Tugas 7 : Polymorphism

Pemrograman Berorientasi Objek

Nama : Agista Diva Briliani

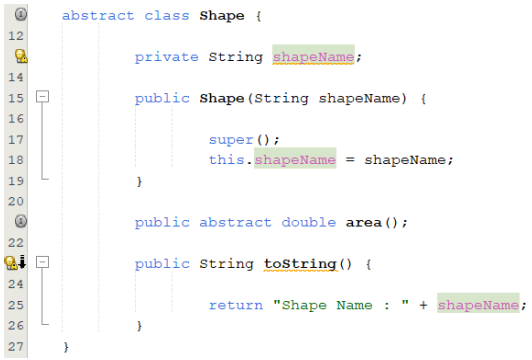
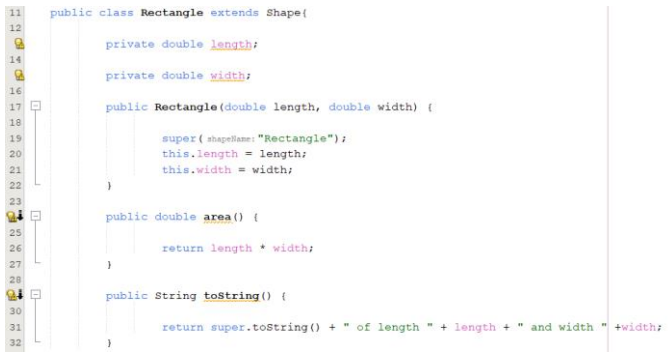
NIM : 221511038

Prodi : D3 Teknik Informatika

Kelas : 2B JTK

PROGRAM STUDI DIII TEKNIK INFORMATIKA

POLITEKNIK NEGERI BANDUNG

Hasil Akhir Program	Keterangan
Studi Kasus 1 : - Commission.java  - Staff.java 	Output :  1. Menambahkan class baru yaitu commission. 2. Update class Staff. Letak Polymorphism : Ketika memanggil methods ‘pay’ pada objek yang disimpan dalam array ‘staffList’ dalam methods ‘payday’ di class Staff.
Studi Kasus 2 : - Shape.java  - Rectangle.java 	Output :  1. Membuat abstract class shape. 2. Membuat class Rectangle dan Cylinder. 3. Menyesuaikan file paintThings dengan apa yang diperintahkan. Letak Polymorphism : Dalam abstract class Shape yang memiliki methods abstract area(). Dengan karakteristik bentuk geometris masing masing. Polymorphism ini memungkinkan untuk menghitung jumlah cat yang dibutuhkan untuk berbagai bentuk geometris tanpa perlu mengubah implementasi methods amount.

- Cylinder.java

```java
public class Cylinder extends Shape {

        private double radius;
        private double height;

        public Cylinder(double radius, double height) {

                super( shapeName: "Cylinder");
                this.radius = radius;
                this.height = height;
        }

        public double area() {

                return Math.PI*(Math.pow( a: radius,  b: 2))*height;
        }

        public String toString() {

                return super.toString() + " of radius " + radius + " and height " +height;
        }
}
```

- PaintThings.java

```java
import java.text.DecimalFormat;

public class PaintThings {

        public static void main(String[] args) {

                final double COVERAGE = 350;
                Paint paint = new Paint( n: COVERAGE);

                Rectangle deck = new Rectangle( length: 20, width: 35);
                Sphere bigBall = new Sphere( r: 15);
                Cylinder tank = new Cylinder( radius: 10,  height: 30);

                double deckAmt = paint.amount( s: deck);
                double ballAmt = paint.amount( s: bigBall);
                double tankAmt = paint.amount( s: tank);

                DecimalFormat fmt = new DecimalFormat( pattern: "0.#");
                System.out.println( x: "\nNumber of gallons of paint needed...");
                System.out.println("Deck " + fmt.format( number: deckAmt));
                System.out.println("Big Ball " + fmt.format( number: ballAmt));
                System.out.println("Tank " + fmt.format( number: tankAmt));
        }

}
```

