学生学号	0121710880414	实验课成绩	

武漢理工大學

学 生 实 验 报 告 书

实验课程名称	数据结构与算法综合实验
开 课 学 院	计算机科学与技术学院
指导教师姓名	夏红霞
学 生 姓 名	穆逸诚
学生专业班级	软件 1704 班

2018 -- 2019 学年 第 二 学期

实验教学管理基本规范

实验是培养学生动手能力、分析解决问题能力的重要环节；实验报告是反映实验教学水平与质量的重要依据。为加强实验过程管理，改革实验成绩考核方法，改善实验教学效果，提高学生质量，特制定实验教学管理基本规范。

1、 本规范适用于理工科类专业实验课程，文、经、管、计算机类实验课程可根据具体情况参照执行或暂不执行。

2、 每门实验课程一般会包括许多实验项目，除非常简单的验证演示性实验项目可以不写实验报告外，其他实验项目均应按本格式完成实验报告。

3、 实验报告应由实验预习、实验过程、结果分析三大部分组成。每部分均在实验成绩中占一定比例。各部分成绩的观测点、考核目标、所占比例可参考附表执行。各专业也可以根据具体情况，调整考核内容和评分标准。

4、 学生必须在完成实验预习内容的前提下进行实验。教师要在实验过程中抽查学生预习情况，在学生离开实验室前，检查学生实验操作和记录情况，并在实验报告第二部分教师签字栏签名，以确保实验记录的真实性。

5、 教师应及时评阅学生的实验报告并给出各实验项目成绩，完整保存实验报告。在完成所有实验项目后，教师应按学生姓名将批改好的各实验项目实验报告装订成册，构成该实验课程总报告，按班级交课程承担单位（实验中心或实验室）保管存档。

6、 实验课程成绩按其类型采取百分制或优、良、中、及格和不及格五级评定。

附表：实验考核参考内容及标准

	观测点	考核目标	成绩组成
实验预习	1. 预习报告 2. 提问 3. 对于设计型实验，着重考查设计方案的科学性、可行性和创新性	对实验目的和基本原理的认识程度，对实验方案的设计能力	20%
实验过程	1. 是否按时参加实验 2. 对实验过程的熟悉程度 3. 对基本操作的规范程度 4. 对突发事件的应急处理能力 5. 实验原始记录的完整程度 6. 同学之间的团结协作精神	着重考查学生的实验态度、基本操作技能；严谨的治学态度、团结协作精神	30%
结果分析	1. 所分析结果是否用原始记录数据 2. 计算结果是否正确 3. 实验结果分析是否合理 4. 对于综合实验，各项内容之间是否有分析、比较与判断等	考查学生对实验数据处理和现象分析的能力；对专业知识的综合应用能力；事实求实的精神	50%

实验课程名称：_____算法设计与分析实验_____

实验项目名称	欢乐连连看（C++&MFC）			实验成绩	
实 验 者	穆逸诚	专业班级	软件 1704	组　　别	
同 组 者				实验日期	2019 年 5 月 27 日

