

# 东南大学电子学院

## 计算机综合课程设计

### 报告

设计题目: Atom-Molecule Calculate 计算器

组长姓名: 孙寒石 学号: D2219117

组员姓名: 张扬 学号: D2219118

陶星宇 学号: D2219113

车旭明 学号: D2219114

#### 一、内容简介（书写空间自行拓展）

我们团队做的课题为: Atom-Molecule Calculate 计算器。

首先实现基础的计算器功能（Atom-Calculate），如带括号的加减乘除以及任意次方的混合运算，对结果 Ans 的进一步函数运算，如三角函数，对数函数等。进一步地，在对话框创建并完善了菜单功能。在拥有 Help, contact us 等基本功能的同时，新增 Molecule mode 功能，以实现更强的算力与数学功能，例如：解线性方程，解一元多次方程，矩阵运算，复数运算，日期换算，分解质因数，时间换算，简单的积分计算，多进制换算等（Molecule-Calculate）。又增加了 Draw mode，可以用来绘图。另外，系统还增加了 Leisure mode，链接了几个未与计算器系统完全封装在一起的小游戏。最后，为了提高对用户操作地友好性与安全性，将对开发的界面进行美化，同时增加用户登录密码等功能，提高安全性。

系统继承了常见计算器系统的大多数特点，尤其在带括号的加减乘除以及任意次方的混合运算都会完成经典再现。本系统创新主要在于其不仅局限于普通的计算器程序，还在模式 Molecule-Calculate 里面新增了解线性方程，解一元多次方程，矩阵运算，复数运算，日期换算，分解质因数，时间换算，简单的积分计算，多进制换算等功能。又增加了 Draw mode，可以用来绘图。

二、完成情况（详细叙述工程实现细节，书写空间自行拓展，以下各单元格都可自行扩展，可贴图）

### I. 课程设计任务书

#### • 任务

开发《Atom-Molecule Calculate 计算器》软件。

#### • 基本要求

使用面向对象的系统分析和设计，开发基于 MFC 对话框的计算器软件。

#### • 任务陈述

首先实现基础的计算器功能（Atom-Calculate），如带括号的加减乘除以及任意次方的混合运算，对结果 Ans 的进一步函数运算，如三角函数，对数函数等。进一步地，在对话框创建并完善了菜单功能。在拥有 Help, contact us 等基本功能的同时，新增 Molecule mode 功能，以实现更强的算力与数学功能，例如：解线性方程，解一元多次方程，矩阵运算，复数运算，日期换算，分解质因数，时间换算，简单的积分计算，多进制换算等（Molecule-Calculate）。又增加了 Draw mode，可以用来绘图。另外，系统还增加了 Leisure mode，链接了几个未与计算器系统完全封装在一起的小游戏。最后，为了提高对用户操作地友好性与安全性，将对开发的界面进行美化，同时增加用户登录密码等功能，提高安全性。

#### • 特色与创新体现

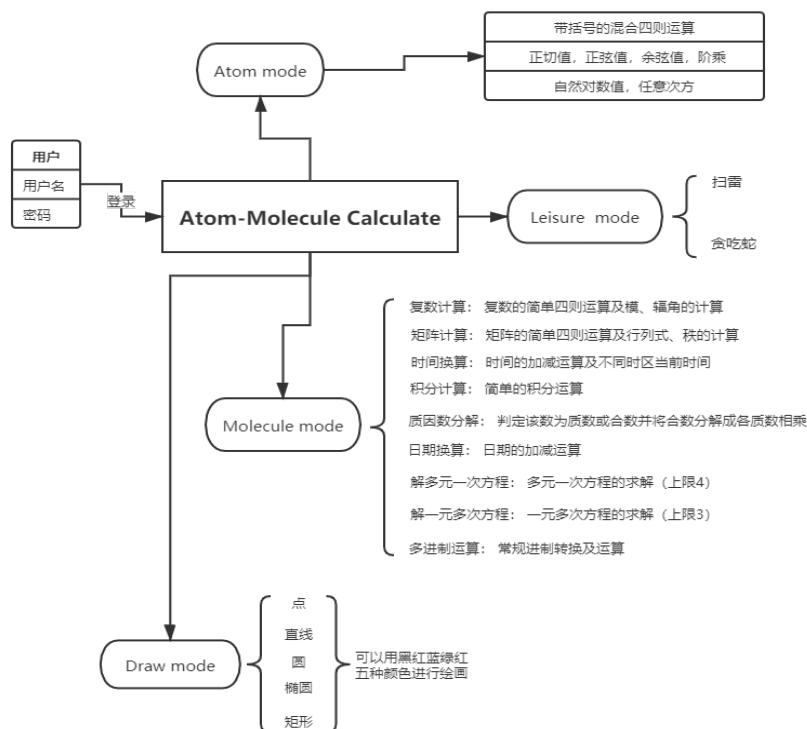
系统继承了常见计算器系统的大多数特点，尤其在带括号的加减乘除以及任意次方的混合运算都会完成经典再现。本系统创新主要在于其不仅局限于普通的计算器程序，还在模式 Molecule-Calculate 里面新增了解线性方程，解一元多次方程，矩阵运算，复数运算，日期换算，分解质因数，时间换算，简单的积分计算，多进制换算等功能。又增加了 Draw mode，可以用来绘图。

