Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

|  |
| --- |
| **Отчет** |
| к лабораторной работе №2  по дисциплине «ООП»  «Разработка классов и их использование» |
| наименование темы  Вариант №1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ЭВМб-19-2 |  |  |  | А. А. Бодров |
|  |  | шифр |  | подпись |  | И.О. Фамилия |
| Проверил |  |  |  |  |  | Т.В. Маланова |
|  |  | должность |  | подпись |  | И.О. Фамилия |
| Работа защищена с оценкой | | | |  | | |

Иркутск 2020 г.

# **1 Постановка задачи**

Провести объектную декомпозицию задачи из индивидуального задания и разработать класс, содержащий соответствующие поля для хранения необходимых данных и методы, обеспечивающие достаточную для решения задачи функциональность класса. Создание объекта класса решающего задание и вызов его методов должны осуществляться из главного класса. Ввод и вывод данных осуществляется как в предыдущей лабораторной работе, для передачи исходных данных экземпляру класса, решающему задачу, должны быть разработаны соответствующие методы

**2 Индивидуальное задание:**

a. Вычислите сумму диагоналей квадратной матрицы.

b. Определите, является ли заданная строка символов числом.

c. Выполнить сложение двух матриц.

# 3 Структура классов

Addition

StringNew

Matrix

JavaApplication2

(Рисунок 1 – Описание структуры классов)

*Описание классов*

а) класс **Program** – главный класс, метод **main** которого используется в качестве точки входа в программу. Данный метод предоставляет интерфейс взаимодействия с пользователем и содержит в себе решение поставленных задач (индивидуальные задания 1-3).

б) класс **Matrix** – класс-обертка, предоставляющий дополнительные методы для работы с квадратной матрицей целых чисел и обеспечивающий решение 1-го индивидуального задания.

в) класс **StringNew** - класс-обертка, предоставляющий дополнительные методы для работы с массивом строк и обеспечивающий решение 2-го индивидуального задания.

г) класс **Addition** – класс-обертка, предоставляющий дополнительные методы для работы с квадратной матрицей целых чисел и обеспечивающий решение 3-го индивидуального задания

4 Спецификация

Таблица 1 – Описание полей классов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Имя | Назначение | Тип данных | Тип доступа | ОДЗ |
| Класс Matrix | | | | | |
| 1 | matrixA | Квадратная матрица | int [][] | private | [-231:231-1] |
| 2 | size | Размерность матрицы | int | private | [-231:231-1] |
|  | | | | | |
| Класс StringNew | | | | | |
| 1 | s1 | Массив букв | String [] | private | ‘A’ .. ‘z’,  ‘А’ .. ‘я’,  ‘0’ .. ‘9’, |
| Класс Additition | | | | | |
| 1 | size | Размерность матрицы | int | private | [-231:231-1] |
| 2 | matrixA | Квадратная матрица | int [][] | private | [-231:231-1] |

Таблица 2 – Описание методов класса Matrix

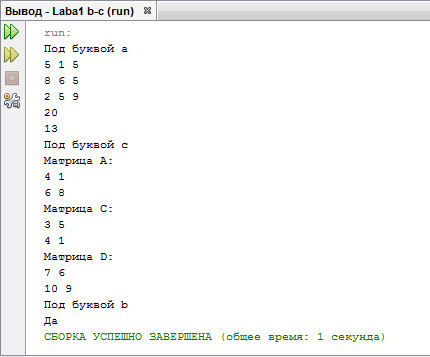
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Имя | Назначение | Тип данных | Тип доступа | Параметры |
| 1 | getElement | Получаем элемент матрицы | int[][] | public | int i – *индекс строки* |
| int j – *индекс столбца* |
| 2 | setElement | Устанавливаем значение в элемент | int[][] | public | int i – *индекс строки* |
| int j – *индекс столбца* |
| value – *добавляемый новый элемент в матрицу* |
| 3 | Matrix | Конструктор по умолчанию | - | public | - |
| 4 | Matrix | Конструктор с параметрами | - | public | *int size* – размерность матрицы |
| 5 | Matrix | Конструктор копирования | - | public | *Matrix kopirovan –* объект копирования. |
| 6 | sum1 | Нахождение суммы диагоналей | int | public | *-* |
| 7 | sum2 | Нахождение суммы второй дигонали | int | public | *-* |