Studi Kasus 3 :

- String.java

```java
import java.util.Scanner;

public class Strings {

        public static void main(String[] args) {

                Comparable[] intList;
                int size;

                Scanner scan = new Scanner( source: System.in);

                System.out.println( x: "\nHow many word do you want to sort? ");
                size = scan.nextInt();
                intList = new Comparable[size];

                System.out.println( x: "\nEnter the word...");
                for (int i =0; i < size; i++)
                        intList[i] = scan.nextLine();
                Sorting.selectionSort( list: intList);
                System.out.println( x: "Your word in sorted order...");
                for ( int i = 0; i < size; i++)
                        System.out.println(intList[i] + " ");
                System.out.println();
        }

}
```

- Sorting.java

```java
public class Sorting {

        public static void selectionSort(Comparable[] list) {
                int min;
                Comparable temp;

                for (int index = 0; index < list.length-1 ; index++) {
                        min = index;
                        for (int scan = index + 1; scan < list.length; scan++)
                                if (list[scan].compareTo(list[min]) > 0)
                                        min = scan;

                        temp = list[min];
                        list[min] = list[index];
                        list[index] = temp;

                }
        }

        public static void insertionSort (Comparable[] list) {
                for (int index =1; index < list.length ; index++) {
                        Comparable key = list[index];
                        int position = index;

                        while (position>0 && key.compareTo(list[position-1])>0) {
                                list[position] = list[position-1];
                                position--;
                        }
                        list[position] = key;
                }
        }
}
```

Output :

```
--------------------< com.mycompany:WeeklySales >---------------------
Building WeeklySales 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ WeeklySales ---

Ranking of Sales for the week

Taylor, Harry:  7300
Adams, Andy:    5000
Duck, Daffy:    4935
Jones, Jane:    3000
Jones, James:   3000
Black, Jane:    3000
Smith, Walt:    3000
Doe, Jim:       2850
Walter, Dick:   2800
Trump, Don:     1570
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.345 s
Finished at: 2023-10-20T22:28:12+07:00
------------------------------------------------------------------------
```

1. Mengcompile dan membetulkan fungsi yang kurang tepat pada class number.
2. Modif InsertsionSort.
3. Membuat class string, untuk membaca array objek String dan mengurutkannya.
4. Menggunakan nama pegawai penjualan.

- WeeklySales.java

```
public class WeeklySales {

    public static void main(String[] args) {

        SalesPerson[] salesStaff = new SalesPerson[10];

        salesStaff[0] = new SalesPerson( first: "Jane",  last: "Jones",  sales: 3000);
        salesStaff[1] = new SalesPerson( first: "Daffy",  last: "Duck",  sales: 4935);
        salesStaff[2] = new SalesPerson( first: "James",  last: "Jones",  sales: 3000);
        salesStaff[3] = new SalesPerson( first: "Dick",  last: "Walter",  sales: 2800);
        salesStaff[4] = new SalesPerson( first: "Don",  last: "Trump",  sales: 1570);
        salesStaff[5] = new SalesPerson( first: "Jane",  last: "Black",  sales: 3000);
        salesStaff[6] = new SalesPerson( first: "Harry",  last: "Taylor",  sales: 7300);
        salesStaff[7] = new SalesPerson( first: "Andy",  last: "Adams",  sales: 5000);
        salesStaff[8] = new SalesPerson( first: "Jim",  last: "Doe",  sales: 2850);
        salesStaff[9] = new SalesPerson( first: "Walt",  last: "Smith",  sales: 3000);

        Sorting.insertionSort( list: salesStaff);

        System.out.println( x: "\nRanking of Sales for the week\n");

        for (SalesPerson s:salesStaff)
            System.out.println( x: s);

    }
```

Letak Polymorphism :
Dalam penggunaan interface Comparable yang digunakan untuk mendefine methods ‘compareTo’.

Teman yang membantu :

- Mahesya Setia Nugraha

Source Code : https://github.com/agistadivab/PBO.git