#### • 项目可行性

使用 C++ 语言编写主要算法程序，编译器使用 VS2019，界面开发采用 MFC，算法程序难度系数不高，只涉及简单的页面开发，自主学习需求较多，难度较高，但是四人合作可以完成。

四个人都有笔记本电脑，故具备设备可行性。

## • 预期结果



## • 项目时间进度

本次项目保守估计有三周时间，经商讨，我们定下如下时间节点：

第一周：学习 MFC、软件工程在课程设计中的应用之后，界面开发和代码编写同时进行，实现基础的计算器功能，称之为 Atom-Calculate。功能包括：带括号的加减乘除运算，对结果 Ans 的进一步函数运算，如三角函数，对数函数等。

第二周：创建并完善菜单功能，在拥有 Help, contact us 等基本功能的同时，新增 Molecule mode 功能，以实现更强的算力与数学功能，包括：解线性方程，解一元多次方程，矩阵运算，复数运算，分解质因数等，日期换算，时间换算，简单的积分计算，多进制换算等。这个阶段称之为 Molecule-Calculate。随后便开发 Leisure mode。

第三周：开发 Draw mode，测试调试并修改所有程序的 bug，对界面进行美化与升级，提高对用户的友好性。同时增加用户登录密码等功能，提高安全性。组长准备答辩材料。

第四周：最后收尾，维护报告，组长答辩等。

## II.系统分析报告

### • 系统任务分析（需求分析）

首先，用户在 Atom mode 里面可以使用基本的混合四则运算，完成对运算符的优先级和结合性的操作。同时，每次用鼠标点击按键时，文本框中会出现相应的数字或者符号，点击“=”之后则在文本框中显示运算结果并覆盖之前的式子。这些需要我们对每个按键都编写正确的响应函数。另外，我们还可以不依赖按键，直接在文本框中输入想要算的式子之后，点击“=”之后直接运算出结果。对于 Atom mode 里面的三角函数运算或者对数指数运算我们将调用 C++ 的 `cmath` 库函数进行辅助运算。

其次，在 Molecule mode 里，用户可以使用了以下功能。

- 1.复数计算：复数的加减乘除运算以及复数的模，辐角的计算。
- 2.矩阵计算：矩阵的加减运算，乘法运算，行列式的计算，秩的计算。
- 3.时间换算：时间的加减运算，对于不同时区的换算，显示当前的北京时间。
- 4.积分计算：对于给定的简易积分表达式的定积分计算。
- 5.质因数分解：对输入的某数进行分解质因数运算。
- 6.日期换算：日期的加减运算，日期换算，显示当前日期。
- 7.多进制运算：包括二进制，八进制，十进制，十六进制，三十二进制等一些常规进制的相互换算。
- 8.解一元多次方程：对三次方程的求解。
9. 解多元一次方程：由于对话框的限制，此处我们仅实现了四元方程的求解，但是运用的算法可以解任意元的方程。

