



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

2020 年春季学期
计算机学院《软件构造》课程

Lab 1 实验报告

姓名	金煥凡
学号	L180300902
班号	1803009
电子邮件	faer9876@naver.com
手机号码	13149506801

目录

1 实验目标概述	1
2 实验环境配置	1
3 实验过程	오류! 책갈피가 정의되어 있지 않습니다.
3.1 Magic Squares	1
3.1.1 isLegalMagicSquare()	1
3.1.2 generateMagicSquare()	4
3.2 Turtle Graphics	5
3.2.1 Problem 1: Clone and import	6
3.2.2 Problem 3: Turtle graphics and drawSquare	6
3.2.3 Problem 5: Drawing polygons	6
3.2.4 Problem 6: Calculating Bearings	6
3.2.5 Problem 7: Convex Hulls	6
3.2.6 Problem 8: Personal art	6
3.2.7 Submitting	6
3.3 Social Network	7
3.3.1 设计/实现 FriendshipGraph 类	7
3.3.2 设计/实现 Person 类	7
3.3.3 设计/实现客户端代码 main()	7
3.3.4 设计/实现测试用例	8
4 实验进度记录	9
5 实验过程中遇到的困难与解决途径	10
6 实验过程中收获的经验、教训、感想	10
6.1 实验过程中收获的经验教训	10
6.2 针对以下方面的感受	10

1 实验目标概述

根据实验手册简要撰写。

本次实验通过求解三个问题, 训练基本 Java 编程技能, 能够利用 Java OO 开发基本的功能模块, 能够阅读理解已有代码框架并根据功能需求补全代码, 能够为所开发的代码编写基本的测试程序并完成测试, 初步保证所开发代码的正确性。另一方面, 利用 Git 作为代码配置管理的工具, 学会 Git 的基本使用方法。基本的 Java OO 编程 基于 Eclipse IDE 进行 Java 编程 基于 JUnit 的测试 基于 Git 的代码配置管理

1.1 Magic Squares

在这里简要概述你对该任务的理解。

n 阶段变换方的配置 $n \times n$ 数字, 通常是不同的整数, 等值中 n 是否输入所有行, 所有列, 对角线及相同常数, main 函数中 isLegal。需要处理输入文件的各种特殊情况, 例如文件中的数据不符合 Magic Square 的定义, 或者队伍中的某些数字不是整数, 数字之间 \t 分段。在这种情况下, 停止运行程序 (isLegalMagicSquare 函数返回 false) 控制中控台输出错误提示信息。如果把发生于 generatorMagicSquare 的 magic square 排除在文件 \src\Pl\txt\6.txt 之外, 输入的 n 是合法的。使用前面已经制作的 isLegalMagicSquare 函数, 在 main 函数中满足该函数新生成的文本文件 6.txt 是 magic square。

1.1.1 isLegalMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

```
package isLegalMagicSquare;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

public class isLegalMagicSquare {

    private static final String FileName = null;

    public static boolean main(String[] args) throws FileNotFoundException,
```

```

IOException {
    int lineNumber = 0; int columnnumber = 0;
    try (FileReader reader = new
FileReader("src/isLegalMagicSquare.P1/txt/" + NameOfFile);
        BufferedReader br = new BufferedReader(reader)) {
        String line;
        int i=0,j=0,k=0;

        while ((line = br.readLine()) != null) {
            j++;
            String[] word = line.split("\t");
            i = word.length;
            if(k<=1)
                k=i;
        }
        lineNumber = j;
        columnnumber = k;
    } catch (IOEExeption e) {
        e.printStackTrace();
    }
    System.out.println(fileName + "lineNumber" + "," +
"columnnumber" );
    if (lineNumber != columnnumber) {
        System.out.println("is not correct");
        return false;
    }

    int matrix[][] = new int [lineNumber][columnnumber];
    int[] and = new int [lineNumber + columnnumber + 2];

    String fileName = null;
    try (FileReader reader = new
FileReader("src/isLegalMagicSquare.P1/txt/" + fileName);
        BufferedReader br = new BufferedReader(reader)) {
        String line;
        int i,j,k;
        j=0;
        i = 0;
        while ((line = br.readLine()) != null) {
            String[] word = line.split("\t");
            for (i = 0; i < word.length; i++) {
                for (k = word[i].length(); --k >= 0;) {
                    if (!Character.isDigit(word[i].charAt(k))) {
                        System.out.println("");
                    }
                }
            }
        }
    }
}

