

22 September 2020

Aayush Shah

19BCE245

Practical 6

OOP Lab

Practical 6 A

Define a class Rectangle with its length and breadth.

Provide appropriate constructor(s), which gives facility of constructing rectangle object with default values of length and breadth as 0 or passing value of length and breadth externally to constructor.

Provide appropriate accessor & mutator methods to Rectangle class.

Provide methods to calculate area & to display all information of Rectangle.

Design different class TestRectangle class in separate source file, which will contain main function. From this main function, create 5 Rectangle objects by taking all necessary information from the user.

The class has attributes length and width, each of which defaults to 1. It should have member functions that calculate the perimeter and area of the rectangle. It should have set and get functions for both length and width. The set functions should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0.

CODE

FILE : TestRectangle.java

```
import java.util.Scanner;
class TestRectangle {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Choose : ");
        System.out.println("[1.] Default Rectangle      (press 1) ");
        System.out.println("[2.] Manual Rectangle      (press 2) ");
        int choice;
        choice = sc.nextInt();
        if(choice == 1){
            float l,b;
            Rectangle r = new Rectangle();    //set default value to zero
```

```

        System.out.println("Keep values as by Default (press 1)\nEnter
Values (press 2)");
        choice = sc.nextInt();
        if(choice==2){
            System.out.println("Enter length : ");
            l = sc.nextFloat();
            System.out.println("Enter breadth : ");
            b = sc.nextFloat();
            Rectangle r1 = new Rectangle(l,b);
            r1.getLength();
            r1.getBreadth();
            r1.calculateArea();
            r1.calculatePerimeter();
        }
    }
    else if(choice == 2){
        float l,b;
        l=0;
        b=0;
        int flag=0;
        Rectangle r1 = new Rectangle(l,b);
        // r1.getLength();
        // r1.getBreadth();
        while(flag==0){
            System.out.println("Enter length : ");
            l = sc.nextFloat();
            if(r1.checkNumber(l)==1){
                flag=1;
            }
            else{
                System.out.println("This value is not in Range :(");
            }
        }
        flag=0;
        while(flag==0){
            System.out.println("Enter breadth : ");
            b = sc.nextFloat();
            if(r1.checkNumber(b)==1){
                flag=1;
            }
            else{
                System.out.println("This value is not in Range :(");
            }
        }
        r1 = new Rectangle(l,b);
        r1.calculateArea();
        r1.calculatePerimeter();
    }
    else

```

```

        System.out.println("Invalid Choice :");
        Rectangle r[] = new Rectangle[5];           //This is array of objects,
references that will create 5 references pointing to separate memory location in heap
area

```

```

        //simple array of int data type that will store 5 int values in stack area.
        /*
        rectangle r -> NULL REFERENCE
        r = new rectangle() -> memory allocation is done for r .this is calling the
constructor
        */
        //that above statement will only 5 create null references
        float l,b;
        l=0;
        b=0;
        System.out.println("making and printing 5 objects : ");
        for(int i=0;i<r.length;i++){
            System.out.println("For Rectangle object : r[ " + i + " ]");
            System.out.println("Enter length : ");
            l = sc.nextFloat();
            System.out.println("Enter breadth : ");
            b = sc.nextFloat();
            r[i] = new Rectangle(l,b); //memory allocation as we have only
created NULL references in above declaration of array statement.
        //
            r[i].Rectangle(l,b);
            r[i].getLength();
            r[i].getBreadth();
            r[i].calculateArea();
            r[i].calculatePerimeter();
        }
    }
}

```

FILE : Rectangle.java

```

import java.util.Scanner;
class Rectangle{
    float length,breadth;
    Scanner sc = new Scanner(System.in);
    Rectangle(){           //constructor with 0 arguments
        System.out.println("Inside No argument Constructor : Length and Breadth
set to Zero.");
        length = breadth = 0;
    }
    Rectangle(float l, float b){           //constructor with 2 arguments
        System.out.println("Inside 2 argument Constructor : ");
        // length = l;
        // breadth = b;
        setLength(l);
        setBreadth(b);
    }
}

