

Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

ОТЧЕТ
Лабораторная работа №3

Выполнил:
Студент 4 курса
Группы АС-50
Барболин М.О.
Проверил:
Крощенко А.А.

Брест 2020

Вариант 1.

Задание.

1.

Реализовать простой класс.

Требования к выполнению

- Реализовать пользовательский класс по варианту.
- Создать другой класс с методом `main`, в котором будут находиться примеры использования

пользовательского класса.

Для каждого класса

- Создать поля классов
- Создать методы классов
- Добавьте необходимые `get` и `set` методы (по необходимости)
- Укажите соответствующие модификаторы видимости
- Добавьте конструкторы
- Переопределить методы `toString()` и `equals()`

- 1) Равнобедренный треугольник, заданный длинами сторон – Предусмотреть возможность определения площади и периметра, а так же логический метод, определяющий существует или такой треугольник. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализовать метод `equals`, выполняющий сравнение объектов данного типа.

Код программы:

```
//package com.company;
class Triangle
{
    //инициализировали стороны треугольника. Т.к. он равнобедренный, то две
    стороны (А и В) равны
    double sideAB, sideC;
    //задали начальные значения
    public Triangle()
    {
        this.sideAB = 5;
        this.sideC = 8;
    }

    public Triangle(double sideAB, double sideC)
    {
        this.sideAB = sideAB;
        this.sideC = sideC;
    }

    //задаем и получаем стороны АВ
    public void setSideAB(double sideAB)
```

```

    {
        this.sideAB = sideAB;
    }
    public double getSideAB()
    {
        return this.sideAB;
    }
    //задаем и получаем стороны C
    public void setSideC(double sideC)
    {
        this.sideC = sideC;
    }
    public double getSideC()
    {
        return this.sideC;
    }
    //нахождение площади и периметра, варианты с заданными сторонами или
    сразу по треугольнику
    public double perimeter()
    {
        return ((sideAB * 2) + sideC);
    }
    public double perimeter(double sideAB, double sideC)
    {
        return ((sideAB * 2) + sideC);
    }
    //площадь = 1/2 основания * высота. высота = корень из (гипотенуза^2 +
    катет^2)
    public double square()
    {
        return (sideC * Math.sqrt(Math.pow(sideAB, 2) - Math.pow((sideC / 2),
2)) / 2);
    }
    public double square(double sideAB, double sideC)
    {
        return (sideC * Math.sqrt(Math.pow(sideAB, 2) - Math.pow((sideC / 2),
2)) / 2);
    }
    //проверка существования
    public boolean isCanBe()
    {
        return (sideAB * 2 >= sideC && sideAB + sideC >= sideAB);
    }
    //проверка идентичности
    public boolean isIdentity(Triangle triangle)
    {
        return (sideAB == triangle.sideAB && sideC == triangle.sideC);
    }
    //вывод информации по сторонам треугольника
    public String showParams()
    {
        return ("Sides of triangle are:\nA = "+sideAB+"\nB = "+sideAB+"\nC =
"+sideC+"\n");
    }
}
//демонстрация использования класса треугольник
public class task1 {
    public static void main(String[] args) {
        Triangle trial_triangle_1 = new Triangle();
        Triangle trial_triangle_2 = new Triangle(3,4);
        System.out.println(trial_triangle_1.showParams());
        double changeSideC = 9;
        trial_triangle_1.setSideC(changeSideC);
        System.out.println("Let's see the new side in tr1:

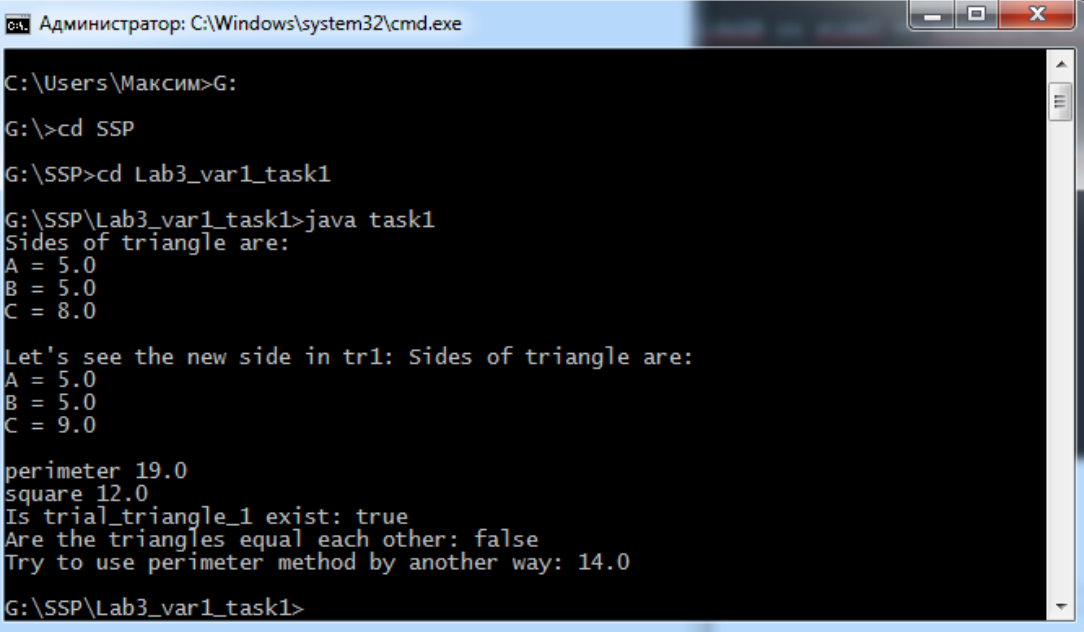
```

```

"+trial_triangle_1.showParams());
    System.out.println("perimeter "+trial_triangle_1.perimeter());
    System.out.println("square "+trial_triangle_2.square());
    System.out.println("Is trial_triangle_1 exist: " +
trial_triangle_1.isCanBe());
    System.out.println("Are the triangles equal each other:
"+trial_triangle_1.isIdentity(trial_triangle_2));
    System.out.println("Try to use perimeter method by another way: "+
trial_triangle_2.perimeter(4,6));
    }
}