```

```
        return false;
    }
}
matrix[j][i] = Integer.valueOf(word[i]);
}
j++;
}
} catch (IOException e) {
    e.printStackTrace();
}

for (int m = 0; m < linenumber; m++) {
    for (int n = 0; n < columnnumber; n++) {
        if (matrix[m][n] == 0) {
            System.out.println("defalut");
            return false;
        }
    }
}

}

for (int m = 0; m < linenumber; m++) {
    int c = 0;
    for (int n = 0; n < columnnumber; n++) {
        c = c + matrix[m][n];
    }
    and[m] = c;
}

for (int m = 0; m < columnnumber; m++) {
    int c = 0;
    for (int n = 0; n < linenumber; n++) {
        c = c + matrix[n][m];
    }
    and[m + linenumber] = c;
}

int d = 0, p = 0;
for (int m = 0; m < linenumber; m++) {
    d = d + matrix[m][m];
    p = p + matrix[m][columnnumber - m - 1];
}
```

```
        and[linenumber + columnnumber] = d;
        and[linenumber + columnnumber + 1] = p;

        for (int q = 1; q < linenumber + columnnumber + 2; q++) {
            if (and[0] != and[q]) {
                System.out.println("defalut");
                return false;
            }
        }
        System.out.println("correct" + and[0]);
        return true;
    }

    // TODO Auto-generated method stub

}
```

1.1.2 generateMagicSquare()

按步骤给出你的设计和实现思路/过程/结果。

```
package generateMagicSqaure;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.Writer;

public class Main {

    public static void main(String[] args) throws IOException {

        int n = 0;
        int magic[][] = new int[n][n];
        int row = 0, col = n / 2, i, j, square = n * n;

        for (i = 1; i <= square; i++) {
            magic[row][col] = i;
            if (i % n == 0)
                row++;
            else {

```

```
        if (row == 0)
            row = n - 1;
        else
            row--;
        if (col == (n - 1))
            col = 0;
        else
            col++;
    }
}
File file = new File("src/P1/txt/6.txt");
Writer er = new FileWriter(file);
er.close();
Writer out = new FileWriter(file, true);

for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        System.out.print(magic[i][j] + "\t");
        String data = magic[i][j] + "\t";
        out.write(data);
    }
    System.out.println();
    out.write("\n");
}
out.close();
}
}
```

数组创建时元素个数不能为负数

1.2 Turtle Graphics

在这里简要概述你对该任务的理解。

请阅读 <http://web.mit.edu/6.031/www/fa18/psets/ps0/>，遵循该页面内的要求完成编程任务。

在 Problem 1: Clone and import 中你无法连接 MIT 的 didit 服务器，请从 https://github.com/rainywang/Spring2020_HITCS_SC_Lab1/tree/master/P2 获取代码。

忽略 Problem 2: Collaboration policy。

在 Problem 4: Commit and push your work so far 步骤的第 g 步，忽略页面 Lab Manuals for Software Construction Lab-1 Fundamental Java Programming and Testing 6 中涉及 Athena 和 didit 的内容，请使用 git 指令将代码 push 到

你的 GitHub 仓库中。

在页面最后的 Submitting 步骤中, 请同样将你的代码 push 到你的 GitHub Lab1 仓库上。

其他步骤请遵循 MIT 作业页面的要求。

1.2.1 Problem 1: Clone and import

如何从 GitHub 获取该任务的代码、在本地创建 git 仓库、使用 git 管理本地开发。

1.2.2 Problem 3: Turtle graphics and drawSquare

turtle 前进 sideleength, 右转 90 度后循环 4 次就变成正方形

1.2.3 Problem 5: Drawing polygons

多边形外壳的和值为 360 度, 已知变量和变的长短, 利用 $180 * (\text{sides} - 2) / \text{sides}$ / 内阁 turtle 前进 sideleength 数次求援, 右转弯

1.2.4 Problem 6: Calculating Bearings

不断获得一个点和下一个点坐标, 循环 xCords。size - 1 次, n 存储与上面的变变量, m 相应与 y 轴的偏同量, 变调 $n = \text{calculateBearingToPoint}(m, x1, y1, x2, y2)$, $m = m + n$; list

1.2.5 Problem 8: Personal art

```
turtle.color(PenColor.blue);
for (int i = 0; i < 4; i++) {
    turtle.turn(30);
    turtle.front(50);
    turtle.turn(120);
    turtle.front(50);
    turtle.turn(300);
}
```

1.2.6 Submitting

如何通过 Git 提交当前版本到 GitHub 上你的 Lab1 仓库。

1.3 Social Network

在这里简要概述你对该任务的理解。

1.3.1 设计/实现 FriendshipGraph 类

给出你的设计和实现思路/过程/结果。

```
private List nameList = new ArrayList<>(); // 名字列表,邻接表的展
示结构,效率较高。在 List 类型内装入 Person 类型,而且每一个 Person 内有
List 存储每一个人的序号。
private boolean existName(Person name) // 判断是否 graph 中存在
public boolean addVertex(Person name) // 向图中添加成员
public boolean addEdge(Person name1, Person name2) //
加上关系
public int getDistance(Person name1, Person name2) //获得
距离(bfs)
```