```

```
}
//Accessor method or getter method
void getLength(){
    System.out.println("Value of Length is " + length);
}
void getBreadth(){
    System.out.println("Value of Breadth is " + breadth);
}
//Mutator method or setter method
void setLength(float l){
    length = l;
}
void setBreadth(float b){
    breadth = b;
}
void calculateArea(){
    float area;
    area = length*breadth;
    System.out.println("Area of the Rectangle : " + area);
}
void calculatePerimeter(){
    float perimeter;
    perimeter = 2*(length+breadth);
    System.out.println("Perimeter of the Rectangle : " + perimeter);
}
int checkNumber(float number){
    if(number>0 && number<20)
        return 1;
    else
        return 0;
}
}
```

INPUT :

```
1
2
3
5
1
2
3
4
5
6
7
8
9
10
```

OUTPUT :

```
Choose :
[1.] Default Rectangle      (press 1)
[2.] Manual Rectangle      (press 2)
1
Inside No argument Constructor : Length and Breadth set to Zero.
Keep values as by Default (press 1)
Enter Values (press 2)
2
Enter length :
3
Enter breadth :
5
Inside 2 argument Constructor :
Value of Length is 3.0
Value of Breadth is 5.0
Area of the Rectangle : 15.0
Perimeter of the Rectangle : 16.0
making and printing 5 objects :
For Rectangle object : r[ 0 ]
Enter length :
1
Enter breadth :
2
Inside 2 argument Constructor :
Value of Length is 1.0
Value of Breadth is 2.0
Area of the Rectangle : 2.0
Perimeter of the Rectangle : 6.0
For Rectangle object : r[ 1 ]
Enter length :
```

Enter Values (press 2)

2

Enter length :

3

Enter breadth :

5

```
Enter length :  
3  
Enter breadth :  
4  
Inside 2 argument Constructor :  
Value of Length is 3.0  
Value of Breadth is 4.0  
Area of the Rectangle : 12.0  
Perimeter of the Rectangle : 14.0  
For Rectangle object : r[ 2 ]  
Enter length :  
5  
Enter breadth :  
6  
Inside 2 argument Constructor :  
Value of Length is 5.0  
Value of Breadth is 6.0  
Area of the Rectangle : 30.0  
Perimeter of the Rectangle : 22.0  
For Rectangle object : r[ 3 ]  
Enter length :  
7  
Enter breadth :  
8  
Inside 2 argument Constructor :  
Value of Length is 7.0  
Value of Breadth is 8.0  
Area of the Rectangle : 56.0  
Perimeter of the Rectangle : 30.0  
For Rectangle object : r[ 4 ]  
Enter length :  
9  
Enter breadth :  
10  
Inside 2 argument Constructor :  
Value of Length is 9.0  
Value of Breadth is 10.0  
Area of the Rectangle : 90.0  
Perimeter of the Rectangle : 38.0
```



Run Succeeded

Time 255 ms

Symbol ↕

Tabs: 4 ↕

Line 88, Column 3

CONCLUSION :

From the practical 6 A, We learned about the idea of class and the development of class objects. We heard about the object-oriented programming core and its execution. We also learned how to declare methods and data members within the class and how we can access them.

Practical 6 B

Create a class Term. This class represents a term of a polynomial such as $2x^4$ where 2 is coefficient and 4 is exponent of the term.

Data members:-

- int coefficient
- int exponent

Create another class Polynomial. The internal representation of a polynomial is an array of Terms. The size of this array should be fixed.

Provide a constructor for this class that will set all terms of a polynomial object as zero (where coefficient is 0 and exponent is 0).

Provide following functions:

- setTerm(int, int) – Setting a term of a polynomial object. Each successive call of this function should set next term of the polynomial object.

It should do the following validations:-

- Whether the exponent of the term being set is already used.
- Whether the array size limit is exceeded.
- Whether the exponent is negative.

In all the cases it should not set the term and display an appropriate message.