第一部分：实验分析与设计

一、实验目的和要求

（1）了解项目业务背景，调研与连连看相同类型的游戏，了解连连看的功能和规则等；

（2）掌握 C++开发工具和集成开发环境（Visual C++ 6.0 或 Microsoft Visual Studio 2010）；

（3）掌握 C++面向对象的编程思想和 C++的基础编程；

（4）理解 MFC 基本框架，包括 MFC Dialog 和 GDI 编程；

（5）掌握数据结构，包括算法控制、数组、栈、图；同时掌握算法，比如数组的遍历、图的遍历（如：DFS）、连通判断等；

（6）实施项目的调研和分析，阅读和填充项目的过程文档；

（7）了解项目的开发流程，学习软件工程的迭代思想，开发 GUI 应用程序；

（8）养成良好的编码习惯，提高 C++语言编程能力，开发应用程序“欢乐连连看”。

二、实验分析与设计

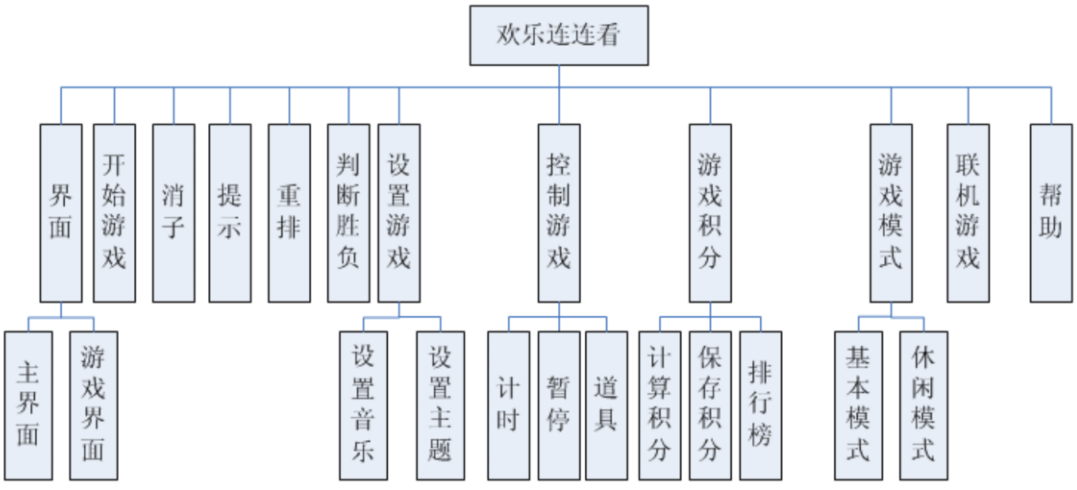
1、 文件组织形式

本次实验主要涉及到图这一数据结构，因而采取二维数组（矩阵）的形式存储图，包括顶点与边数据。

2、 数据结构设计

在本次实验中，运用到了多种数据结构算法，如：深度优先搜索等，主要用于寻找一条有效连通路径（最多由 3 条直线组成的路径）。

3、 系统功能



（1）开始游戏

玩家选择游戏模式，进入游戏后，选择开始游戏，系统根据设置的主题风格生成一个图片的布局（游戏地图），以供玩家点击消除；

（2）消子

对玩家选中的两个团进行判断，是否符合消除的规则。只有符合以下条件的图案对才会消

失；

（3）提示

当玩家选择提示功能时，提示玩家符合游戏规则可以消除一对图片；

（4）重排

选择重排功能时，系统将会对游戏地图中剩下的图片进行重新排列，重新排列只是将所有的图片的位置随机互换，不会增减图案的种类和个数；

（5）判断胜负

当游戏完成后，需要判断游戏胜负，不同模式下判断胜负的规则不同；

（6）设置游戏

设置游戏模块可以设置游戏的音乐效果、设置游戏的主题；

（7）控制游戏

控制游戏模块为对游戏进行控制，有如下控制方式：计时、暂停、道具；

（8）游戏积分

游戏积分是根据玩家在游戏时，消除的图片数量和消除的速度，进行积分计算，在游戏结束时保存游戏积分。玩家可以在主界面中查看排行榜信息；

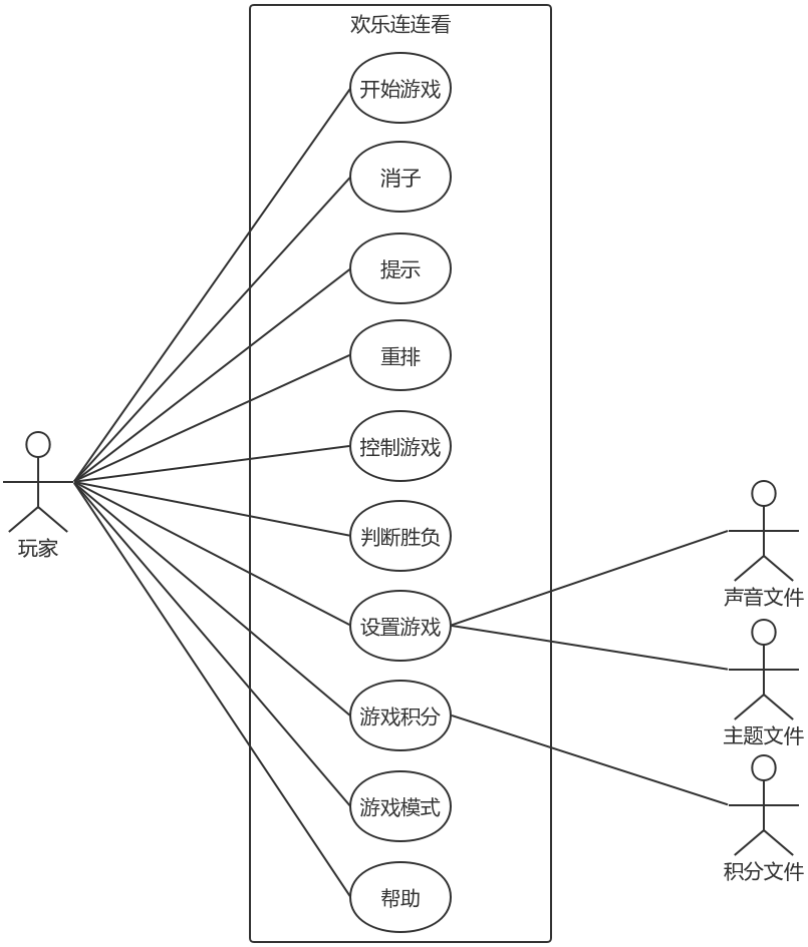
（9）游戏模式

欢乐连连看游戏有多种模式供玩家选择，分别为：基本模式、休闲模式等。当玩家在主界面中选择一种模式，则进入玩家选择的游戏模式，开始游戏；

（10）帮助

在任何界面选择帮助时，在帮助窗口中用图片显示欢乐连连看游戏帮助信息和版权信息。

4、 系统用例图



5、 核心技术

（1）Visual Studio 2017 工具的使用；

（2）C++面向对象的基础编程语言；

（3）MFC 框架；

（4）GDI 绘图；

（5）数据结构和算法：数组、栈、图；

（6）数据存储：文件操作；

（7）程序的三层结构：表示层、业务逻辑层、数据访问层。

6、 头文件函数展示

（1）ChelpDialog.h

```
#pragma once
// CHelpDialog 对话框
class CHelpDialog : public CDialogEx
{
	DECLARE_DYNAMIC(CHelpDialog)

public:
	CHelpDialog(CWnd* pParent = nullptr);	// 标准构造函数
	virtual ~CHelpDialog();

// 对话框数据
#ifdef AFX_DESIGN_TIME
	enum { IDD = IDD_DIALOG_HELP };
#endif

protected:
	virtual void DoDataExchange(CDataExchange* pDX);	// DDX/DDV 支持

	DECLARE_MESSAGE_MAP()

public:

	virtual BOOL OnInitDialog();

	afx_msg void OnPaint();

protected:
	HICON m_hIcon;		//系统图片
	CDC m_dcMem;		//内存 DC
	CDC m_dcHelp;		//帮助图片 DC
	CRect m_rtHelp;		//帮助图片显示区域

public:
	afx_msg void OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar);
```

```
    void UpdateHelp(int nPos);      //根据滚动条的位置，设置帮助图片的位置显示的内容
};
```

