Лабораторна робота №13

Поліморфізм.Форматований вивід

***Мета робот:***

- використання шаблонів проектування Command, Singleton и Factory

Method;

- використання модульного тестування.

**1 ЗАГАЛЬНЕ ЗАВДАННЯ**

Используя созданные ранее классы и шаблон проектирования Command,

разработать класс Menu как расширяемый контейнер команд, реализовать

обработку данных коллекции и отдельных элементов (масштабирование,

интерполяция, нормализация, сортировка, поиск и т.д.).

Реализовать возможность отмены (undo) операций (команд).

Продемонстрировать понятие "макрокоманда".

При разработке приложения использовать шаблон Singletone.

Обеспечить диалоговый интерфейс с пользователем.

Разработать класс для тестирования функциональности приложения.

Использовать комментарии для автоматической генерации документации

средствами javadoc.

**2 ОПИС ПРОГРАМИ**

**2.1 Ієрархія та структура класів**

**class Info-організовує діалог з користувачем**

**class Calculator виконує індивідуальне завдання**

**class Serializator виконує збереження данних классу Calculator у файл**

**interface Calculator- містить основні методи калькуляторів**

**class Calculator Area –рахує площу**

**class Calculator FactoreArea – створює об єкт типу CalculatorArea**

**class Calculator Perimetr – рахує періметр індивідуального завдання**

**class Calculator FactoryPerimetr- створює об єкт типу CalculatorPerimetr**

**class GuiCalulator-додатковий класс для графічного інтерфейсу**

**class Main –викликає класс контроллер та файл з версткою**

**sample.fxml-містить верстку вікна Windows**

**2.2 Опис програми**

На рисунку 2.1 наведена структура розробленої програми

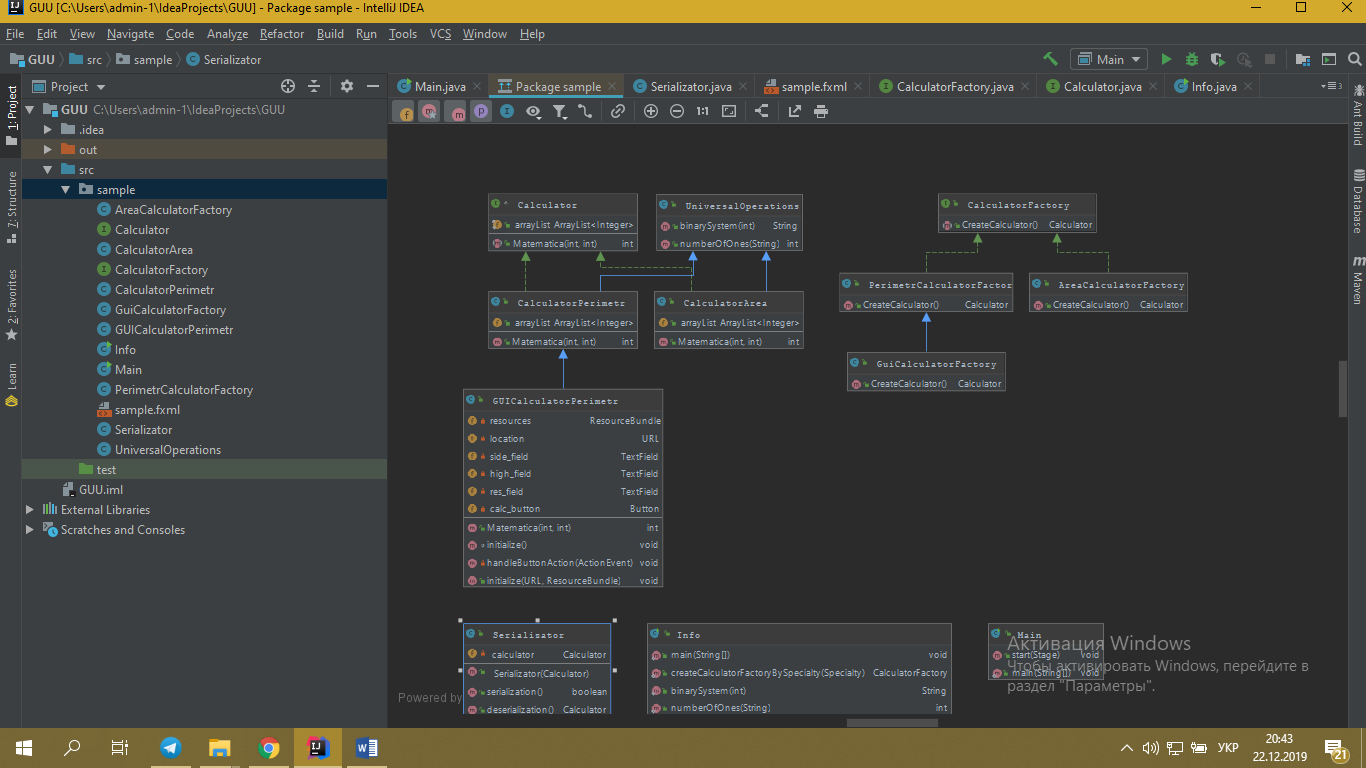


Рис 2.1 – структура програми

**2.3 Важливі фрагменти програми**

2.3.1 Файл*GuiCalculatorPerimetr.fxml*

package sample;  
  
import java.net.URL;  
import java.util.ResourceBundle;  
  
import javafx.event.ActionEvent;  
import javafx.event.EventHandler;  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.Button;  
import javafx.scene.control.TextField;  
import javafx.scene.input.MouseEvent;  
  
public class GUICalculatorPerimetr extends CalculatorPerimetr implements Initializable {  
 @Override  
 public int Matematica(int h, int side) {  
 return super.Matematica(h, side);  
 }  
  
  
 @FXML  
 private ResourceBundle resources;  
  
 @FXML  
 private URL location;  
  
 @FXML  
 private TextField side\_field;  
  
 @FXML  
 private TextField high\_field;  
  
 @FXML  
 private TextField res\_field;  
  
 @FXML  
 private Button calc\_button;  
  
 @FXML  
 void initialize() {  
 calc\_button.addEventHandler(MouseEvent.*MOUSE\_CLICKED*, new EventHandler<MouseEvent>() {  