- sort() – to arrange the terms in ascending order of exponents.
- provide a function to print a polynomial object

CODE

FILE : Main.java

```
//this program can take upto 5 terms as mentioned in problem statement that size of array should be fixed.  
import java.util.Scanner;
```

```

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Polynomial P = new Polynomial();

        int choice = 0;
        while(choice!=3){
//            System.out.println("count : " + P.count);
            System.out.println("MENU : ");
            System.out.println("[1.] Set Term      (press 1)");
            System.out.println("[2.] Display Terms    (press 2)");
            System.out.println("[3.] Exit        (press 3)");
            choice = sc.nextInt();
            switch (choice) {
                case 1:
                    System.out.print("Enter Coefficient : ");
                    int coeff = sc.nextInt();
                    System.out.print("Enter Exponent : ");
                    int exp = sc.nextInt();
                    if(P.checkTerms(coeff, exp)==2)
                        System.out.println("You cannot add more terms now :(");
                    else{
                        while(P.checkTerms(coeff, exp)==0){
                            System.out.print("Enter Coefficient : ");
                            coeff = sc.nextInt();
                            System.out.print("Enter Exponent : ");
                            exp = sc.nextInt();
                        }
                        P.setTerm(coeff, exp);
                    }
                    break;
                case 2:
                    P.display();
                    break;
                case 3:
                    System.out.println("*** THANK YOU ***");
                    break;
                default:
                    System.out.println("Invalid Input :(");
                    break;
            }
        }
    }
}

```

FILE : Term.java

```

public class Term {
    int coefficient,exponent;
    Term(){
        coefficient=0;
    }
}

```



```

        exponent=-1;
        //Exponents of every term is set to -1 so that we can also add 0 to the
        exponent as we are checking for condition that adding exponent should not be
        already present in the polynomial so setting 0 as exponent of every term , then we
        cannot assign zero to any term's exponent as it will check that if the given exponent
        value is already present or not in the polynomial
    }
    int getCoefficient(){
        return coefficient;
    }
    int getExponent(){
        return exponent;
    }
    void setCoefficient(int coefficient){
        this.coefficient = coefficient;
    }
    void setExponent(int exponent){
        this.exponent = exponent;
    }
}

```

FILE : Polynomial.java

```

import java.util.Scanner;
public class Polynomial {
    Scanner sc = new Scanner(System.in);
    Term T[] = new Term[5];
    int count=0;
    // Term T = new Term();
    Polynomial(){
        for(int i=0;i<5;i++){
            T[i] = new Term();           //set all values of terms to zero
        }
    }
    // void setTerms(){
    //     for(int i=0;i<5;i++){
    //         System.out.print("Enter Coefficient : ");
    //         int coeff = sc.nextInt();
    //         System.out.print("Enter Exponent : ");
    //         int exp = sc.nextInt();
    //         this.setTerms(coeff,exp,i);
    //     }
    // }
    void setTerm(int coefficient, int exponent){
        T[count].setCoefficient(coefficient);
        T[count++].setExponent(exponent);
    //     if(count==5)
    //         this.sort();
    }
    int checkTerms(int coeff, int exp){

```

```

        int flag1,flag2,flag3;
        flag1=0;
        flag2=0;
        flag3=0;
        int i;
        for(i=0;i<5;i++){          //checking exponent of the term is being set is
already used condition
            if(T[i].exponent==exp){
already used :("");
                System.out.println("The exponent of the term being set is
already used :(");
                break;
            }
        }
        if(i==5)
        flag1=1;
        if(count>=5){          //checking array size limit condition
            System.out.println("the array size limit is exceeded :(");
            return 2;
        }
        else{
            flag2=1;
        }
        if(exp<0){
            System.out.println("The exponent is negative :(");
        }
        else{
            flag3=1;
        }
        if(flag1==1 && flag2==1 && flag3==1)
        return 1;
        else
        return 0;
    }
    void sort(){          //sorts the terms in ascending order of exponents
        for(int pick=0;pick<count;pick++){
            for(int comp=pick+1;comp<count;comp++){
                if(T[pick].exponent>T[comp].exponent){
                    int temp = T[pick].exponent;          //swapping exponent
                    T[pick].exponent = T[comp].exponent;
                    T[comp].exponent = temp;
                    temp = T[pick].coefficient;          //swapping coefficient
                    T[pick].coefficient = T[comp].coefficient;
                    T[comp].coefficient = temp;
                }
            }
        }
    }
    void display(){
        for(int i=0;i<count;i++){