（2）GameControl.h

```
#pragma once
#include "global.h";
#include "Graph.h"

class CGameControl
{
public:
    CGameControl();
    ~CGameControl();

protected:
    CGraph m_graph;              //初始游戏地图
    Vertex m_ptSelFirst;     //第一次选中的点(x 表示列)
    Vertex m_ptSelSec;       //第二次选中的点

public:
    //开始游戏函数
    void StartGame(void);

    //获得某行某列的图片编号函数
    int GetElement(int nRow, int nCol);

    void SetFirstPoint(int nRow, int nCol);                    //设置第一个点函数
    void SetSecPoint(int nRow, int nCol);                       //设置第二个点函数

    //连接判断函数
    bool Link(Vertex avPath[MAX_VERTEX_NUM], int &nVexnum);

    //获胜    //加入计时功能后规则进行修改
    //bool IsWin();
    bool IsWin(int nTime);

    //帮助、提示方法
    bool Help(Vertex avPath[MAX_VERTEX_NUM], int &nVexnum);

    //实现重排功能
    void Reset(void);
};
```

（3）GameDlg.h

```
#pragma once
#include "global.h"
#include "GameControl.h"
#include "GameLogic.h"
#include "CHelpDialog.h"

// CGameDlg 对话框

class CGameDlg : public CDialogEx
{
    DECLARE_DYNAMIC(CGameDlg)

public:
    CGameDlg(CWnd* pParent = nullptr);    // 标准构造函数
    virtual ~CGameDlg();

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_GAME_DIALOG };
#endif

protected:

    CDC m_dcMem;              //内存 DC
    CDC m_dcElm;              //元素内存 DC
    CDC m_dcMask;             //掩码内存 DC
    CDC m_dcBG;                   //背景 DC
    CDC m_dcCache;            //掩盖游戏地图

    CPoint m_ptGameTop;       //起始点坐标
    CSize m_sizeElem;      //图片元素大小
    CRect m_rtGameRect;       //游戏区域的大小

    bool m_bFirstPoint;           //选中的图片是不是第一次选中的，TRUE    是，FALSE
否
    Vertex m_ptSelFirst;    //第一次选中的点(x 表示列)
    Vertex m_ptSelSec;      //第二次选中的点

    CGameControl m_gameControl; //游戏控制类

    bool m_bPlaying; //是否在玩游戏
    bool m_bPause;    //是否暂停 false 进行游戏中
    int timeCount;         //计时
```

```
protected:

    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV 支持

    DECLARE_MESSAGE_MAP()
public:
    void InitBackground();  //初始化背景
    void InitElement();     //初始化元素
    void UpdateWindow();        //
    void UpdateMap();
    void DrawTipFrame(int nRow, int nCol);          //绘制游戏提示框
    void DrawTipLine(Vertex avPath[MAX_VERTEX_NUM], int nVexnum);     //绘制提示
线

    virtual BOOL OnInitDialog();
    afx_msg void OnPaint();

    afx_msg void OnClickedButtonStart();
    afx_msg void OnLButtonUp(UINT nFlags, CPoint point);        //鼠标点击事件

    afx_msg void OnClickedButtonTips();
    afx_msg void OnClickedButtonRestart();
    CProgressCtrl m_GameProgress;
    afx_msg void OnTimer(UINT_PTR nIDEvent);

    void JudgeWin(void);          //判断胜负

    afx_msg void OnClickedButtonStop();
    afx_msg void OnChangeEditTime();
    afx_msg void OnBnClickedButtonHelp();
};
```

（4）GameLogic.h

```
#pragma once
#include "global.h";
#include "Graph.h";
class CGameLogic
{
public:
    CGameLogic();
    ~CGameLogic();

protected:
    Vertex m_avPath[MAX_VERTEX_NUM];     //保存连接路径的起始点、拐点、终点
```

```
    int m_anPath[MAX_VERTEX_NUM];      //保存在进行连接判断时所经过的顶点
    int m_nCorner;                     //保存路径中的拐点数
    int m_nVexNum;                     //表示顶点数

public:

    //初始化游戏地图
    void InitMap(CGraph &graph);

    //生成图的边的数组，更新边
    void UpdateArc(CGraph &graph, int nRow, int nCol);

    //连接判断函数
    bool IsLink(CGraph &graph, Vertex v1, Vertex v2);

    //消子函数
    void Clear(CGraph &graph, Vertex v1, Vertex v2);


    //判断选择的两个顶点是否联通
    bool SearchPath(CGraph &graph, int nV0, int nV1);

    //判断顶点是否已在路径中存在
    bool IsExsit(int nVi);

    //判断拐点的有效性
    bool IsCornor(void);

    //添加一个路径顶点
    void PushVertex(int v);

    //取出一个路径顶点
    void PopVertex();

    //得到路径，返回的是顶点数
    int GetVexPath(Vertex avPath[MAX_VERTEX_NUM]);

    ////判断图中顶点是不是全是空
    bool IsBlank(CGraph &graph);

    //提示算法，
    bool SearchValidPath(CGraph &graph);

    //实现图结构的重排
```

```
        void ResetGraph(CGraph& graph);
    };
```