```

Скриншот:



```

Администратор: C:\Windows\system32\cmd.exe

C:\Users\Максим>G:
G:\>cd SSP
G:\SSP>cd Lab3_var1_task1
G:\SSP\Lab3_var1_task1>java task1
Sides of triangle are:
A = 5.0
B = 5.0
C = 8.0

Let's see the new side in tr1: Sides of triangle are:
A = 5.0
B = 5.0
C = 9.0

perimeter 19.0
square 12.0
Is trial_triangle_1 exist: true
Are the triangles equal each other: false
Try to use perimeter method by another way: 14.0
G:\SSP\Lab3_var1_task1>

```

2.

Разработать автоматизированную систему на основе некоторой структуры данных, манипулирующей объектами пользовательского класса.

Реализовать требуемые функции обработки данных

Требования к выполнению

- Задание посвящено написанию классов, решающих определенную задачу автоматизации;
- Данные для программы загружаются из файла (формат произволен).
Файл создать и написать

вручную.

1) Стековый калькулятор. Написать стековый калькулятор, который принимает в качестве аргумента командой строки имя файла,

содержащего команды. Если аргумента нет, то использовать стандартный поток ввода для чтения команд. Для вычислений допускается использовать вещественные числа.

Реализовать следующий набор команд:

- # – строка с комментарием.
- POP , PUSH – снять/положить число со/на стек(a).
- + , - , * , / , SQRT – арифметические операции. Используют один или два верхних элемента

стека, изымают их из стека, помещая результат назад

- PRINT – печать верхнего элемента стека (без удаления).
- DEFINE – задать значение параметра. В дальнейшем везде использовать вместо параметра

это значение.

Содержимое стека и список определенных именованных параметров передавать команде в виде специального объекта – контекста исполнения. Разработать группу классов исключений, которые будут выбрасывать команды при исполнении. В случае возникновения исключения – выводить информацию об ошибке и продолжать исполнение программы (из файла или команд вводимых с консоли).