```

```

        System.out.print(T[i].coefficient + "x^(" + T[i].exponent + ") + ");
    }
    System.out.println();
}
}

```

INPUT :

```

1
1
3
1
2
1
1
4
5
2
3

```

OUTPUT :

```

MENU :
[1.] Set Term   (press 1)
[2.] Display Terms (press 2)
[3.] Exit       (press 3)
1
Enter Coefficient : 1
Enter Exponent : 3
MENU :
[1.] Set Term   (press 1)
[2.] Display Terms (press 2)
[3.] Exit       (press 3)
1
Enter Coefficient : 2
Enter Exponent : 1
MENU :
[1.] Set Term   (press 1)
[2.] Display Terms (press 2)
[3.] Exit       (press 3)
1
Enter Coefficient : 4
Enter Exponent : 5
MENU :
[1.] Set Term   (press 1)
[2.] Display Terms (press 2)
[3.] Exit       (press 3)
2
2x^(1) + 1x^(3) + 4x^(5) +
MENU :
[1.] Set Term   (press 1)
[2.] Display Terms (press 2)
[3.] Exit       (press 3)

```

Run Succeeded Time 219 ms Symbol ↕ Tabs: 4 ↕ 46 lines, 1308 characters

CONCLUSION :

From the practical 6 B, We got a hands-on class and object formation from this realistic one. We learned how we can construct an array of objects and how many classes can be managed together. We also learned how, just like swapping every other primitive data form, we can swap objects.

Practical 6 C

Create a class called complex for performing arithmetic operations with complex numbers. Use floating point variables to represent the private data of the class. Provide a default constructor that initializes the object with some default values. Provide public member functions for each of the following

- Addition of two complex numbers: It returns the result obtained by adding the respective real parts and the imaginary parts of the two complex numbers.
- Subtraction of two complex numbers: It returns the result obtained by subtracting the respective real parts and the imaginary parts of the two complex numbers.
- `display()` – It displays the complex number in `a+bi` format.

The output should be displayed as follows:-

Sum of a_1+b_1i & a_2+b_2i is : a_3+b_3i

CODE :

FILE : Main.java

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int choice;
        Complex l = new Complex();
        Complex l1 = new Complex();
        Complex l2 = new Complex();
        float real,imaginary;
        choice = 0;
        while(choice!=4){
            System.out.println("MENU : ");
            System.out.println("[1.]create a complex number and Display it
            (press 1)");
```

```
2)");  
System.out.println("[2.]Addition of two complex number (press  
(press 3)");
```

```
System.out.println("[4.]Exit (press 4)");  
System.out.print("Choice : ");  
choice = sc.nextInt();  
switch (choice) {  
    case 1:  
        System.out.print("Enter real part : ");  
        real = sc.nextInt();  
        System.out.print("Enter imaginary part : ");  
        imaginary = sc.nextInt();  
        l.setReal(real);  
        l.setImaginary(imaginary);  
        System.out.print("Complex Representation : ");  
        Complex.Display(l);  
        break;  
    case 2:  
        System.out.println("For number 1 :");  
        System.out.print("Enter real part : ");  
        real = sc.nextInt();  
        System.out.print("Enter imaginary part : ");  
        imaginary = sc.nextInt();  
        l1.setReal(real);  
        l1.setImaginary(imaginary);  
        System.out.println("For number 2 :");  
        System.out.print("Enter real part : ");  
        real = sc.nextInt();  
        System.out.print("Enter imaginary part : ");  
        imaginary = sc.nextInt();  
        l2.setReal(real);  
        l2.setImaginary(imaginary);  
        System.out.print("Complex Addition : ");  
        Complex.Display(Complex.Addition(l1,l2));  
        break;  
    case 3:  
        System.out.println("For number 1 :");  
        System.out.print("Enter real part : ");  
        real = sc.nextInt();  
        System.out.print("Enter imaginary part : ");  
        imaginary = sc.nextInt();  
        l1.setReal(real);  
        l1.setImaginary(imaginary);  
        System.out.println("For number 2 :");  
        System.out.print("Enter real part : ");  
        real = sc.nextInt();  
        System.out.print("Enter imaginary part : ");  
        imaginary = sc.nextInt();
```

