    // Write employee data to a file
    writeToEmployeeFile("EmployeeData.txt", employees);

    // Display initial employee data
    System.out.println("Initial Employee Data:");
    readEmployeeFile("EmployeeData.txt");

    // Update records for the second and third employee
    employees.get(1).setSalary("45000"); // Update salary for Jane Smith
    employees.get(2).setStatus("Inactive"); // Update status for Alice Johnson

    // Write updated employee data to a new file
    writeToEmployeeFile("UpdatedEmployeeInfo.txt", employees);

    // Display updated employee data
    System.out.println("\nUpdated Employee Data:");
    readEmployeeFile("UpdatedEmployeeInfo.txt");
}

// Function to write employee data to a file
public static void writeToEmployeeFile(String filename, List<Employee> employees) {
    try (PrintWriter writer = new PrintWriter(new FileWriter(filename))) {
        for (Employee employee : employees) {
            writer.println(employee.toString());
        }
    } catch (IOException e) {
        e.printStackTrace();
```

```
        }
    }

    // Function to read employee data from a file
    public static void readEmployeeFile(String filename) {
        try (BufferedReader reader = new BufferedReader(new FileReader(filename))) {
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output:

On IDE:

```
Initial Employee Data:
Muhammad Raza,CS,1234567890,Manager,50000,Active
Muskan Khan,CS,9876543210,Assistant,40000,Active
Ali,IT,5555555555,Developer,60000,Active

Updated Employee Data:
Muhammad Raza,CS,1234567890,Manager,50000,Active
Muskan Khan,CS,9876543210,Assistant,45000,Active
Ali,IT,5555555555,Developer,60000,Inactive
------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------
```

In File (Before Updating)

EmployeeData - Notepad

File Edit Format View Help

```
Muhammad Raza,CS,1234567890,Manager,50000,Active
Muskan Khan,CS,9876543210,Assistant,40000,Active
Ali,IT,5555555555,Developer,60000,Active
```

In File (After Updating)

UpdatedEmployeeInfo - Notepad

File Edit Format View Help

```
Muhammad Raza,CS,1234567890,Manager,50000,Active
Muskan Khan,CS,9876543210,Assistant,45000,Active
Ali,IT,5555555555,Developer,60000,Inactive
```