Код программы:

```
package com.company;
import java.io.*;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.util.Scanner;
import java.util.Stack;

//создадим несколько исключений
class StackIsEmptyException extends Exception{
}
//если запрещенные операции с нулем
class DivisionByNullException extends ArithmeticException{
}
//попытка ввода несуществующей операции
class InvalidOperationException extends Exception{
}

//создадим непосредственно класс калькулятора
class Calculator //extends CalcStack
{
    CalcStack myStack = new CalcStack();
    Double x1 = 0.0, x2 = 0.0, res = 0.0;
    //метод берет 2 верхних числа из стека, складывает их и возвращает число
    в стек
```

```

        public void plus() {
            x1 = myStack.getNumbFromStack();
            x2 = myStack.getNumbFromStack();
            res = x1 + x2;
            myStack.addNumbToStack(res);
        }
        //минус
        public void minus()
        {
            x1 = myStack.getNumbFromStack();
            x2 = myStack.getNumbFromStack();
            res = x1 - x2;
            myStack.addNumbToStack(res);
        }

        //умножение
        public void multiply()
        {
            x1 = myStack.getNumbFromStack();
            x2 = myStack.getNumbFromStack();
            res = x1 * x2;
            myStack.addNumbToStack(res);
        }
        //деление
        public void division()
        {
            x1 = myStack.getNumbFromStack();
            x2 = myStack.getNumbFromStack();
            res = x2 / x1;
            myStack.addNumbToStack(res);
        }
        //корень
        public void sqrt()
        {
            x1 = myStack.getNumbFromStack();
            res = Math.sqrt(x1);
            myStack.addNumbToStack(res);
        }
        //конец класса
    }
    //определим отдельно наследуемый класс, содержащий методы относящиеся только
    с стеку
    class CalcStack{
        public Stack<Double> calcStack = new Stack<Double>();
        //методы:
        //положить число в стек
        public void addNumbToStack(double number) {
            calcStack.push(number);
        }
        //достать число из стека (удаляет при доставании)
        public Double getNumbFromStack() {
            return calcStack.pop();
        }
        //просмотреть верхнее число, не удаляя его из стека
        public Double LookTopNumb() {
            return calcStack.peek();
        }
        //конец класса
    }

    //демонстрация работы программы - калькулятор
    public class Main {

```

```

        public static void main(String[] args) throws Exception,
ArithmeticException{
            Scanner scr = new Scanner(System.in);
            System.out.println("Вы собираетесь работать с калькулятором через
файл или вручную?" +
                "\n1-Через файл" +
                "\n2-Вручную");
            int fileOrHandle = scr.nextInt();
            BufferedReader reader = null;
            FileReader fr = null;
            if(fileOrHandle == 1) {
                System.out.println("Вы выбрали работу через файл. Пожалуйста,
введите имя файла:");
                String path = scr.next();
                try{
                    fr = new FileReader(path);
                }
                catch (FileNotFoundException e){
                    System.out.println("Ошибка доступа к файлу");
                }
                if(fr != null) {
                    reader = new BufferedReader(fr);
                }
                else System.out.println("Path is null");
            }
            Calculator myCalc = new Calculator();
            boolean isRun = true;
            String variable = "";
            while (isRun || (reader.readLine() != null)) {
                //менюинт
                if(fileOrHandle != 1) {
                    System.out.println("\nВыберите пункт меню:\n" +
                        "1 - Добавить число в стек\n" +
                        "2 - Удалить число из стека\n" +
                        "3 - Показать верхнее число в стеке\n" +
                        "4 - Выполнить арифметическую операцию\n" +
                        "5 - Выход из программы\n" +
                        "Введите пункт меню:\n");
                }
                if (fileOrHandle == 1) {
                    variable = reader.readLine();
                } else {
                    variable = scr.next();
                }
                switch (variable) {
                    //добавление числа в стек для дальнейших вычислений
                    case "1": {
                        Double number;
                        if(fileOrHandle == 1){
                            number = Double.parseDouble(reader.readLine());
                        }
                        else {
                            System.out.println("Введите число: (типа 4,3 или
21)");
                            number = scr.nextDouble();
                        }
                        myCalc.myStack.addNumbToStack(number);
                        break;
                    }
                    case "2": {
                        try{
                            double del = myCalc.myStack.getNumFromStack();
                        }
                        catch (Exception e){

```

```

        System.out.println("Ошибка пустой стек");
    }
    break;
}
case "3": {
    try{
        System.out.println("Top number is:" +
myCalc.myStack.LookTopNumb());
    }
    catch (Exception e){
        System.out.println("Ошибка пустой стек");
    }
    break;
}
case "4": {
    String change;
    if(fileOrHandle == 1){
        change = reader.readLine();
    }
    else{
        System.out.println("Введите желаемую операцию: "
+
        "\n+, -, *, /, sqrt");
        change = scr.next();
    }
    switch (change) {
        case "+": {
            try {
                myCalc.plus();
            }
            catch (Exception e){
                System.out.println("Ошибка: недостаточно
операндов в стеке");
            }
            break;
        }
        case "-": {
            try{
                myCalc.minus();
            }
            catch (Exception e){
                System.out.println("Ошибка: недостаточно
операндов в стеке");
            }
            break;
        }
        case "*": {
            try{
                myCalc.multiply();
            }
            catch (Exception e){
                System.out.println("Ошибка: недостаточно
операндов в стеке");
            }
            break;
        }
        case "/": {
            try{
                myCalc.division();
            }
            catch (Exception e){
                System.out.println("Ошибка:
недостаточно операндов в стеке");
            }
        }
    }
}
}