```

        I2.setReal(real);
        I2.setImaginary(imaginary);
        System.out.print("Complex Subtraction :
");Complex.Display(Complex.Addition(I1,I2));
        Complex.Display(Complex.Subtraction(I1,I2));
        break;
    case 4:
        System.out.println("*** THANK YOU ***");
        break;
    default:
        System.out.println("Invalid Choice :(");
        break;
    }
}
}
}

```

```

}

```

FILE : Complex.java

```

//import java.awt.*;

```

```

public class Complex {
    private float real,imaginary;
    Complex(){           //default constructor
        real = 1;
        imaginary = 1;
    }
    Complex(float real, float imaginary){
        this.real = real;
        this.imaginary = imaginary;
    }
    void setReal(float real){
        this.real = real;
    }
    void setImaginary(float imaginary){
        this.imaginary = imaginary;
    }
    float getReal(){
        return real;
    }
    float getImaginary(){
        return imaginary;
    }
    public static Complex Addition(Complex I1,Complex I2){
        Complex result = new Complex();
        result.real = I1.real + I2.real;
        result.imaginary = I1.imaginary + I2.imaginary;
        return result;
    }
    public static Complex Subtraction(Complex I1,Complex I2){

```

```
        Complex result = new Complex();
        result.real = l1.real - l2.real;
        result.imaginary = l1.imaginary - l2.imaginary;
        return result;
    }
    public static void Display(Complex l1){
        System.out.println(l1.real + " + " + l1.imaginary + "i");
    }
}
```

INPUT :

```
2
1
4
2
5
3
5
4
3
2
```

OUTPUT :

```
MENU :
[1.]create a complex number and Display it (press 1)
[2.]Addition of two complex number (press 2)
[3.]Subtraction of two complex number (press 3)
[4.]Exit (press 4)
Choice : 2
For number 1 :
Enter real part : 1
Enter imaginary part : 4
For number 2 :
Enter real part : 2
Enter imaginary part : 5
Complex Addition : 3.0 + 9.0i
MENU :
[1.]create a complex number and Display it (press 1)
[2.]Addition of two complex number (press 2)
[3.]Subtraction of two complex number (press 3)
[4.]Exit (press 4)
Choice : 3
For number 1 :
Enter real part : 5
Enter imaginary part : 4
For number 2 :
Enter real part : 3
Enter imaginary part : 2
Complex Subtraction : 8.0 + 6.0i
2.0 + 2.0i
MENU :
[1.]create a complex number and Display it (press 1)
[2.]Addition of two complex number (press 2)
[3.]Subtraction of two complex number (press 3)
[4.]Exit (press 4)
Choice : 4
Exit
```

Run Succeeded Time 227 ms Symbol 4 Tabs: 4 76 lines, 2482 characters

CONCLUSION :

From the practical 6 C, We have mainly made use of objects and classes. We have created a method for the first time that returns an object. So we built a method with an object return type. For the first time, we used the keyword and acknowledged the importance of its use. In a method to perform operations on it, we then accessed data members of 2 objects of the same class simultaneously and got an object as our output.

Practical 6 D

Create an object called GSSArray. (It stands for growable self-sorting array)

This object will manage an array of type int. Create a private variable for an array of type int. In the constructor for this object, take in an int value which will determine the starting size of the array. The constructor should also instantiate the array.

Create a public method called insert, which will take in an int and find the location in the array where it belongs and insert it there. If the array is full, then before inserting the value, method insert should call private method increaseSize, which will create a new array which is double the size of the current array. Then it will copy the values from the original array into the new array and set the private variable to this new array.

The array should keep track of how many of its indexes are filled. Create a private variable called lastIndex which will be equal to the last index of the array that has a value.

Create a public method delete, which will take an int and it will remove the 1st instance of that number in the array. If the number doesn't exist, the method should return false, otherwise it should return true. (Don't forget to update variable lastIndex in methods delete and insert.)

CODE :

FILE : GSS.java