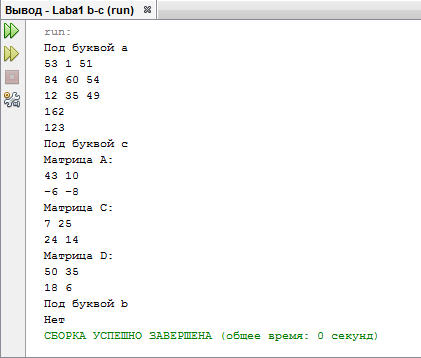
Таблица 3 – Описание методов класса StringNew

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Имя | Назначение | Тип данных | Тип доступа | Параметры |
| 1 | StringNew | Конструктор по умолчанию | - | public | - |
| 2 | StringNew | Конструктор с параметрами | - | public | String s1 – Данная строка |
| 3 | StringNew | Конструктор копирования | - | public | StringNew stringNew2 – объект копирования |
| 4 | isDigitalSrting | Проверка на символы | boolian | public | - |

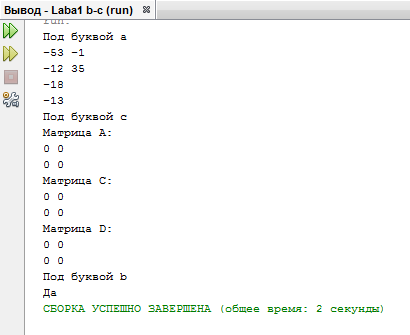
Таблица 4 – Описание методов класса Additition

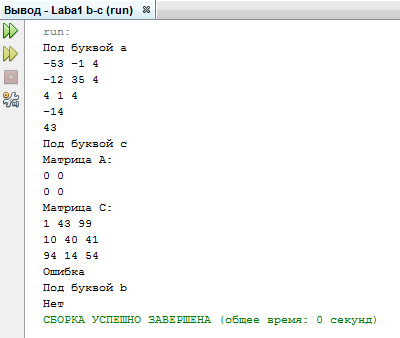
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Имя | Назначение | Тип данных | Тип доступа | Параметры |
| 1 | getElement | Получаем элемент матрицы | int[][] | public | int i – *индекс строки* |
| int j – *индекс столбца* |
| 2 | setElement | Устанавливаем значение в элемент | int[][] | public | int i – *индекс строки* |
| int j – *индекс столбца* |
| value – *добавляемый новый элемент в матрицу* |
| 3 | Addition | Конструктор по умолчанию | - | public | - |
| 4 | Addition | Конструктор с параметрами | int[][] | public | *int size* – размерность матрицы |
| 5 | Addition | Конструктор копирования | int[][] | public | *Addition addcopirov – копируемый объект* |
| 6 | sumTwoMatrix | Нахождение суммы матрицы | Int[][] | public | *int [][] matrix – прибавляемая матрица* |

**5 Результаты тестирования**

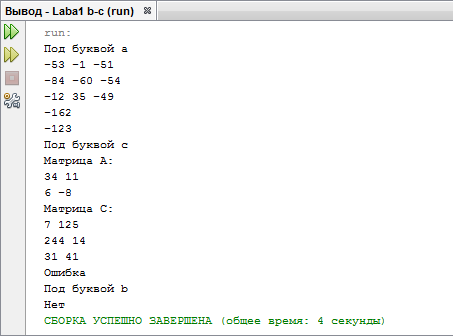
 **(**Рисунок 2 – Тестирование программы**)**

**(**Рисунок 3 – Тестирование программы**)**

**(**Рисунок 4 – Тестирование программы**)**



**(**Рисунок 5 – Тестирование программы**)**

**(**Рисунок 6 – Тестирование программы**)**

**6 Таблица тестов**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Буква a | | | Буква b | | Буква c | | |
| № | Матрица | Сумма главной диагонали | Сумма второй диагонали | Строка | Результат | Матрица A | Матрица C | Матрица D |
| 1 | 5 1 5  8 6 5  2 5 9 | 20 | 13 | “-123” | “Да” | 4 1  6 8 | 3 5  4 1 | 7 6  10 9 |
| 2 | 53 1 51  84 60 54  12 35 49 | 162 | 123 | “-123-“ | “Нет” | 43 10  -6 -8 | 7 25  24 14 | 50 35  18 6 |
| 3 | -53 -1 -81  -84 -60 -54  -12 35 -49 | -162 | -123 | “12с3” | “Нет” | 34 11  6 -8 | 7 126  244 14  31 41 | “Ошибка” |
| 4 | -53 -1  -12 35 | -18 | -13 | “3201” | “Да” | 0 0  0 0 | 0 0  0 0 | 0 0  0 0 |
| 5 | -53 -1 4  -12 35 4  4 1 4 | -14 | 43 | “4-21” | “Нет” | 0 0  0 0 | 1 43 99  10 40 41  94 14 54 | “Ошибка” |