 @FXML  
 public void handle(MouseEvent mouseEvent) {  
 calc\_button.setText("Thanks!");  
 System.*out*.println("Hello");  
 }  
 });  
 }  
@FXML  
private void handleButtonAction(ActionEvent event)  
{  
 int h=Integer.*parseInt*(high\_field.getText());  
 int s=Integer.*parseInt*( side\_field.getText());  
 res\_field.setText(String.*valueOf*( Matematica(h,s)));  
}  
  
 @Override  
 public void initialize(URL location, ResourceBundle resources) {  
  
 }  
}

2.3.1 Файл*sample.fxml*

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.shape.Line?>  
<?import javafx.scene.shape.Rectangle?>  
<?import javafx.scene.text.Font?>  
  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="699.0" style="-fx-background-color: #8E3299;" xmlns="http://javafx.com/javafx" xmlns:fx="http://javafx.com/fxml" fx:controller="sample.GUICalculatorPerimetr">  
 <children>  
 <AnchorPane layoutX="-1.0" layoutY="78.0" prefHeight="322.0" prefWidth="699.0" style="-fx-background-color: #fafafa;">  
 <children>  
 <Line endX="100.0" layoutX="362.0" layoutY="33.0" startX="77.5" startY="50.5" />  
 <Line endX="97.70709228515625" endY="-1.70709228515625" layoutX="365.0" layoutY="34.0" startX="123.5" startY="49.5" />  
 <Line endX="127.0" endY="51.0" layoutX="361.0" layoutY="33.0" startX="78.0" startY="51.0" />  
 <Rectangle arcHeight="5.0" arcWidth="5.0" fill="WHITE" height="36.0" layoutX="553.0" layoutY="40.0" stroke="BLACK" strokeType="INSIDE" width="75.0" />  
 <TextField fx:id="side\_field" layoutX="83.0" layoutY="93.0" />  
 <TextField fx:id="high\_field" layoutX="83.0" layoutY="45.0" />  
 <Label layoutX="27.0" layoutY="49.0" prefHeight="18.0" prefWidth="57.0" text=" High :">  
 <font>  
 <Font name="Verdana" size="12.0" />  
 </font>  
 </Label>  
 <Label layoutX="30.0" layoutY="97.0" prefHeight="17.0" prefWidth="42.0" text="Side :">  
 <font>  
 <Font name="Verdana" size="12.0" />  
 </font>  
 </Label>  
 <TextField fx:id="res\_field" layoutX="83.0" layoutY="149.0" />  
 <Label layoutX="20.0" layoutY="153.0" prefHeight="17.0" prefWidth="50.0" text="Result :">  
 <font>  
 <Font name="Verdana" size="12.0" />  
 </font>  
 </Label>  
 <Button id="calc\_button" layoutX="253.0" layoutY="149.0" mnemonicParsing="false" prefHeight="25.0" prefWidth="50.0" text="Calc" onAction="#handleButtonAction" />  
 </children>  
 </AnchorPane>  
 <Label layoutX="38.0" layoutY="14.0" prefHeight="57.0" prefWidth="635.0" style="-fx-background-color: #8E3299;" text="Sum Perimeters of rectangular and triangle" textFill="#e0ff15">  
 <font>  
 <Font name="Tw Cen MT Condensed" size="45.0" />  
 </font>  
 </Label>  
 </children>  
</AnchorPane>