```
import java.util.Scanner;
```

```
public class GSS
{
    private int arr[]=new int[5];
    private int lastIndex;
    private int size;

    public GSS()
    {
        lastIndex=0;
        size = 5;
    }

    public void insert( int x )
    {
```



```
        if ( lastIndex==size )
        {
            increaseSize(x);
        }
        else
        {
            arr[lastIndex++]=x;
        }
    }

    private void increaseSize(int x)
    {
        int temp[] = new int[2*size];
        for ( int i=0 ; i<size ; i++ )
        {
            temp[i]=arr[i];
        }
        temp[lastIndex++]=x;
        size=size*2;
        arr = new int[size];
        for ( int i=0 ; i<=lastIndex ; i++ )
        {
            arr[i]=temp[i];
        }
    }

    public void delete( int x )
    {
        int z = lastIndex;
        for ( int i=0 ; i<=lastIndex ; i++ )
        {
            if ( arr[i]==x )
            {
                for ( int j=i+1 ; j<=lastIndex ; j++ )
                {
                    arr[j-1]=arr[j];
                }
                lastIndex--;
                break;
            }
        }
        if ( z==lastIndex )
        {
            System.out.println("The number cannot be found");
        }
        else
        {
            System.out.println("The number is deleted");
        }
    }
}
```

```
}

public void display()
{
    System.out.print("GSS Array : ");
    int n = lastIndex;
    for (int i = 1 ; i < n; ++i)
    {
        int key = arr[i];
        int j = i - 1;

        while (j >= 0 && arr[j] > key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
    for ( int i=0 ; i<lastIndex ; i++ )
    {
        System.out.print(arr[i]+" ");
    }
    System.out.println();
}

public static void main(String[] args)
{
    Scanner sc = new Scanner ( System.in );

    GSS o1 = new GSS();
    int option;
    do
    {

        System.out.println("Enter the appropriate option. Press 0 to exit");
        System.out.println("1. Insert");
        System.out.println("2. Delete");
        System.out.println("3. Display");

        option = sc.nextInt();

        switch(option)
        {
            case 0:
                break;
            case 1:
                int a;
                System.out.println("Enter the number");
                a = sc.nextInt();
            case 2:
            case 3:
                break;
        }
    }
    while (option != 0);
}
```

```
        o1.insert(a);
        break;
    case 2:
        int b;
        System.out.println("Enter the number which you want to
delete");

        b = sc.nextInt();
        o1.delete(b);
        break;
    case 3:
        o1.display();
        break;
    default:
        System.out.println("Enter appropriate number");
    }

    }while(option!=0);


    sc.close();
}
}
```

INPUT :


1
1
1
3
1
5
1
22
1
4
1
6
3
0

OUTPUT :

```
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
1
Enter the number
1
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
1
Enter the number
3
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
1
Enter the number
5
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
1
Enter the number
22
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
```

 Run Succeeded


Time 206 ms

 increaseSize ↕


Tabs: 4 ↕

Line 29, Column 34

```
Enter the number
22
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
1
Enter the number
4
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
1
Enter the number
6
Now, Size of the Array is two times of the original
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
3
GSS Array : 1 3 4 5 6 22
Enter the appropriate option. Press 0 to exit
1. Insert
2. Delete
3. Display
0
```

 Run Succeeded

Time 206 ms

 IncreaseSize ↕

Tabs: 4 ↕

Line 29, Column 34

CONCLUSION :

From the practical 6 D, We made use of class definitions and the majority of objects. But we used access modifiers and public, private keywords here to limit their access according to our needs. To build our menu-driven software, we also made use of switch-case statements. As per question requirement, we generated multiple methods. The remainder was regular programming.