（5）Graph.h

```
    #pragma once
    #include "global.h";
    //图   数据结构类

    class CGraph
    {
    public:
        CGraph();
        ~CGraph();

        typedef int Vertices[MAX_VERTEX_NUM];                                   //顶点数据类型
        typedef bool AdjMatrix[MAX_VERTEX_NUM][MAX_VERTEX_NUM];          // 边 数据类型矩阵



    protected:

        Vertices m_Vertices;                            //顶点数组   一位数组，保存连连看游戏地图中的顶点
        AdjMatrix m_AdjMatrix;                          //关系矩阵   二位数组，保存连连看游戏中的顶点的边

        int m_nVexnum;          //顶点数
        int m_nArcnum;          //边数

    public:

        //初始化图
        void InitGraph();

        //添加顶点，并获得顶点个数
        int AddVertex(int nInfo);

        //添加生成边
        void AddArc(int nV1Index,int nV2Index);

        //获取顶点索引号
        int GetVertex(int nIndex);

        //获得两个顶点的边信息
```

```
        bool GetArc(int nV1Index, int nV2Index);

        //更新顶点,将图顶点数组中索引号为 NIndex 的顶点的值更新为 info
        void UpdateVertex(int nIndex,int info);

        //获取图中顶点的个数
        int GetVexnum();

        //清理  图结构
        void ClearGraph(void);

        //调换两个点的位置
        void ChangeVerex(int nIndex1, int nIndex2);

};
```

（6）LLKDlg.h

```
#include "CHelpDialog.h"
// LLKDlg.h: 头文件
//

#pragma once


// CLLKDlg 对话框
class CLLKDlg : public CDialogEx
{
// 构造
public:
        CLLKDlg(CWnd* pParent = nullptr);   // 标准构造函数

// 对话框数据
#ifdef AFX_DESIGN_TIME
        enum { IDD = IDD_LLK_DIALOG };
#endif

        protected:
        virtual void DoDataExchange(CDataExchange* pDX);       // DDX/DDV 支持

// 实现
protected:
        HICON m_hIcon;
        CDC m_dcMem;
        // 生成的消息映射函数
```

```
    virtual BOOL OnInitDialog();
    afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    DECLARE_MESSAGE_MAP()

    void InitBackground();
public:
    afx_msg void OnClickedButtonBasic();
    afx_msg void OnBnClickedButtonLlkhelp();
};
```

7、 核心算法展示

（1）深度优先算法

```
//使用深度优先搜索法搜寻一条有效连通路径
bool CGameLogic::SearchPath(CGraph &graph, int nV0, int nV1)
{
    //得到顶点数
    int nVexnum = graph.GetVexnum();

    //遍历图中 nV0 行，从 0 列到 nVexnum 列，值为 true 的点
    for (int nVi = 0; nVi < nVexnum; nVi++)
    {
        if (graph.GetArc(nV0, nVi) && !IsExsit(nVi))
        {
            //压入当前顶点，假设为路径的一个有效顶点
            PushVertex(nVi);
            //当拐点数大于 2 时，直接放弃该顶点
            if (m_nCorner > 2)
            {
                PopVertex();            //取出压入的顶点，与 PushWertex(nVi)对应
                continue;
            }
            //当前顶点不是 nVi 时，继续搜寻下一个相邻且连通的顶点
            if (nVi != nV1)
            {
                //当中间顶点不为空时，表示该条路径不通
                if (graph.GetVertex(nVi) != BLANK)
                {
                    PopVertex();        //取出压入的顶点，与 PushWertex(nVi)对应
                    continue;
                }
                //如果 nVi 是一个已消除的点，则判断（nVi，nV1）是否连通
                if (SearchPath(graph, nVi, nV1))
```

```
                    {
                        //SearchPath(garph, nVi, nV1) == true,表示已经找到一条连通路径，则返回 true
                        return true;
                    }
                }
                else
                {
                    return true;
                }

                PopVertex();        //取出压入的顶点，与 PushWertex(nVi)对应
            }
        }
        return false;
    }
```

（2）图的重排

```
    //实现图结构的重排
    void CGameLogic::ResetGraph(CGraph& graph)
    {
        //随机交换顶点数组中两个顶点的值
        for (int i = 0; i < 200; i++)
        {
            //随机得到两个坐标
            int nIndex1 = rand() % MAX_VERTEX_NUM;
            int nIndex2 = rand() % MAX_VERTEX_NUM;

            //交换两个数值
            graph.ChangeVerex(nIndex1, nIndex2);
        }

        //更新弧信息
        for (int i = 0; i < MAX_ROW; i++)
        {
            for (int j = 0; j < MAX_COL; j++)
            {
                UpdateArc(graph, i, j);
            }
        }
    }
```