在 Leisure mode 中，我们将按钮与未与本程序完全封装在一起的一些小游戏进行了链接，用户可以打开进行游玩。

在 Draw mode 中，用户可以用五种颜色来绘制点，线，圆，椭圆和矩形。

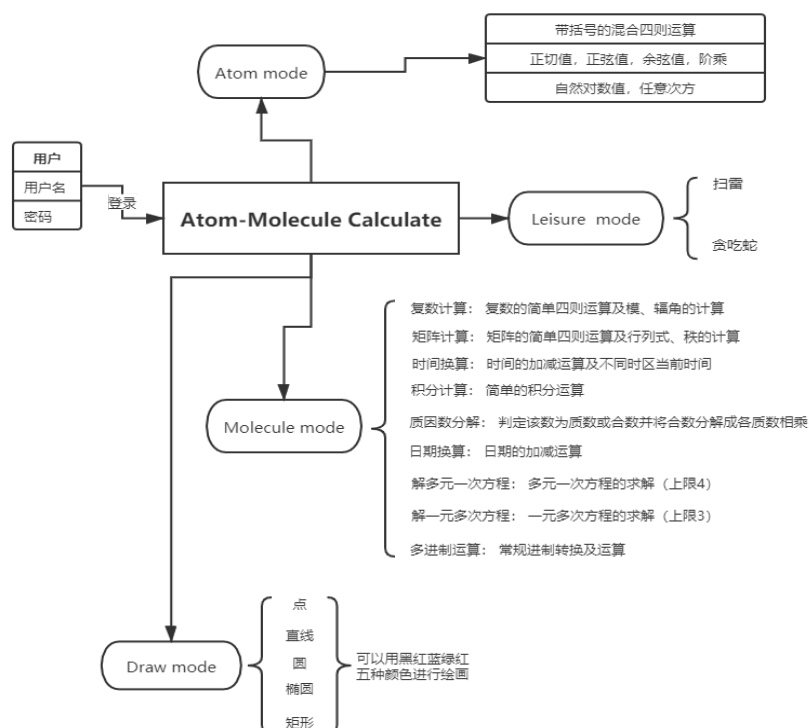
最后，我们将会为本程序编写登陆界面，完善对用户的安全性保证。同时对界面进行美化工作，提高对用户的友好性。

注：本程序内提供了使用说明供用户参考。（通过扫描二维码等途径获取）

### • 执行者分析

对于本程序来说，执行者只有使用本程序的用户。用户通过登录自己的账户来获得使用本程序的权限，从而可以使用所有的功能。

## • 用例分析



用例的确定参照上图即可。

## • 实体类的静态模型以及说明

本程序中仅包含对话框类（边界类），无实体类，故无实体类的静态模型以及说明。关于对话框类的说明会再系统设计报告中具体说明。

## III. 系统设计报告

### • 实体类设计

本程序中仅包含对话框类（边界类），无实体类，故无实体类设计部分的说明。

### • 对话框概要设计

主控对话框（IDD\_CALCULATE\_ATOM\_DIALOG）：Atom mode 功能，拥有菜单，可以切换 mode，并且有 Help，Contact us 等选项。

