

**数学与信息学院学生实验报告**

**实验课程名称：** 算法分析与设计基础 **教师：\_\_**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验项目名称** | **实验一递归算法设计与应用** | | | **实验成绩** |  |
| **学生姓名** |  | **学 号** | **158** | **年级专业班级** |  |
| **小组成员** | **无** | | | **实验日期** | **2019年3月** |

# 1. 实验目的和要求

## 1.1 实验目的

① 加深对**递归算法**的理解，并针对具体问题设计算法；

② 分析算法的复杂性，寻找比较高效的算法，并实现。

③ 分析**格雷码**问题，并设计递归算法求解之。

## 1.2 实验软硬件环境

① 操作系统

WIN10操作系统

②编译环境

Java语言，Eclipse编译器编译

## 1.3 实验要求

① 问题描述

对于给定的正整数n，格雷码为满足如下条件的一个编码序列：

(1) 序列由2n个编码组成，每个编码都是长度为n的二进制位串。

(2) 序列中无相同的编码。

(3) 序列中位置相邻的两个编码恰有一位不同。

例如：n=2时的格雷码为：{00, 01, 11, 10}。

设计求格雷码的递归算法并实现。

② 具体要求

输入：输入的第一行是一个正整数m，表示测试例个数。接下来几行是m个测试例的数据，每个测试例的数据由一个正整数n组成。

输出：对于每个测试例n，输出2n个长度为n的格雷码。（为方便查看，在每个格雷码内，两个位之间用一个空格隔开，如，00输出为：0 0）。两个测试例的输出数据之间用一个空行隔开，最后一个测试例后无空行。

③ 测试数据和结果

输入：2

4

5

输出：0 0 0 0

0 0 0 1

0 0 1 1

0 0 1 0

0 1 1 0

0 1 1 1

0 1 0 1

0 1 0 0

1 1 0 0

1 1 0 1

1 1 1 1

1 1 1 0

1 0 1 0

1 0 1 1

1 0 0 1

1 0 0 0

0 0 0 0 0

0 0 0 0 1

0 0 0 1 1

0 0 0 1 0

0 0 1 1 0

0 0 1 1 1

0 0 1 0 1

0 0 1 0 0

0 1 1 0 0

0 1 1 0 1

0 1 1 1 1

0 1 1 1 0

0 1 0 1 0

0 1 0 1 1

0 1 0 0 1

0 1 0 0 0

1 1 0 0 0

1 1 0 0 1

1 1 0 1 1

1 1 0 1 0

1 1 1 1 0

1 1 1 1 1

1 1 1 0 1

1 1 1 0 0

1 0 1 0 0

1 0 1 0 1

1 0 1 1 1

1 0 1 1 0

1 0 0 1 0

1 0 0 1 1

1 0 0 0 1

1 0 0 0 0

④设计与实现的提示

长度为n的格雷码是由长度为n-1的格雷码变换而成的。

可以用数组或字符串来存储格雷码。注意：对于较大的正整数n，用数组存储容易引起死机。

# 2. 实验记录

## 2.1 格雷码（Gray Code）

### 2.1.1 格雷码

① 概述

在一组数的编码中，若任意两个相邻的代码只有一位二进制数不同，则称这种编码为**格雷码**（Gray Code），另外由于最大数与最小数之间也仅一位数不同，即“首尾相连”，因此又称**循环码**或**反射码**。

### 2.1.2递归（recursion）

① 概述

命程序调用自身的编程技巧称为递归（ recursion）。

② 实现方式

1. 子问题须与原始问题为同样的事，且更为简单；

2. 不能无限制地调用本身，须有个出口，化简为非递归状况处理。

## 2.2 实验过程

### 2.2.1 实验思路

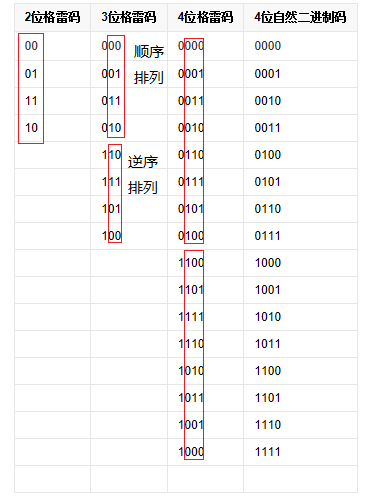


图1 思路发现

### 2.2.2格雷码规则

1.位格雷码有两个码字

2.(n+1)位格雷码中的前2n个码字等于n位格雷码的码字，按顺序书写，加前缀0

3.(n+1)位格雷码中的后2n个码字等于n位格雷码的码字，按逆序书写，加前缀1 [4] 

4.n+1位格雷码的集合 = n位格雷码集合(顺序)加前缀0 + n位格雷码集合(逆序)加前缀1

### 2.2.3程序代码

Graycode.java

**package** Graycode;

**import** java.util.Scanner;

**public** **class** Graycode {

**public** **static** String[] GrayCode(**int** n) {

//定义一个格雷码数组函数

String[] graycode = **new** String[(**int**) Math.*pow*(2, n)];