三、主要仪器设备及耗材

1．PC 机

2．开发环境：VS2017

第二部分：实验调试与结果分析

一、**调试过程**（包括调试方法描述、实验数据记录，实验现象记录，实验过程发现的问题等）

1．调试方法描述

① 输入 C++程序，并保存；

② 编译 C++程序，找出程序的语法错误并改正；

③ 输入测试数据（除每区域普遍值外包括一些特殊临界值），运行 C++程序，若有错，查找并修改程序的逻辑错误；

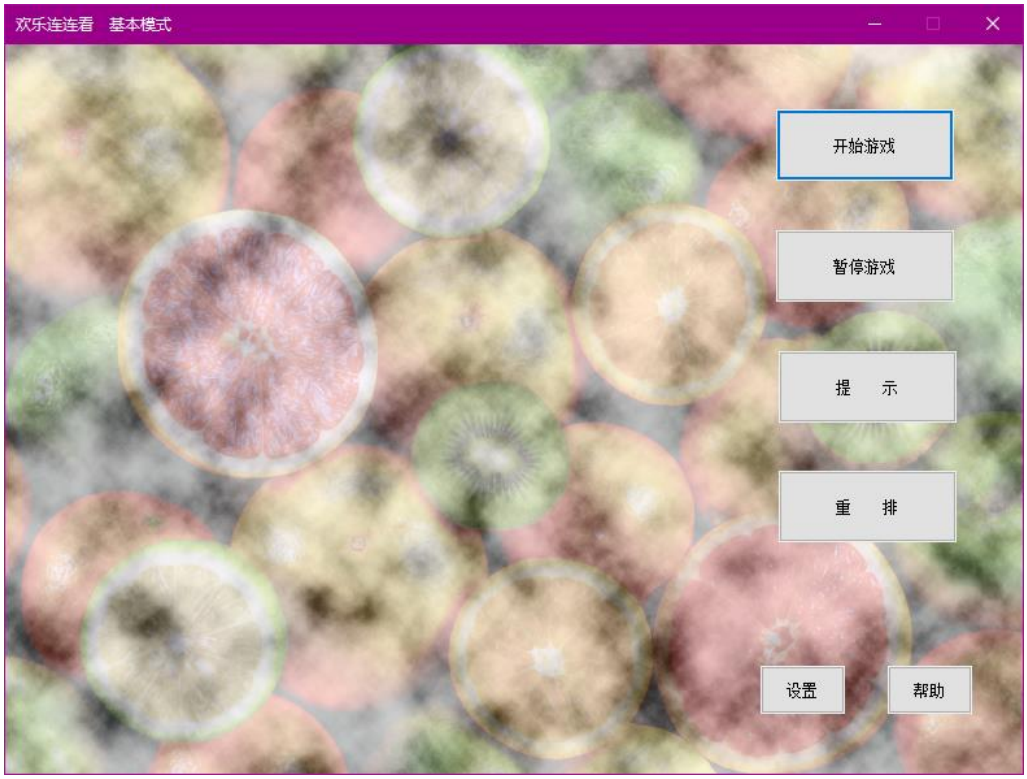
④ 重复②-③步，直到得到正确的运行结果

2．实验输入/输出数据记录

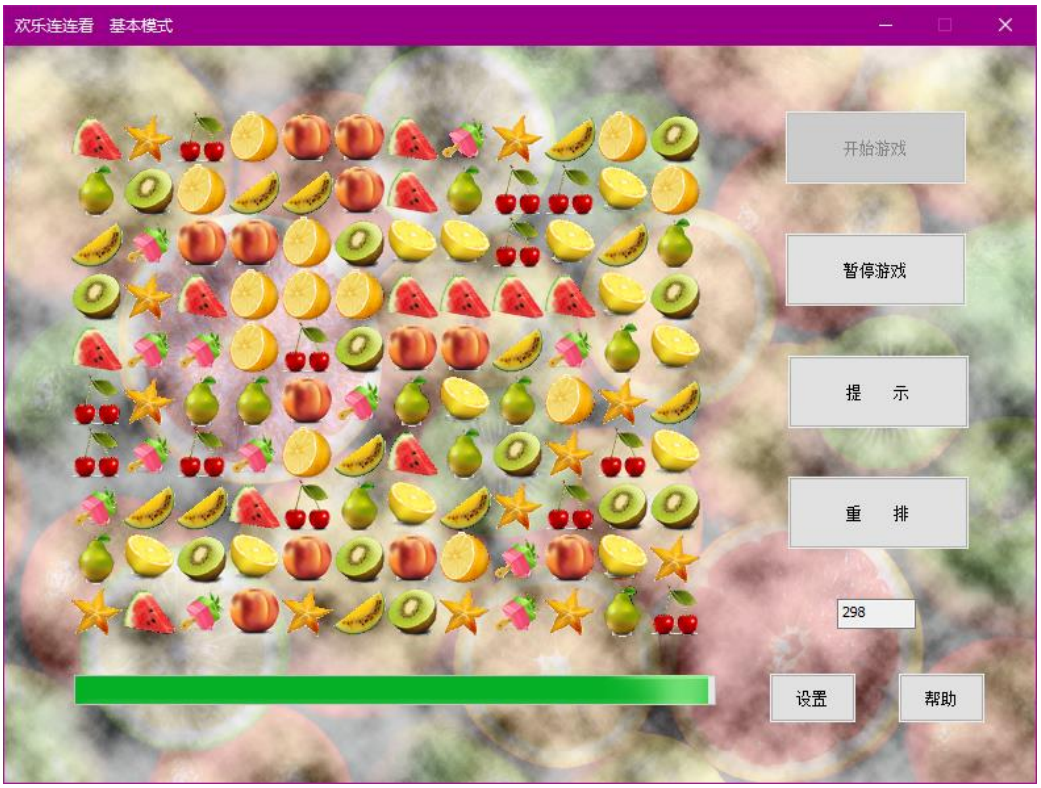
—>★具体实验操作截图如下：

（1）游戏主界面；

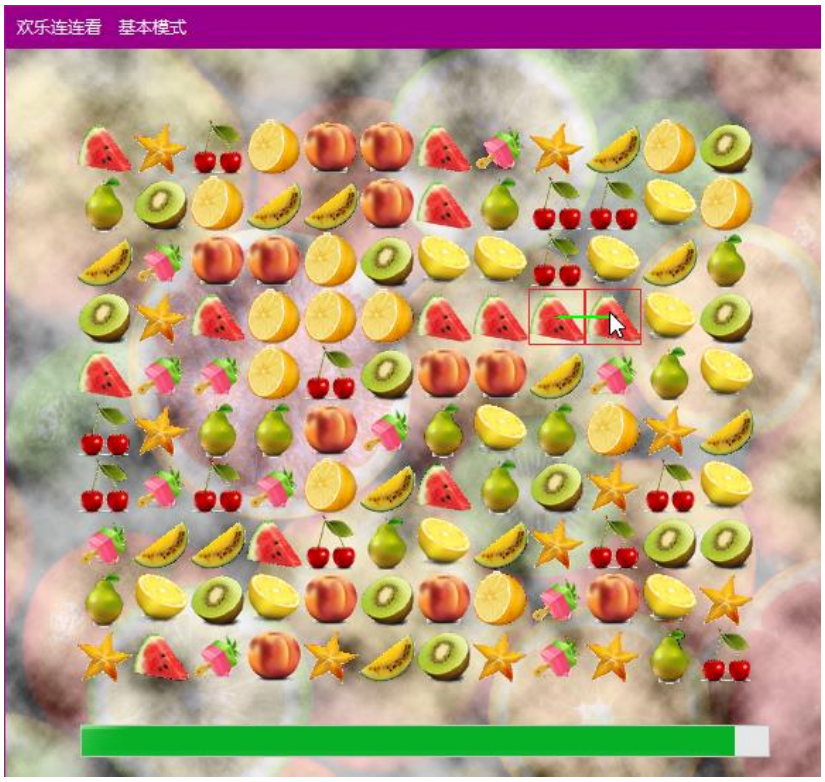
（2）基本模式界面；



（3）游戏运行界面；

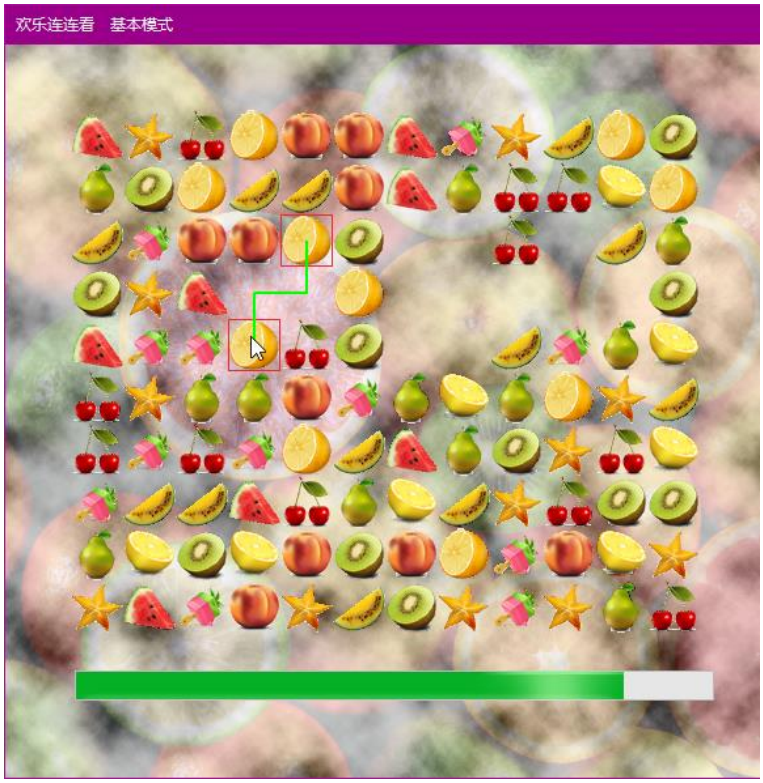


（4）同色消除（1 直线）；

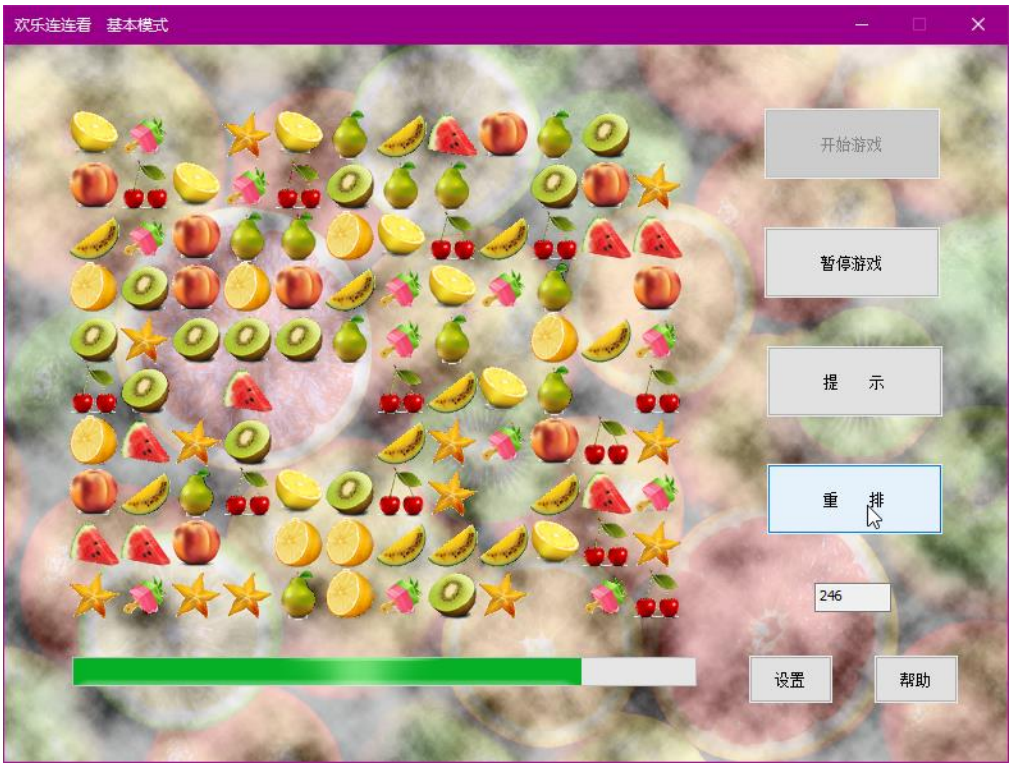