```



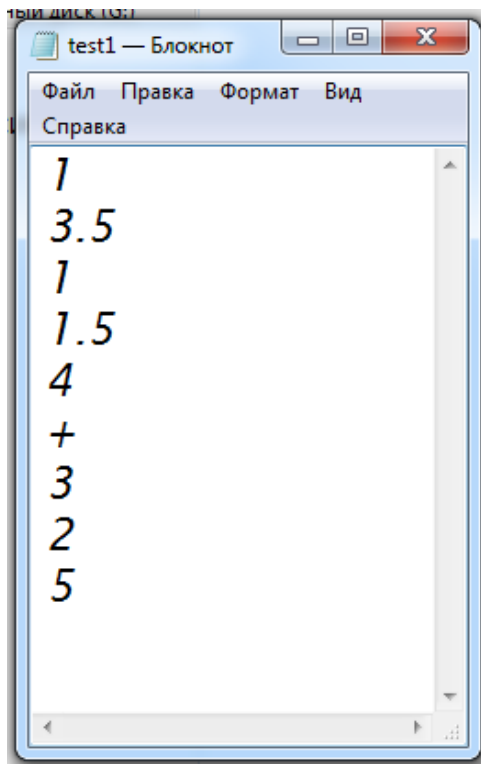
```

        break;
    }
    case "sqrt": {
        try{
            myCalc.sqrt();
        }
        catch (Exception e){
            System.out.println("Ошибка: недостаточно
операндов в стеке");
        }
        break;
    }
    default: {
        System.out.println("Incorrect operation");
        break;
    }
}
break;
}
case "5": {
    isRun = false;
    break;
}
default: {
    System.out.println("Incorrect operation");
    break;
}
//конец оператора switch
}
//конец цикла while
}
//конец метода мейн
}

//конец класса
}

```

Пример заполнения текстового файла:



Скриншоты работы программы:

```
C:\Users\Максим\.jdk\openjdk-15.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\I
Вы собираетесь работать с калькулятором через файл или вручную?
1-Через файл
2-Вручную
1
Вы выбрали работу через файл. Пожалуйста, введите имя файла:
G:\SSP\Lab3_var1_task2\test1.txt
Top number is:5.0

Process finished with exit code 0
|
```

```
C:\Users\Максим\.jdk\openjdk-15.0.1\bin\ja
Вы собираетесь работать с калькулятором чер
1-Через файл
2-Вручную
2

Выберите пункт меню:
1 - Добавить число в стек
2 - Удалить число из стека
3 - Показать верхнее число в стеке
4 - Выполнить арифметическую операцию
5 - Выход из программы
Введите пункт меню:

1
Введите число: (типа 4,3 или 21)
3,6

Выберите пункт меню:
1 - Добавить число в стек
2 - Удалить число из стека
3 - Показать верхнее число в стеке
4 - Выполнить арифметическую операцию
5 - Выход из программы
Введите пункт меню:

1
Введите число: (типа 4,3 или 21)
1,6
```

```
Выберите пункт меню:
1 - Добавить число в стек
2 - Удалить число из стека
3 - Показать верхнее число в стеке
4 - Выполнить арифметическую операцию
5 - Выход из программы
Введите пункт меню:

4
Введите желаемую операцию:
+, -, *, /, sqrt
+

Выберите пункт меню:
1 - Добавить число в стек
2 - Удалить число из стека
3 - Показать верхнее число в стеке
4 - Выполнить арифметическую операцию
5 - Выход из программы
Введите пункт меню:

3
Top number is:5.0
```

```
Выберите пункт меню:
1 - Добавить число в стек
2 - Удалить число из стека
3 - Показать верхнее число в стеке
4 - Выполнить арифметическую операцию
5 - Выход из программы
Введите пункт меню:

4
Введите желаемую операцию:
+, -, *, /, sqrt
-
Ошибка: недостаточно операндов в стеке

Выберите пункт меню:
1 - Добавить число в стек
2 - Удалить число из стека
3 - Показать верхнее число в стеке
4 - Выполнить арифметическую операцию
5 - Выход из программы
Введите пункт меню:

2
Ошибка пустой стек
```

Вывод: в ходе лабораторной работы я изучил работу с файлами, обработку исключений, познакомился с некоторыми типами хранения данных.