**if**(n<1) {

System.***out***.println("你输入的格雷码位数有误！");

**return** graycode;

}

**if** (n == 1) {

graycode[0] = "0";

graycode[1] = "1";

**return** graycode;

} //两个出口

//上一位的格雷码

String[] lastGrayCode = *GrayCode*(n - 1);

//在上一位元的顺序列上加一层0,在上一位元的逆序列上加一层1

**for** (**int** i = 0; i < lastGrayCode.length; i++) {

graycode[i] = "0" + lastGrayCode[i]; //加0

graycode[graycode.length - 1 - i] = "1" + lastGrayCode[i];//加1

}

**return** graycode;

}

**public** **static** **void** printGrayCode(**int** n){

String[] graycode=**new** Graycode().*GrayCode*(n);

**for**(**int** i=0;i<graycode.length;i++){

System.***out***.println(graycode[i]);

}

}

**public** **static** **void** main(String[] args) {

Scanner scan = **new** Scanner(System.***in***);

System.***out***.println("输入打印的格雷码个数：");

**int** m = scan.nextInt();

**int**[] n = **new** **int**[m];

System.***out***.println("输入格雷码位数n:");

**for**(**int** k=0;k<m;k++){

n[k]=scan.nextInt();

}

**for**(**int** i = 0 ; i < m ; i ++){

*printGrayCode*(n[i]);

**if**(i<m-1)

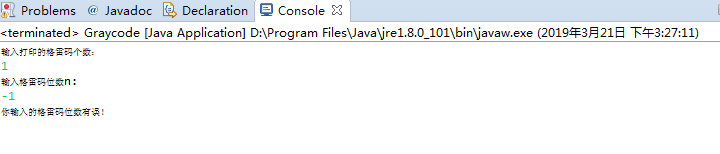
System.***out***.println();

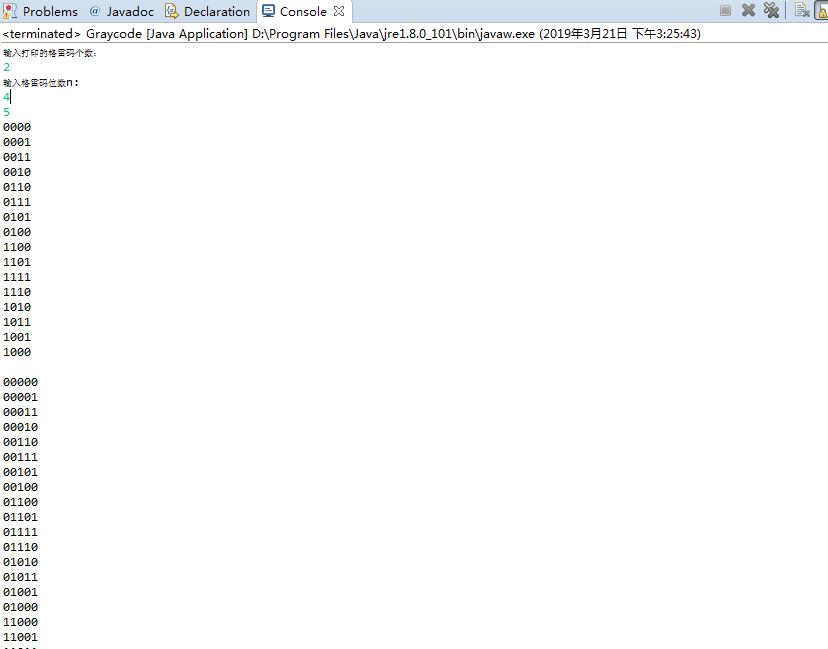
}

}

}

## 2.3 实现结果





顺利输出实验结果，符合实验预期。

# 3.实验总结

这次的实验课开始时，我可能还没做好准备，春节期间的代码量几乎为零，所以刚上课看到实验内容时时有点懵的，尤其是出现了我认知以外的格雷码。但是好在在查阅百度百科的情况下我认识了格雷码，百度百科提供的表格让我们很容易的总结出了格雷码的变化规律，也就是n+1位格雷码的集合 = n位格雷码集合(顺序)加前缀0 + n位格雷码集合(逆序)加前缀1。在总结出规律动手写代码的过程中运用了Java语言进行实现

graycode[i] = "0" + lastGrayCode[i];

graycode[graycode.length - 1 - i] = "1" + lastGrayCode[i];

这两行代码是实现该递归写法的重要代码。

**if**(n<1) {

System.***out***.println("你输入的格雷码位数有误！");

**return**graycode;

}

**if** (n == 1) {

graycode[0] = "0";

graycode[1] = "1";

**return**graycode;

}

这段代码为递归循环的出口，也是重要代码之一。

这次的实验报告完成与编写，让我收益良多，辛苦老师批改审阅了。