装

订

线

Color Linez设计报告

班级：计算机一班

姓名：王哲源

学号：1652228

完成日期：2016.12.22

装 订 线

1.总结游戏规则

以下先进行对于该游戏基本设计思路的一些分析

1.1 游戏区域的大小

游戏操作界面为 9×9 的矩阵，同时游戏操作界面上方左右侧各有 6 字符长度的位置作为最高分（初始为 100）及当前分数显示，正中间有 1×3 的区域显示接下来生成的三个彩球颜色

1.2 球的颜色种类

包含红、青蓝、深蓝、绿、黄、棕、粉色七种颜色

1.3 初始球的数量

5 个

1.4 移动规则

每个球只能向上下左右四个方向移动。被鼠标点击选择的球以当前位置为起始位置，鼠标再次点击位置为终点位置，且成功移动的前提是终点位置不存在球，且从起始位置存在一条满足球移动规则的通路到达终点位置。

1.5 计分规则

若当前存在 5 球以上的同色球位于同一行/列/斜线，则这些同色球同时消去，且按照如下方式计分：

5 球连珠：10 分
6 球连珠：12 分
7 球连珠：18 分
8 球连珠：28 分
9 球连珠：42 分
…

由网上获得的总得分计算公式为,若由 A 个球同时打成消去条件,且形成了 B 条可消去的线，则总的得分 S=10+2(A+B-6)^2

同时注意，若是由于系统自动放上去的球导致的消去，不会使得分增加，且这些球会被消去并被重新放一个随机颜色的球到一个随机的空位置上。

1.6 游戏结束规则

当 9×9 的游戏界面完全被球填满时，游戏结束

2.在 CMD 界面下实现游戏需要考虑的问题

2.1 关于内部的存储

游戏操作界面由 9×9 的二维数组进行存储。map[i][j]表示第 i 行第 j 列位置的球的颜色。球颜色由 1-7 进行表示，空白位置则由 0 作为缺省值。

另外，也可将 i,j 以 9*(i-1)+j（此处 i,j 均以 1 开始为例）的方式以一维数组进行存储。下文为便于解释均以二维数组为例。

2.2 移动的判断方法

此处可以采用深度/广度遍历的方法。

深度优先遍历方法：设当前点为第 i 行第 j 列，每次调用向上下左右 4 个方向中可以移动的位置的递归函数，用一个布尔值作为返回值，1 表示递归找到一条通路到达指定位置；0 表示未找到。每次移动操作通过调用该递归函数判断移动是否成立即可。

广度优先遍历方法：以当前点为起点，每次扩展当前点的上下左右四个方向中可以扩展的点。通过将图刷新一遍的方式查询是否存在一条从起始位置到达目标位置的通路。若目标位置被扩展到则说明存在，否则说明不存在。

注意，不管哪个方法，每次判定都需要对于已经扩展的点进行标记不对其进行二次扩展，否则将出现死循环情况。同时可以开两个二维数组存储当前点是由哪一个点扩展而来（或以 2.1 中提及的一个一维数组来存储），便于之后移动操作中的路径显示。

2.3 关于是否连成线的判定

首先可以确定，在每次移动操作前界面中不存在满足条件的连线方式。因此我们只需要对于每次移动操作结束后被移动球到达的新位置，以其为中心进行判定是否存在满足条件的连线方式即可。判定方式分为以下四种：

一，向左右同时扩展寻找相同颜色的球并统计个数

二，向上下同时扩展寻找相同颜色的球并统计个数

三，向左上及右下同时扩展寻找相同颜色的球并统计个数

四，向左下及右上同时扩展寻找相同颜色的球并统计个数

进行完这些操作后对于可行方案的连线进行分数统计，并进行界面的更新。

同时应当注意的是，生成新的球时也有可能出现满足条件的连线情况（虽然非常小但也有能）。因此对于新生成的三个球也有必要进行以上判定。

2.4 关于游戏结束的判断

判断游戏结束有两种思路。第一种是可以对于每次生成新的球之前对整张图的空格进行统计，判断是否存在三个以上（不包括三个）的空格，若存在则继续生成，否则游戏结束。第二种方法可以用一个全局变量实时存储当前游戏的空格个数，每当生成新的球减三，连线时加上对应个数，剩余判定方法同第一种。

2.5 关于 Fn 键位的读取

获取键位值的方法有两种，一种是可以直接去查询键位的 ascii 码，另一种我们可以利用以下函数

demo.cpp

```cpp
#include<iostream>
#include<cstdio>
#include<conio.h>

using namespace std;

int main()
{
    while(1)
    {
        int tmp=_getch();
        printf("%d\n",tmp);
    }

    return 0;
}
```

（注意，_getch()非标准库函数，注意移植性）

通过运行该函数发现 F1 有两个键位置，第一个为 0，第二个为 59，F2-F10 依此类推。特别应当注意的是，F11 和 F12 的第一个 ascii 为 224，F11 第二个 ascii 为 133，F12 为 134。因此我们在读入键位部分时先进行第一次读取，若为 0 则进行第二次读取，判断其值是否落在 Fn 范围内即可。