(Таблица 2 - Тестирование)

# 7 Листинг программы

Класс JavaApplication2

package javaapplication2;

import java.util.Arrays;

import java.util.Scanner;//Подключаем библиоте для ввода данных

//Scanner - Класс, преднозначенный для потокового ввода данных с клавиатуры.

public class JavaApplication2 {

public static void main(String[] args) {

int size = 2;

Matrix matrix = new Matrix();

System.out.println("Введите значения");

Scanner cin = new Scanner (System.in,"CP1251");//CP1251 - ANSI коддировка

for (int i = 0; i<size;i++)

for (int j = 0; j<size;j++) {

int value = cin.nextInt();

matrix.setElement(i,j,value);//Заполнение матрицы

}

System.out.println("matrix2(Конструктор Копирования)");

Matrix matrix2 = new Matrix(matrix);//Копирую данные из одной матрицы в другую(Создаю коппию объекта)

for (int i = 0; i < size; i++) {

for (int j=0;j<size;j++) {

System.out.print(matrix2.getElement(i, j) + " ");

}

System.out.println();

}

System.out.println("matrix");

for (int i = 0; i < size; i++) {

for (int j=0;j<size;j++) {

System.out.print(matrix.getElement(i, j) + " ");

}

System.out.println();

}

System.out.println("sum1 = " + matrix.sum1());

System.out.println("sum2 = " + matrix.sum2());

//Под буквой b

String s1="-123";

System.out.println(s1);

//Обращение к конструктору с параметром

StringNew string = new StringNew (s1);

//Обращение к конструктору копирования

StringNew string2 = new StringNew (string);

System.out.println("Копированая строка" + s1);

boolean number = string.isDigitalSrting();

System.out.println(number? "Да": "Нет");

//Под буквой c

size = 3;

int [][] matrixA = {{41,10,-75},{14,55,-91},{52,1,-1}};

int [][] matrixB = {{1,43,99},{10,40,41},{94,14,54}};

System.out.println("Матрица А:");

System.out.println(Arrays.deepToString(matrixA));

//Копия матрицы А, созданная с помощью конструктора копирования

System.out.println("Матрица А(Копированная):");

System.out.println(Arrays.deepToString(matrixA));

System.out.println("Матрица B:");

System.out.println(Arrays.deepToString(matrixB));

//обращение к конструктору с параметрами

Addition addition = new Addition(matrixA);

//Обращение к конструктору копирования

Addition matrixA2 = new Addition(addition);

matrixB = addition.sumTwoMatrix(matrixB);

System.out.println("Матрица С");

for (int i=0; i<size;i++)

{

for (int j=0;j<size;j++)

{

System.out.print(matrixB[i][j] + " ");

}

System.out.println();

}

}

}

Класс Matrix

package javaapplication2;

public class Matrix {

//Поля(Переменные класса)

final int size;// Создал внутреннюю переменную(Чтобы ее можно было видеть внутри класса)

final int [][] matrixA;

//получаем элемент матрицы по указанным индексам

public int getElement(int i, int j) {

return matrixA[i][j];

}

//Добавляем новый элемент value в матрицу по указанным индексам i и j

public void setElement(int i, int j, int value) {

if (i < size && j < size)

matrixA[i][j] = value;

}

//Конструктор по умолчанию, который создает новый обхект

public Matrix(){

this(2);// this - вызывает конструктор с параметрами

}

//Конструктор с параметрами

public Matrix(int size){//Конструктор с параметрами

matrixA = new int[size][size];

this.size = size;// Внутренней переменной присвоил значение входного параметра

}

//Конструктор копирования

//Присваиваю новому объекту значения из старого

public Matrix(Matrix kopirovan){

size = kopirovan.size;

matrixA = new int[size][size];

//Передает значения в матрицу нового объекта

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

matrixA[i][j]= kopirovan.matrixA[i][j];

}

}

}

public int sum1()

