ANS Elbląg

Instytut Informatyki Stosowanej im. Krzysztofa Brzeskiego

Programowanie obiektowe I – laboratorium

Studium Stacjonarne, sem. 3, 2022/2023

Sprawozdanie nr : 9,

nr grupy: 1,

dzień: wtorek,

godz. 12:00.

Data wykonania ćwiczenia: 20.12

Data oddania sprawozdania: 30.01

Nazwisko i imię: Kuczawski Kacper

Nr albumu: 20195

Nazwa pliku : lab9_kuczawski_kacper20195

1. Utworzyć klasę publiczną **Lab9z1** oraz klasę **Figura** z metodą **rysuj**, polem **dlugosc** i konstruktorem z jednym parametrem nadającym wartość temu polu. Następnie dla klasy **Figura** utworzyć klasy pochodne **Linia**, **Kwadrat** i zaimplementować w każdej z nich metodę **rysuj()** (każda z tych metod może rysować dany kształt lub wyprowadzać odpowiedni napis). W metodzie **main()** utworzyć 4 – elementową tablicę obiektów **Figura** i umieścić w niej 2 obiekty klasy **Linia** i 2 obiekty klasy **Kwadrat**, a następnie narysować poszczególne figury poprzez wywołanie odpowiedniej metody.

Treść:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.text.*;
public class temp{
    public static void main(String args[]){

        Figura tab[] = new Figura[4];

        Linia L1   = new Linia(3);
        Linia L2   = new Linia(5);
        Kwadrat K1 = new Kwadrat(4);
        Kwadrat K2 = new Kwadrat(5);

        tab[0] = L1;
        tab[1] = L2;
        tab[2] = K1;
        tab[3] = K2;

        for(int i=0; i<4; i++){
            tab[i].rysuj();
        }
    }
}

class Figura{

    int dlugosc;

    Figura(int d){
        this.dlugosc = d;
    }
    void rysuj(){
        System.out.print("\n-- Rysowanie --");
    }
}
class Linia extends Figura{
    Linia(int l){
        super (l);
    }
```

```
    void rysuj(){
        super.rysuj();
        System.out.print("\n--    Lini   --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            System.out.print("_");
        }
        System.out.print("\n");
    }
}
class Kwadrat extends Figura{
    Kwadrat(int a){
        super (a);
    }
    void rysuj(){
        super.rysuj();
        System.out.print("\n-- Kwadratu  --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            for(int j=0; j<dlugosc; j++){
                System.out.print("o");
            }
            System.out.print("\n");
        }
        System.out.print("\n");
    }
}
```

Wyniki:

```
-- Rysowanie --
--    Lini   --
___

-- Rysowanie --
--    Lini   --
_____
```

```
-- Rysowanie --
-- Kwadratu  --
oooo
oooo
oooo
oooo
```

```
-- Rysowanie --
-- Kwadratu  --
ooooo
ooooo
ooooo
ooooo
ooooo
```

2. Utworzyć klasę publiczną **Lab9z2** oraz klasę **FiguraN** będącą klasą abstrakcyjną zawierającą te same elementy co klasa **Figura**, w formie odpowiednio zmodyfikowanej. W metodzie **main()** klasy **Lab9z2** pobrać informację od użytkownika, jaka figura ma być tworzona (obiekt klasy **Linia** czy obiekt klasy **Kwadrat**) oraz jej rozmiar, a następnie utworzyć i na rysować wybraną figurę o podanym rozmiarze.

Treść:

```
import java.io.*;
import java.util.*;
```

```java
import java.text.*;
import java.math.*;
import java.text.*;
public class temp{
    public static void main(String args[]){

        int d, w;

        Scanner sc=new Scanner(System.in);
        do{
            System.out.print("Podaj figurę (1-linia, 2-kwadrat):");
            while(! sc.hasNextInt()){
                System.out.print("Blad, podaj poprawnie:");
                sc.next();
            }
            w = sc.nextInt();
            System.out.print("Podaj długość:");
            while(! sc.hasNextInt()){
                System.out.print("Blad, podaj poprawnie:");
                sc.next();
            }
            d = sc.nextInt();

            if(w==1){
                Linia L = new Linia(d);
                L.rysuj();
            } else if(w==2){
                Kwadrat K = new Kwadrat(d);
                K.rysuj();
            }
        } while(w!=1 && w!=2);

    }
}

abstract class FiguraN{

    int dlugosc;