2.3.1 Файл*Calculator.java*

import java.util.ArrayList;

*/\*\**

*\* The interface Calculator.*

*\*/*

interface Calculator {

*/\*\**

*\* The constant arrayList.*

*\*/*

public ArrayList<Integer> *arrayList*=new ArrayList<Integer>();

*/\*\**

*\* Matematica int.*

*\**

*\** ***@param*** *h the h*

*\** ***@param*** *side the side*

*\** ***@return*** *the int*

*\*/*

public int Matematica( int h,int side);

}

2.3.1 Файл  *GUICalculator.java*

public class GuiCalculatorPerimetr extends CalculatorPerimetr {  
  
  
 @Override  
 public int Matematica(int h, int side) {  
 return super.Matematica(h, side);  
 }  
  
}

2.3.1 Файл *CalculatorFactory.java*

public interface CalculatorFactory {

public Calculator CreateCalculator();

}

2.3.1 Файл *CalculatorPerimetr.java*

import java.io.Serializable;

import java.lang.reflect.Array;

import java.util.ArrayList;

*/\*\**

*\* The type Calculator.*

*\*/*

public class CalculatorPerimetr extends UniversalOperations implements Serializable ,Calculator{

public ArrayList<Integer> arrayList=new ArrayList<Integer>();

*/\*\**

*\* Matematica int.*

*\**

*\** ***@param*** *h the h*

*\** ***@param*** *side the side*

*\** ***@return*** *the int*

*\*/*

public int Matematica( int h,int side)

{

if (h<0||side<0)

{

h=0;

side=0;

}

double rectangular=(h\*2)+(side\*2);

double triangle=(Math.*sqrt*(Math.*pow*((side/2),2)+Math.*pow*(h,2)))\*2+side;

double sum= rectangular+triangle;

String s=Double.*toString*(sum);

s= s.substring(0,s.indexOf('.'));

Integer res=Integer.*parseInt*(s);

arrayList.add(res);

return res ;

