**程序设计竞赛初级结课报告**

学生姓名：吴嘉皓，周浩宇

班级：计算机1802，计算机1801

学号：201809000222

201809000132

**实验三**

1. **实验内容**

给定一个高度为 n 的“数字三角形”，其中第 i 行（1<=i<=n）有 i 个数。（例子如下图所示的一个五行数字三角形）

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

初始时，在“数字三角形”的顶部，即第一行的唯一一个数上。每次移动，可以选择移动到当前位置正下方或者当前位置右下方的位置上。即如果在 (i,j)（表示你在第i行从左往右数第j个数上，下同），你可以选择移动到 (i+1,j) 或 (i+1,j+1)。

经过的所有位置（包括起点和终点）的数字总和最大。求这个最大值。

1. **解题思路**

由于递归的方法解题对于程序运行耗时太多，所以本题采用动态规划法，解题过程中，我使用了两个二维数组a,b，数组a用于储存，数组b用来比较,先将数组b全部赋值为0，由数组a储存输入的动态三角形，将数组a的最后一行赋值给数组b，通过边比较边储存的方法，最终得到的数组b最顶端元素的值恰好是最大值。

1. **算法**

import java.util.Scanner;

public class Shusan {

public static void main(String[] args) {

Scanner in=new Scanner(System.in);

System.out.println("请输入数字三角形的行数");

int n=in.nextInt();

int [][]b=new int [n][n];

int max=0;

int [][]a=new int[n][n];

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

System.out.println("请输入数字三角形第"+(i+1)+"行的数据信息");

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

b[i][j]=0;

a[i][j]=in.nextInt();

if(i==n-1) {

b[i][j]=a[i][j];

}

}

}

int i1,t2;

for(int i=n-2;i>=0;i--) {

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

if(b[i+1][j]>b[i+1][j+1]) {

b[i][j]=a[i][j]+b[i+1][j];

}

else {

b[i][j]=a[i][j]+b[i+1][j+1];

}

}

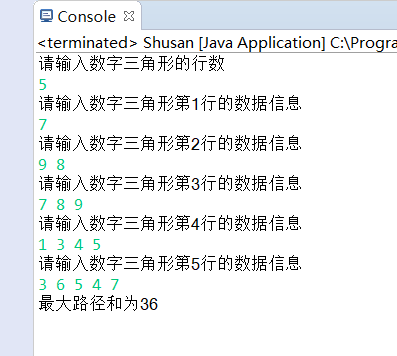
}

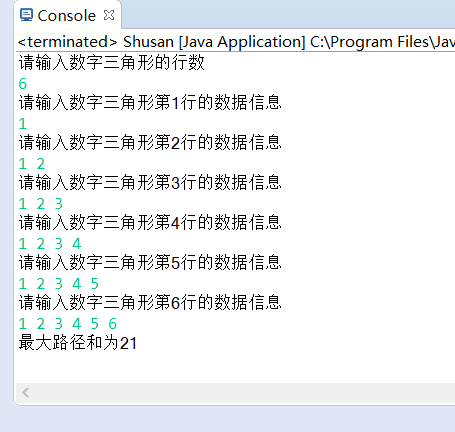
System.out.println("最大路径和为"+b[0][0]);

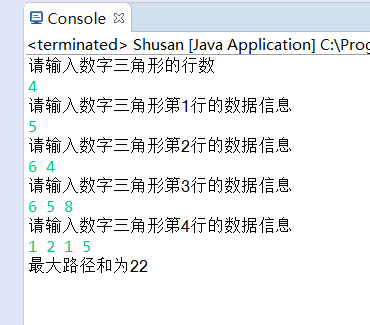
}

}

1. **运行结果**







1. **心得体会**

本次实验中我学会了如何使用动态规划法解数字三角形，数字三角形只是一类题，但是动态规划法却具有很大的意义，这种思路可以在以后的学习或工作中让我多一种可以提高运行速度的方法。在没有学习动态规划法之前，我们只能使用递归的方法或者使用暴力求解法解决这类问题，但是这两种方法都有个共性就是运行时间长而且算法很冗长，但是动态规划法却是在将这种问题处理到了最简，不仅减少了运行时间，还使代码更加简洁，对我们以后的学习有很大的帮助。