    FiguraN(int d){
        dlugosc = d;
    }
    void rysuj(){
        System.out.print("\n-- Rysowanie --");
    }
}
class Linia extends FiguraN{

    Linia(int l){
```

```
        super (l);
    }

    void rysuj(){
        super.rysuj();
        System.out.print("\n--    Lini   --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            System.out.print("_");
        }
        System.out.print("\n");
    }

}
class Kwadrat extends FiguraN{

    Kwadrat(int a){
        super (a);
    }

    void rysuj(){
        super.rysuj();
        System.out.print("\n-- Kwadratu  --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            for(int j=0; j<dlugosc; j++){
                System.out.print("o");
            }
            System.out.print("\n");
        }
        System.out.print("\n");
    }

}
```

Wyniki:

```
Podaj figurę (1-linia, 2-kwadrat):1
Podaj długość:5

-- Rysowanie --
--    Lini   --

_____

Process finished with exit code 0
```

3. Utworzyć klasę publiczną **Lab9z3**. Skopiować klasę **Figura** i klasy **Linia** i **Kwadrat**. W metodzie **main()** klasy **Lab9z3** **wy**losować liczbę całkowitą. Jeśli zostanie wylosowana liczba parzysta, to tworzony jest obiekt klasy **Linia**, jeśli nieparzysta, to klasy **Kwadrat**. Następnie wypisać na ekranie nazwę klasy stworzonego obiektu. Do określenia nazwy klasy tworzonego obiektu zastosować metodę **getClass**. Sposób określania nazwy klasy podano po zadaniu 6.

Treść:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.text.*;
public class temp{
    public static void main(String args[]){

        Random r = new Random();

        int d = r.nextInt(20);

        System.out.print("\n Liczba: "+d+"\n");

        if(d%2==0){
            Linia L = new Linia(d);
            L.rysuj();
            Class a = L.getClass();
            System.out.print("\n Klasa: "+a.getName()+"\n");
        } else if(d%2!=0){
            Kwadrat K = new Kwadrat(d);
            K.rysuj();
            Class a = K.getClass();
            System.out.print("\n Klasa: "+a.getName()+"\n");
        }
    }
}

class Figura{

    int dlugosc;

    Figura(int d){
        dlugosc = d;
    }
    void rysuj(){
        System.out.print("\n-- Rysowanie --");
    }
}
class Linia extends Figura{
    Linia(int l){
        super (l);
    }
```

```
    void rysuj(){
        super.rysuj();
        System.out.print("\n--    Lini   --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            System.out.print("_");
        }
        System.out.print("\n");
    }
}
class Kwadrat extends Figura{

    Kwadrat(int a){
        super (a);
    }

    void rysuj(){
        super.rysuj();
        System.out.print("\n-- Kwadratu  --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            for(int j=0; j<dlugosc; j++){
                System.out.print("o");
            }
            System.out.print("\n");
        }
        System.out.print("\n");
    }
}
```

Wyniki:

```
 Liczba: 7

-- Rysowanie --
-- Kwadratu  --
ooooooo
ooooooo
ooooooo
ooooooo
ooooooo
ooooooo
ooooooo


 Klasa: Kwadrat

Process finished with exit code 0
```

```
 Liczba: 18

-- Rysowanie --
--    Lini   --
__________________

 Klasa: Linia

Process finished with e
```

4. Skopiować klasę **Figura** i klasy **Linia** i **Kwadrat**. Napisać klasę publiczną **Lab9Z4** zawierającą statyczną metodę **test**, która pobiera jako argument obiekt klasy **Figura** i rysuje figurę odpowiadającą temu argumentowi i dodatkowo, jeśli figura jest kwadratem, wypisuje jego pole. Następnie w metodzie **main()**, utworzyć obiekt **lin1** klasy **Linia** oraz obiekt **kw1** klasy **Kwadrat** i wylosować liczbę całkowitą. Jeśli zostanie wylosowana liczba parzysta, to wywoływana jest metoda **test** z argumentem klasy **Linia**, jeśli nieparzysta, to z argumentem typu klasy **Kwadrat**.