（5）同色消除（2 直线）；

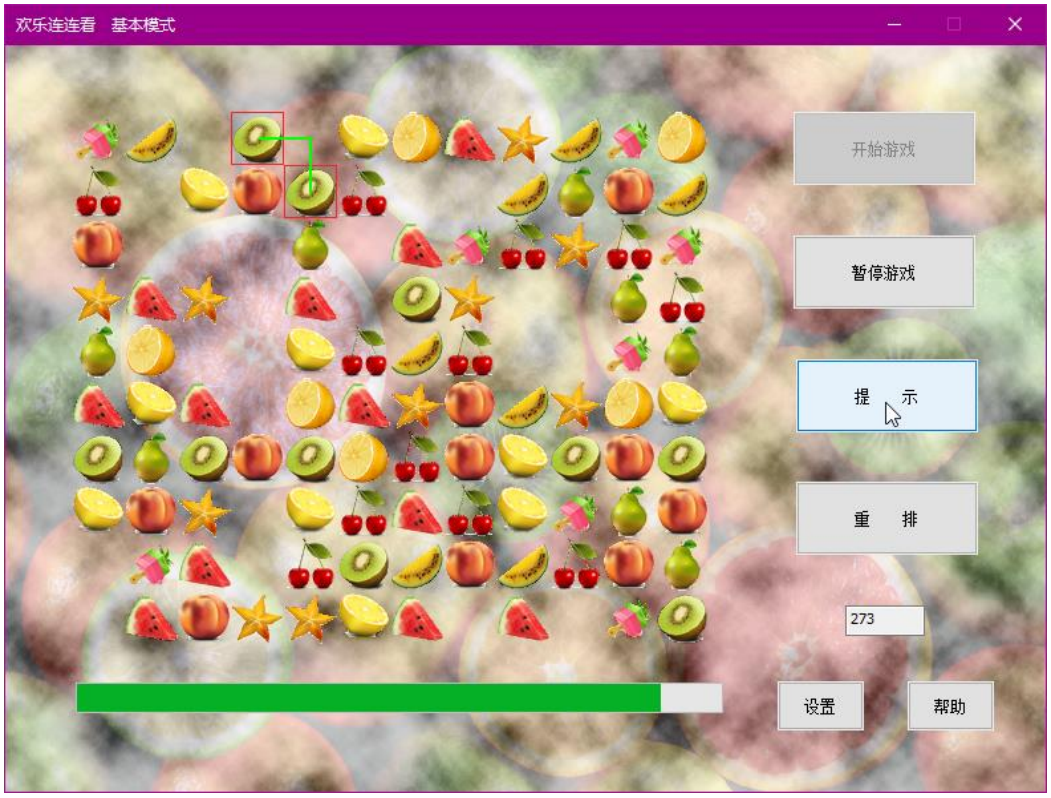
（6）同色消除（3 直线）；



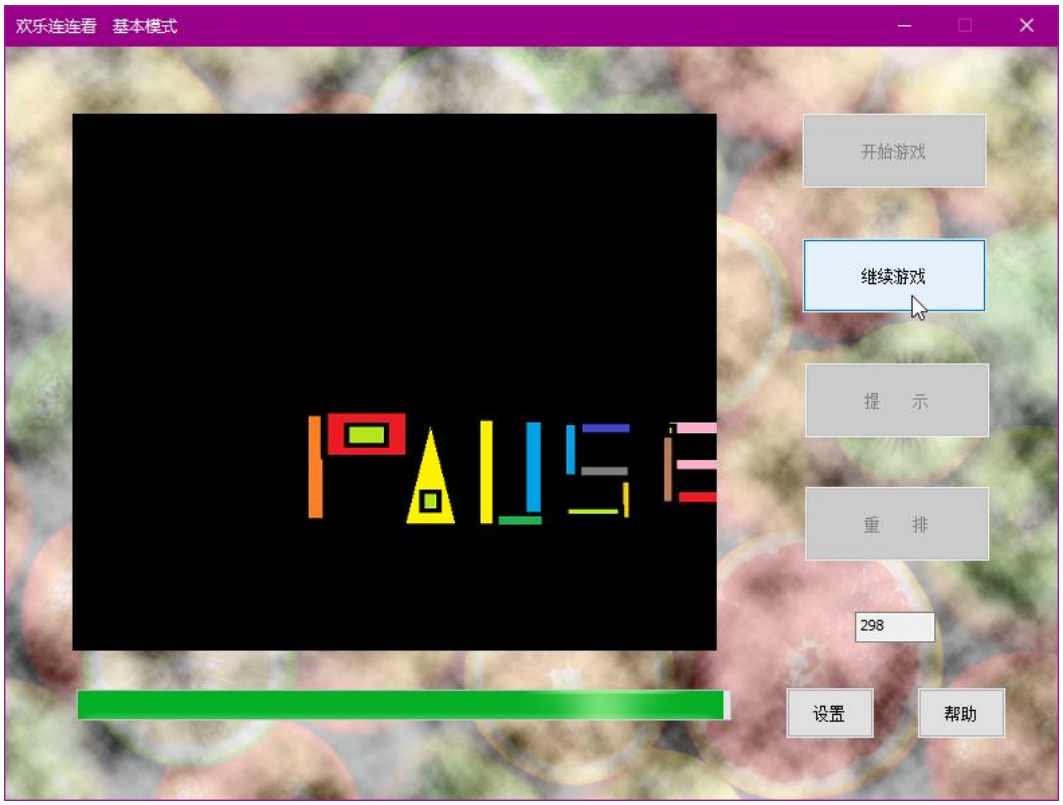
（7）重排（对于（6）中的排列进行重排）；



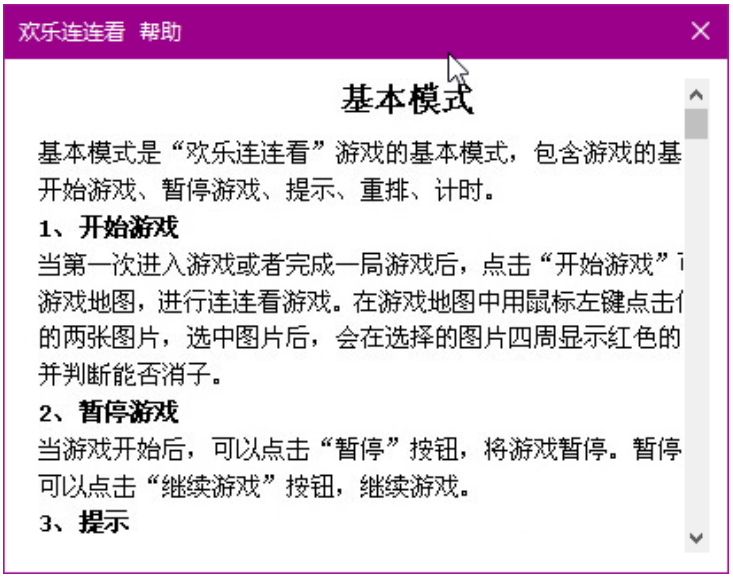
（8）提示；


欢乐连连看 基本模式
开始游戏
暂停游戏
提 示
重 排
273
设置
帮助

（9）暂停界面；


欢乐连连看 基本模式
开始游戏
继续游戏
提 示
重 排
298
设置
帮助

（10）帮助界面；



3. 实验过程发现的问题

（1）载入新图片作为背景出错。在导入新的游戏背景时，运行程序后，游戏背景仍为原来的图片，并没有改变，重启程序后该问题仍然存在。经查阅资料与不断尝试得知，在导入新的图片后，需在 Resource.h 中改变对应图片资源的权重，要小于原图片的数字，方可更新游戏背景；

```
#define IDD_LLK_DIALOG                  102
#define IDR_MAINFRAME                   128