}

}

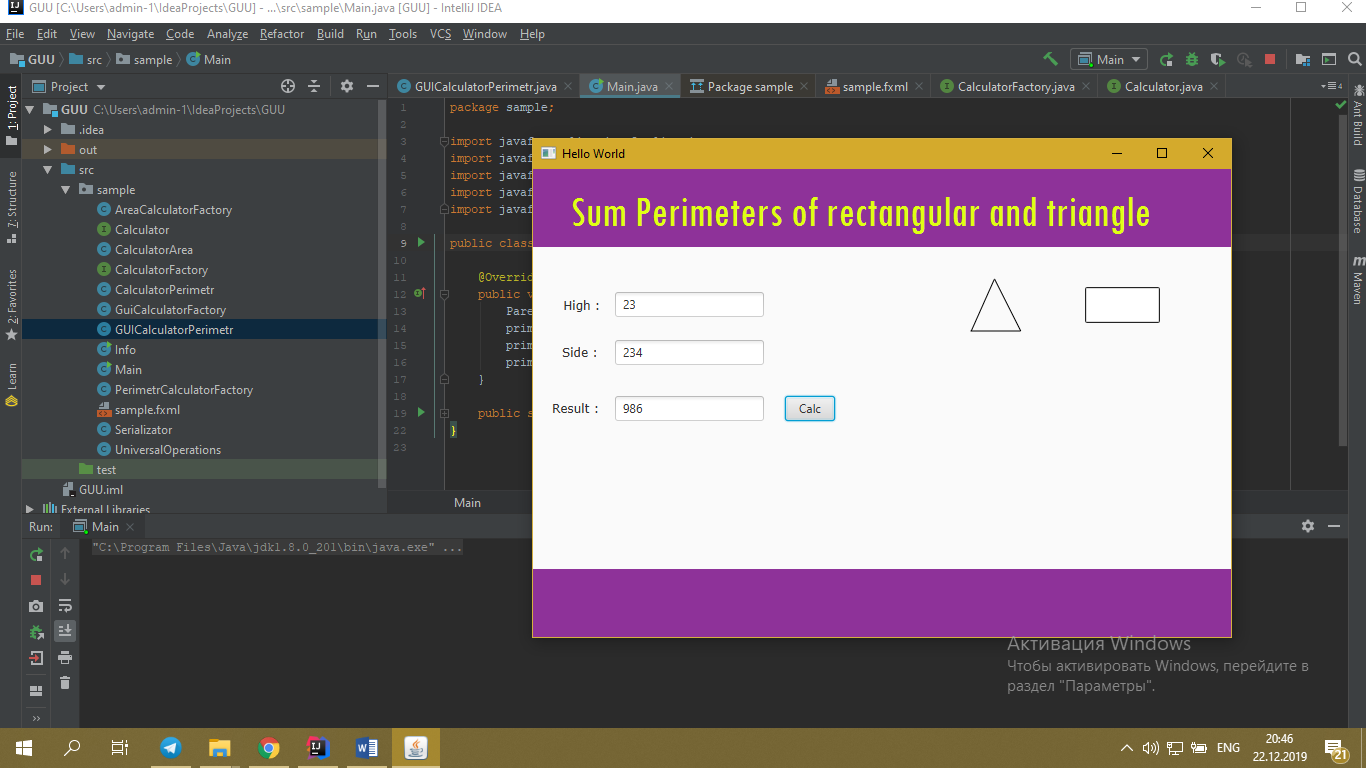
2.3.1 Файл *CalculatorTest.java*

import org.junit.Assert;  
import org.junit.Test;  
  
*/\*\*  
 \* The type Test calculator.  
 \*/*public class TestCalculator {  
 */\*\*  
 \* Test mat.  
 \*/* @Test  
 public void TestMat() {  
 int result=0;  
 Calculator calculator =new Calculator();  
  
  
 result =calculator.Matematica(8,12);  
 Assert.*assertEquals*(72,result);  
  
  
 result =calculator.Matematica(8,6);  
 Assert.*assertEquals*(51,result);  
  
  
 result =calculator.Matematica(0,0);  
 Assert.*assertEquals*(0,result);  
  
  
 result =calculator.Matematica(-1,-8);  
 Assert.*assertEquals*(0,result);  
 }  
  
 */\*\*  
 \* Test count ones.  
 \*/* @Test  
 public void TestCountOnes() {  
 String result;  
 Calculator calculator =new Calculator();  
  
  
  
 result= calculator.CountOnes(72);  
 Assert.*assertEquals*("1001000",result);  
  
  
  
 result= calculator.CountOnes(0);  
 Assert.*assertEquals*("",result);  
  
  
  
 result= calculator.CountOnes(1);  
 Assert.*assertEquals*("1",result);  
  
  
 result= calculator.CountOnes(7);  
 Assert.*assertEquals*("111",result);  
  
  
 result= calculator.CountOnes(1000);  
 Assert.*assertEquals*("1111101000",result);  
  
 result= calculator.CountOnes(2423);  
 Assert.*assertEquals*("100101110111",result);  
  
 }  
  
 */\*\*  
 \* Test count.  
 \*/* @Test  
 public void TestCount()  
 {  
 int result;  
 Calculator calculator =new Calculator();  
  
  
 result =calculator.Count("1111");  
 Assert.*assertEquals*(4,result);  
  
  
  
 result =calculator.Count("1110");  
 Assert.*assertEquals*(3,result);  
  
 result =calculator.Count("1010");  
 Assert.*assertEquals*(2,result);  
  
 result =calculator.Count("0010");  
 Assert.*assertEquals*(1,result);  
  
 result =calculator.Count("0");  
 Assert.*assertEquals*(0,result);  
  
 result =calculator.Count("1");  
 Assert.*assertEquals*(1,result);  
  
  
 result =calculator.Count("1-+ 010");  
 Assert.*assertEquals*(2,result);  
 }  
  
}

2.3.1 Файл *SerializatorTest.java*

import org.junit.Assert;  
import org.junit.Test;  
  
*/\*\*  
 \* The type Test serializator.  
 \*/*public class TestSerializator {  
  
 */\*\*  
 \* Test serialization.  
 \*/* @Test  
 public void TestSerialization()  
 {  
 boolean oper;  
 Calculator calculator=new Calculator();  
  
 Serializator serializator=new Serializator(calculator);  
 oper= serializator.serialization();  
 Assert.*assertEquals*(true,oper);  
 System.*out*.println(calculator.Count(calculator.CountOnes(calculator.Matematica(8, 12))));  
  
  
 Calculator calculator1 =new Calculator();  
 calculator1= serializator.deserialization();  
 int i;  
 i= calculator1.getCount();  
 Assert.*assertEquals*(2,i);  
 }  
  
  
}

**3 РЕЗУЛЬТАТИ**



**ВИСНОВКИ**

Навчились працювати тестами та шаблонами а також JavaFX.