Treść:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.text.*;
public class temp{
    public static void main(String args[]){

        Random r = new Random();

        int d = r.nextInt(20);

        System.out.print("\n Liczba: "+d+"\n");

        if(d%2==0){
            Linia L = new Linia(d);
            test(L);
        } else if(d%2!=0){
            Kwadrat K = new Kwadrat(d);
            test(K);
        }
    }
    static void test(Figura f){
        int d;
        Class a = f.getClass();
        String s = a.getName();
        if(s == "Linia"){
            f.rysuj();
        } else if(s == "Kwadrat"){
            int P;
            f.rysuj();
            P = f.dlugosc*f.dlugosc;
            System.out.print("\nPole kwadratu: "+P+"\n");
        }
    }
}

class Figura{

    int dlugosc;
```

```
    Figura(int d){
        this.dlugosc = d;
    }
    void rysuj(){
        System.out.print("\n-- Rysowanie --");
    }
}
class Linia extends Figura{
    Linia(int l){
        super (l);
    }
    void rysuj(){
        super.rysuj();
        System.out.print("\n--    Lini   --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            System.out.print("_");
        }
        System.out.print("\n");
    }
}
class Kwadrat extends Figura{
    Kwadrat(int a){
        super (a);
    }
    void rysuj(){
        super.rysuj();
        System.out.print("\n-- Kwadratu  --");
        System.out.print("\n");
        for(int i=0; i<dlugosc; i++){
            for(int j=0; j<dlugosc; j++){
                System.out.print("o");
            }
            System.out.print("\n");
        }
        System.out.print("\n");
    }
}
```

Wyniki:

```
 Liczba: 16

-- Rysowanie --
--    Lini   --

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

Process finished with exit code 0
```

```
 Liczba: 3

-- Rysowanie --
-- Kwadratu  --
ooo
ooo
ooo


Pole kwadratu: 9
```

5. Uruchomić program z przykładu 6.2 z wykładu nr 6. Wstawić do sprawozdania program i wyniki działania.

Treść:

```
import java.io.*;
import java.util.*;
class Point { // klasa bazowa
    int x; int y;
    Point () {
        System.out.println(" Konstruktor bezparametrowy klasy bazowej");
    }
    Point (int x, int y) {
        System.out.println("Konstruktor klasy bazowej z dwoma parametrami");
        this.x=x;
        this.y=y;
    }
    Point (Point punkt1) {
        x=punkt1.x;
        y=punkt1.y;
        System.out.println("Konstruktor klasy bazowej z jednym parametrem -
obiektem klasy Point");
    }
}
class Point3D extends Point// klasa pochodna
{ int z;
    Point3D ( ) {
        System.out.println("Konstruktor bezparametrowy klasy pochodnej");
    }
    Point3D ( int x, int y, int z) {
        super(x,y);// wywołanie konstruktora klasy bazowej
        this.z=z;
        System.out.println("Konstruktor klasy pochodnej z trzema parametrami
");
    }
}// zakończenie deklaracji klasy pochodnej
public class temp {
    public static void main(String args[]) throws IOException {
        Point punkt1, punkt2, punkt3;
        System.out.println(" Tworzenie obiektu punkt1 klasy Point");
        punkt1 = new Point();
        System.out.println(" Tworzenie obiektu punkt2 klasy Point");
        punkt2 = new Point(10, 20);
        System.out.println(" Tworzenie obiektu punkt3 klasy Point");
        punkt3 = new Point(punkt2);
        System.out.println(" punkt1.x=" + punkt1.x);
        System.out.println(" punkt1.y=" + punkt1.y);
        System.out.println(" punkt2.x=" + punkt2.x);
        System.out.println(" punkt2.y=" + punkt2.y);
        System.out.println(" punkt3.x=" + punkt3.x);
        System.out.println(" punkt3.y=" + punkt3.y);
```

```
        Point3D obiektPoch1, obiektPoch2;
        obiektPoch1 = new Point3D();// tworzenie obiektu 1 klasy pochodnej
        obiektPoch1 = new Point3D(1, 2, 3);// tworzenie obiektu klasy
pochodnej
    }
}
```

Wyniki:

```
 Tworzenie obiektu punkt1 klasy Point
 Konstruktor bezparametrowy klasy bazowej
 Tworzenie obiektu punkt2 klasy Point
Konstruktor klasy bazowej z dwoma parametrami
 Tworzenie obiektu punkt3 klasy Point
Konstruktor klasy bazowej z jednym parametrem - obiektem klasy Point
 punkt1.x=0
 punkt1.y=0
 punkt2.x=10
 punkt2.y=20
 punkt3.x=10
 punkt3.y=20
 Konstruktor bezparametrowy klasy bazowej
Konstruktor bezparametrowy klasy pochodnej
Konstruktor klasy bazowej z dwoma parametrami
Konstruktor klasy pochodnej z trzema parametrami
```