1.3.2 设计/实现 Person 类

给出你的设计和实现思路/过程/结果。

```
private String name; // person 的名字
private List id = new ArrayList<>(); // 和该 person 有关的其他 person
序号 list
public String getName() // 获得名字
public List getId() // 为获得有关列表
public boolean existId(int id) // 判断是否关系已存在
public void Setid(int a) // 添加关系
public Person(String name) // person 名字赋值
```

1.3.3 设计/实现客户端代码 main()

给出你的设计和实现思路/过程/结果。

```
public static void main(String[] args) {
    FriendshipGraph graph = new FriendshipGraph();
    Person rachel = new Person("Rachel");
    Person ross = new Person("Ross");
    Person ben = new Person("Ben");
    Person kramer = new Person("Kramer");
    graph.addVertex(rachel);
```

```
graph.addVertex(ross);
graph.addVertex(ben);
graph.addVertex(kramer);
graph.addEdge(rachel, ross);
graph.addEdge(ross, rachel);
graph.addEdge(ross, ben);
graph.addEdge(ben, ross);
System.out.println(graph.getDistance(rachel, ross));
//should print 1
System.out.println(graph.getDistance(rachel, ben));
//should print 2
System.out.println(graph.getDistance(rachel, rachel));
//should print 0
System.out.println(graph.getDistance(rachel, kramer));
//should print -1
}
```

1.3.4 设计/实现测试用例

给出你的设计和实现思路/过程/结果。

```
package P3;
import static org.junit.Assert.assertEquals;
import org.junit.jupiter.api.Test;
public class FriendshipTest {
    FriendshipGraph graph = new FriendshipGraph();
    Person rachel = new Person( "Rachel" );
    Person ross = new Person( "Ross" );
    Person ben = new Person( "Ben" );
    Person kramer = new Person( "Kramer" );
    Person tom = new Person( "Tom" );
    Person jack = new Person( "Jack" );
    @Test
    public void addVertextest() {
        assertEquals(true, graph.addVertex(rachel));
        assertEquals(true, graph.addVertex(ross));
        assertEquals(true, graph.addVertex(ben));
        assertEquals(false, graph.addVertex(rachel));
        assertEquals(false, graph.addVertex(ross));
        assertEquals(false, graph.addVertex(ben));
    }

    @Test
    public void addEdgetest() {
```

```
graph.addVertex(rachel);
graph.addVertex(ross);
graph.addVertex(ben);
graph.addVertex(kramer);
assertEquals(true, graph.addEdge(rachel, ross));
assertEquals(true, graph.addEdge(ross, rachel));
assertEquals(true, graph.addEdge(ross, ben));
assertEquals(true, graph.addEdge(ben, ross));
assertEquals(false, graph.addEdge(jack, ross));
assertEquals(false, graph.addEdge(ben, tom));
assertEquals(false, graph.addEdge(jack, tom));
}

@Test
public void getDistancetest() {
    graph.addVertex(rachel);
    graph.addVertex(ross);
    graph.addVertex(ben);
    graph.addVertex(kramer);
    graph.addEdge(rachel, ross);
    graph.addEdge(ross, rachel);
    graph.addEdge(ross, ben);
    graph.addEdge(ben, ross);
    assertEquals(1, graph.getDistance(rachel, ross), 0.001);
    assertEquals(2, graph.getDistance(rachel, ben), 0.001);
    assertEquals(0, graph.getDistance(rachel, rachel), 0.001);
    assertEquals(-1, graph.getDistance(rachel, kramer), 0.001);
    assertEquals(-2, graph.getDistance(tom, rachel), 0.001);
    assertEquals(-2, graph.getDistance(rachel, jack), 0.001);
    assertEquals(-2, graph.getDistance(tom, jack), 0.001);
}
```

2 实验进度记录

请使用表格方式记录你的进度情况，以超过半小时的连续编程时间为一行。

每次结束编程时，请向该表格中增加一行。不要事后胡乱填写。

不要嫌烦，该表格可帮助你汇总你在每个任务上付出的时间和精力，发现自己不擅长的任务，后续有意识的弥补。

日期	时间段	任务	实际完成情况
2020-02-25	14:00-15:30	编写问题 1 的 <code>isLegalMagicSquare</code> 函数并进行测试	按计划完成
			延期 1 小时完成
			遇到困难，未完成

3 实验过程中遇到的困难与解决途径

遇到的难点	解决途径

4 实验过程中收获的经验、教训、感想

4.1 实验过程中收获的经验教训

4.2 针对以下方面的感受

- (1) Java 编程语言是否对你的口味?
- (2) 关于 Eclipse IDE
- (3) 关于 Git 和 GitHub
- (4) 关于 CMU 和 MIT 的作业
- (5) 关于本实验的工作量、难度、deadline
- (6) 关于初接触“软件构造”课程