#define IDB_MAIN_BG                     130
#define IDD_GAME_DIALOG                 131
#define IDD_DIALOG_HELP                 133
```

（2）水果消失异常。在编写关于水果消除后隐藏操作时，未将元素数据的计算公式搞清楚，导致水果隐藏出现异常，无法与背景图片融为一体实现“消失”效果。经过改正，正确代码如下，程序运行正确：

```
//将背景与掩码相或，边保留，图像区域为1
m_dcMem.BitBlt(nLeft + j * nElemW, nTop + i * nElemH,

//将元素图片相与，边保留，图像区域为元素图片
m_dcMem.BitBlt(nLeft + j * nElemW, nTop + i * nElemH,
```

第三部分 实验小结、建议及体会

本次实验主要围绕欢乐连连看（C++&MFC）展开，旨在了解项目业务背景，调研与连连看相同类型的游戏，了解连连看的功能和规则等，并掌握 C++开发工具和集成开发环境（Visual C++ 6.0 或 Microsoft Visual Studio 2010），并养成良好的编码习惯，提高 C++语言编程能力，开发应用程序“欢乐连连看”，实验步骤较多，但实验难度不大。MFC(Microsoft Foundation Classes)，是微软公司提供的一个类库（class libraries），以 C++类的形式封装了 Windows 的 API，并且包含一个应用程序框架，以减少应用程序开发人员的工作量。其中包含的类包含大量 Windows 句柄封装类和很多 Windows 的内建控件和组件的封装类。

在本次实验中，我严格按照实验步骤与计划进行，绘制了用例图与功能图，通过多种数据结构与算法（图、深度优先搜索等），最终实现了欢乐连连看的多个功能，如消子、重排、暂停、提示、帮助等，程序运行无误。实验过程中难免遇到许多问题，如图片导入错误，消子异常等，好在遇到的问题通过不断尝试与查阅相关资料，最终都得以解决。

通过本次实验，我学会了通过 C++&MFC 来编写游戏程序，了解了 MFC 库与框架，对图等数据结构有了更深入的了解，并对深度优先搜索等算法有了进一步的认识与运用。总体而言，本次实验收获不少。

教师签字_____