

**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** | **173** | **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 windows 10 64bit

② 编译环境 eclipse 2018

## 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

# 实验记录

对于格雷码的定义，一开始是先查百度，然后得出以下结论：传统的二进制系统例如数字3的表示法为011，要切换为邻近的数字4，也就是100时，装置中的三个位元都得要转换，因此于未完全转换的过程时装置会经历短暂的，010,001,101,110,111等其中数种状态，也就是代表着2、1、5、6、7，因此此种数字编码方法于邻近数字转换时有比较大的误差可能范围。葛雷码的发明即是用来将误差之可能性缩减至最小，编码的方式定义为每个邻近数字都只相差一个位元，因此也称为最小差异码，可以使装置做数字步进时只更动最少的位元数以提高稳定性。

说实话这段话怎么看都看不懂，后来再找就看到这个

1、除了最高位（左边第一位），格雷码的位元完全上下对称（看下面列表）。比如第一个格雷码与最后一个格雷码对称（除了第一位），第二个格雷码与倒数第二个对称，以此类推。

2、所以，在实现的时候，可以利用递归，在每一层前面加上0或者1，然后就可以列出所有的格雷码。

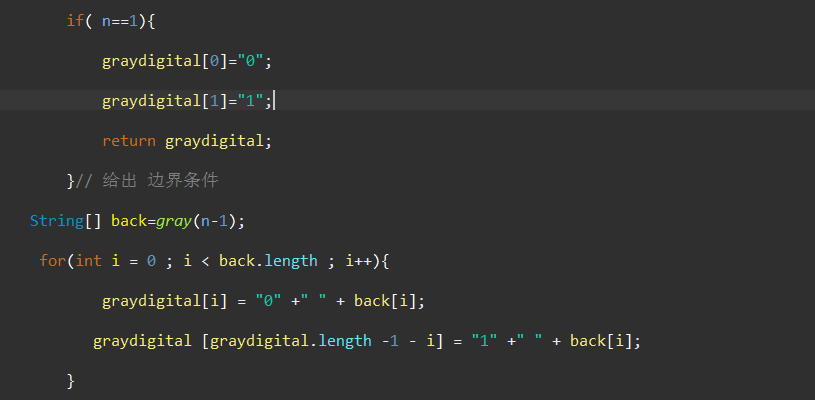
如果要生成4位元格雷码，我们只需要在3位元格雷码上再加一层0,1就可以了： 0000,0001,0011,0010,0110,0111,0101,0100,1100,1101,1111，1110,1010,0111,1001,1000.

也就是说，n位元格雷码是基于n-1位元格雷码产生的。

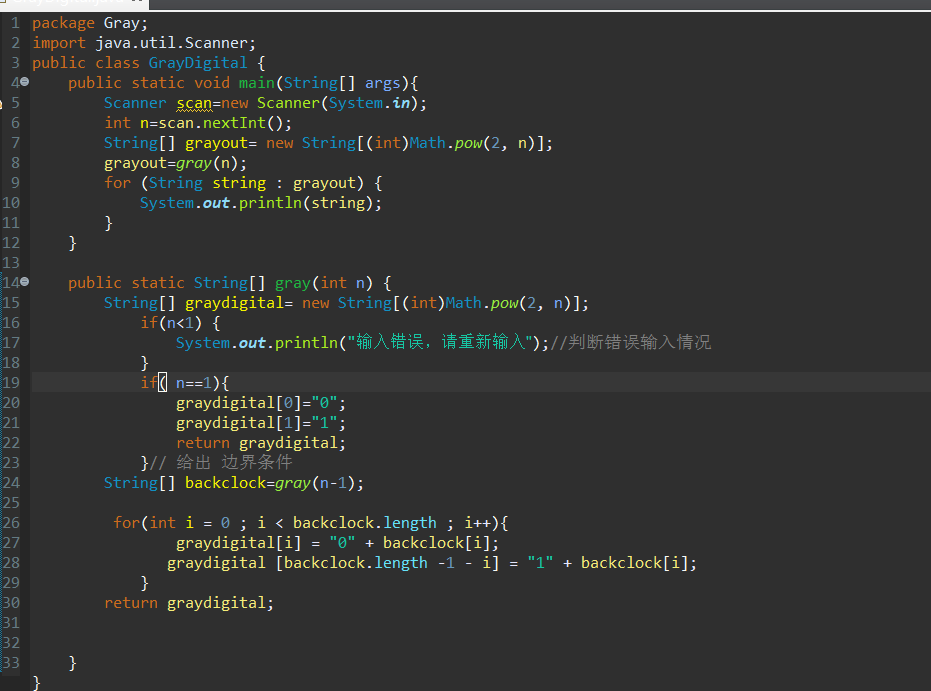
好的，大概了解基本结构后，开始编写递归写法

## 2.2 实验过程

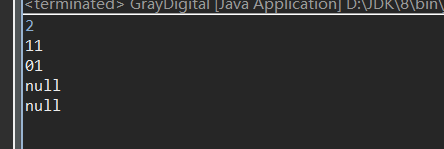
1.一开始是蒙的，就是大概知道格雷码的机制，但是具体写出来总是在哪里进行递归卡主，然后就上网找格雷码的实现方法，一看下来，大概一个语言都能用五六种格雷码递归的写法，但是都大同小异，也就是看了网上的代码之后，我也就仿照着网上的思想写了一份代码