{

int sum1 = 0;

for(int i = 0; i<size; i++)

for(int j = 0; j<size; j++)

if (i==j)

sum1+=matrixA[i][j];

return sum1;

}

//Метод нахождения суммы диагоналей

public int sum2(){

int sum2 = 0;

for(int i = 0; i<size; i++)

for(int j = 0; j<size; j++)

if(j==size-i-1)

sum2+=matrixA[i][j];

return sum2;

}

}

Класс StringNew

package javaapplication2;

class StringNew {

private final String s1;

//Констроктор по умолчанию

public StringNew(){

this("-123");

}

//Конструктор с параметром

public StringNew(String s1) {

this.s1 = s1;

}

//Конструктор копирования

public StringNew (StringNew stringNew2){

//Вспомогательный класс StringBuilder - работает со строками

StringBuilder sb = new StringBuilder();

for (int i = 0; i < stringNew2.s1.length(); i++) {//Прохожусь по старой строке

sb.append(stringNew2.s1.charAt(i)); //Копирую в Sb

}

//С sb в новую строку

s1 = sb.toString();//Приведение к строке

}

public boolean isDigitalSrting(){

boolean number = true;

if((s1.charAt(0)<48 || s1.charAt(0)>57) && s1.charAt(0) != 45) //CharAt - берет символ по указанному индексу

number = false;

if (number)

{

for(int i=1;i<s1.length();i++)// length - длина строки

{

if (s1.charAt(i)<48 || s1.charAt(i)>57) //charAt - берем отдельный символ

{

number = false;

break;

}

}

}

return number;

}

}

Класс Addition

package javaapplication2;

public class Addition {

private final int size;// Создал внутреннюю переменную(Чтобы ее можно было видеть внутри класса)

private int [][] matrixA;

public Addition(int[][] matrixA) {

this(matrixA.length);

this.matrixA = matrixA;

}

public int getElement(int i, int j){

return matrixA[i][j];

}

public void setElement(int i,int j, int value){

if (i<size && j <size)

matrixA[i][j]=value;

}

//Конструктор по умолчанию

public Addition (){

this(3);

}

//Конструктор с параметрами

public Addition (int size){

matrixA = new int [size][size];

this.size = size;

}

//Конструктор копирования

public Addition(Addition addcopirov){

size=addcopirov.size;

matrixA = new int[size][size];

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

matrixA[i][j] = addcopirov.matrixA[i][j];

}

}

}

public int [][] sumTwoMatrix(int [][] matrixB) {

int [][] matrixC = new int [matrixA.length][matrixA.length]; //Беру у любой матрицы ее размер

if (matrixA.length== matrixB.length)

{

int count=matrixA.length;//Cound - размер матрицы

for (int i=0; i<count;i++)

for (int j=0;j<count;j++)

matrixC[i][j]=matrixA[i][j]+matrixB[i][j];

}

else

System.out.println("Ошибка");

return matrixC;

}

}

# Заключение

В ходе выполнения лабораторной работы я познакомился с интегрированной средой разработки Java-приложений IDE NetBeans.

Для решения поставленной задачи мною было разработано консольное приложение с применением таких элементов объектно-ориентированного подхода, как объектная декомпозиция и инкапсуляция механизмов реализации объектов. Мною были получены практические навыки описания и реализации классов, создания объектов, вызова их методов и работа с модификаторами доступа.

В ходе тестирования программа работает исправно, ошибок не было обнаружено.

**Литература**

1. Учебная практика. Руководство и методические указания для студентов

направления 09.03.02 «Информационные системы и технологии», профили подготовки «Информационные системы и технологии в административном управлении», «Информационные системы и технологии в про-мышленности/машиностроении». Составители: Аршинский В.Л., Серышева И.А – Иркутск, Изд-во ИРНИТУ, 2016, 49с.

2. API Specification for the Java 7 SE. [официальный сайт] URL:

http://docs.oracle.com/javase/7/docs/api/ (дата обращения: 05.09.2020)

3. The Java Tutorials. SE [электронный ресурс], URL:

http://docs.oracle.com/javase/tutorial/index.html

4. Грэди Буч. Объектно-ориентированный анализ и проектирование. Второе

издание, перевод с английского под редакцией И. Романовского и Ф. Андреева. Rational Санта-Клара, Калифорния.

5. Гослинг Д., Арнолд К. и др. Язык программирования Java. Издательский

дом "Питер", 1997-2002.