然后，整个代码的核心就是这几行代码

分析：上面的话，因为格雷码的性质，所有的格雷码都是建立在比他小一位的格雷码之上了，而最小一位的格雷码，就是当n=1的情况，此位数下所有格雷码为0 .1，而第二位就是在左边加上1，右边加上0就行，而这就是本次递归的 基础，了解完这些，就可以写出所有代码了。



但是这个代码是会报错的，我怎么运行输出的都是null



后来就觉得我和网上的思想差不多啊，为什么我写出来就不行。

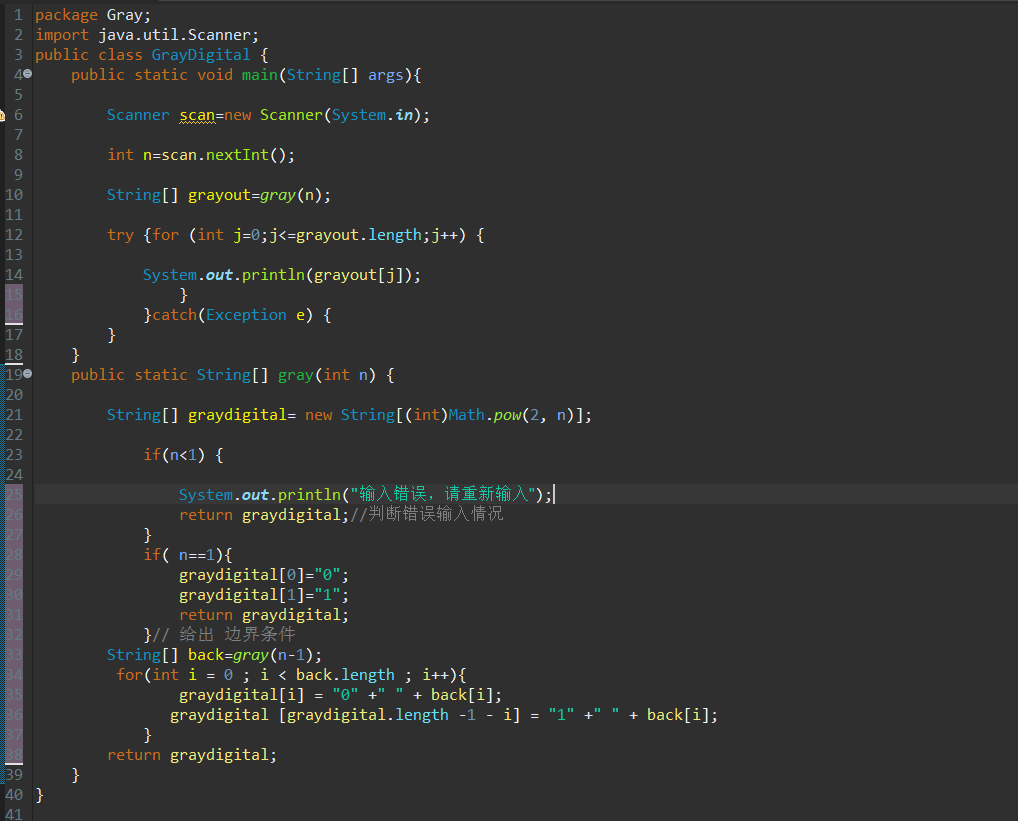
后来发现是第二十八行



这一行出错



就是反复每一行对下来，修改，注释看看是不是单句句子错了，然后发现有一句的数组弄混了，修改完之后，程序也就跑得动了，以下是完整代码



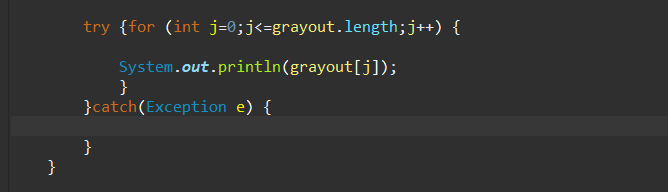
## 2.3 实现结果







# 实验总结

通过上面也可以看到，不知道为什么，我的运行结果虽然是正确的，但是main总是抛出异常，这个问题我对照了网上的很多代码，关键部分我也没有差也会这样，然后自己修改了很多之后还是这样，并不知道什么问题。我哭，改了好久好久就是改不掉，一般打代码遇到这种事也就是细节问题了，但是真的找不到了。最后只能写一个这样把异常捕捉了。

除了上面这件事以外，整个格雷码的学习对我来说还是教会我蛮多的，起码我知道了格雷码这种编码方式很棒，有他一定的优越性。而且对于递归的用法也有了进步，希望还能多加练习。

谢谢老师批阅。