系统自带的关于对话框（IDD\_ABOUTBOX）：提供版本号等版权信息。

Molecule mode 对话框（IDD\_DIALOG1）：拥有多个按钮控件和对与 Molecule mode 功能的说明。

integrate 对话框（IDD\_DIALOG2）：实现简单积分运算。

Solving linear equations 对话框（IDD\_DIALOG3）：解线性齐次方程。

Solving multiple equations of one variable 对话框（IDD\_DIALOG4）：解一元高次方程。

Base conversion 对话框 (IDD\_DIALOG5)：多进制转换。

Complex number calculation 对话框 (IDD\_DIALOG6)：复数计算。

Matrix Calculation 对话框 (IDD\_DIALOG7)：矩阵计算。

Time conversion 对话框 (IDD\_DIALOG8)：时间换算，不同时区换算。

Date Conversion 对话框 (IDD\_DIALOG9)：日期计算。

Prime factor decomposition 对话框 (IDD\_DIALOG10)：分解质因数。

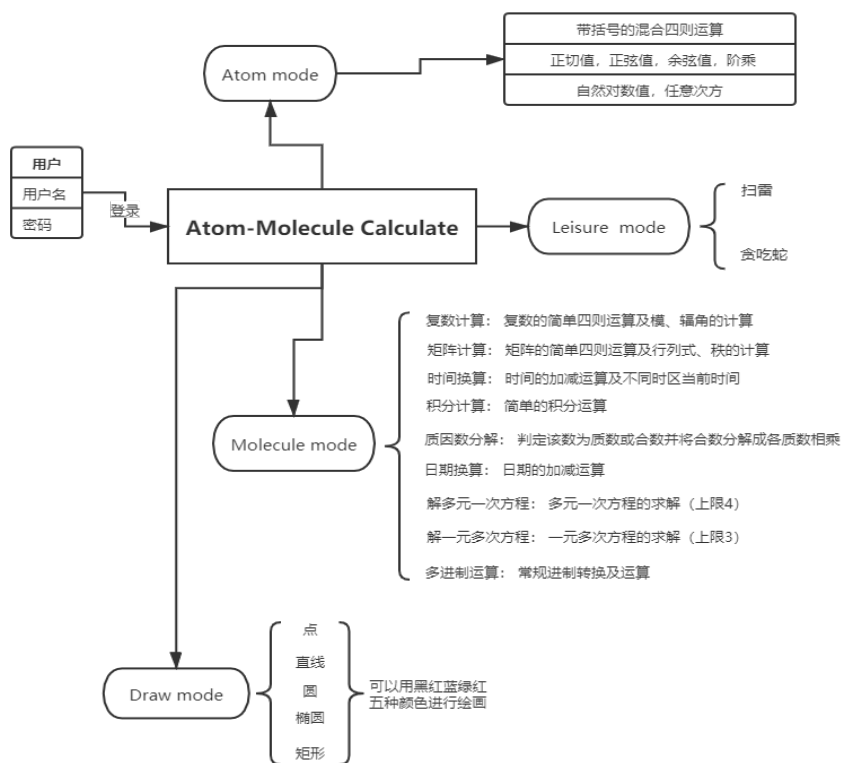
使用说明对话框 (IDD\_DIALOG11)：提供使用说明二维码。

Login 对话框 (IDD\_DIALOG12)：用户登录。

Leisure mode 对话框 (IDD\_DIALOG13)：休闲模式。

Draw mode 对话框 (IDD\_MFC\_DRAW\_DIALOG)：绘图模式。

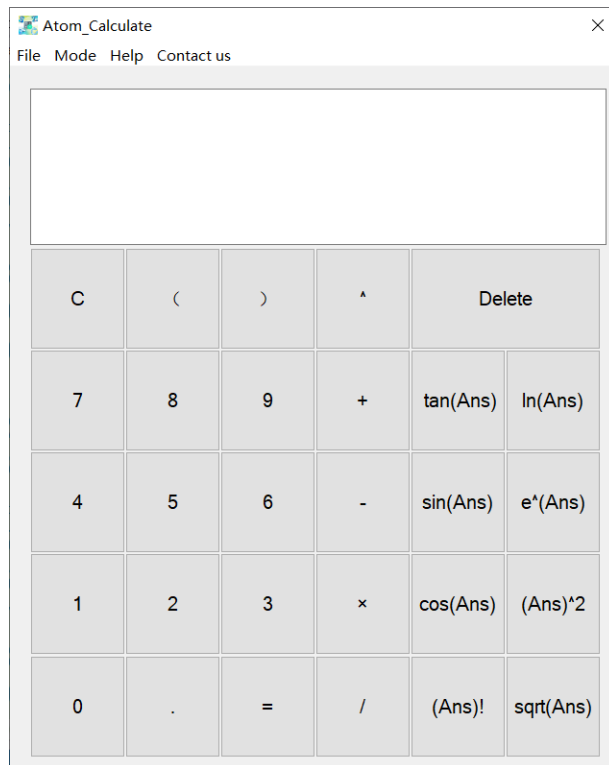
这些对话框成为一个对话框树：



## • 对话框详细设计

主控对话框（IDD\_CALCULATE\_ATOM\_DIALOG）：Atom mode 功能，拥有菜单，可以切换 mode，并且有 Help，Contact us 等选项。

外观图：



控件说明：

编辑框有一个，ID、映射变量和功能是：

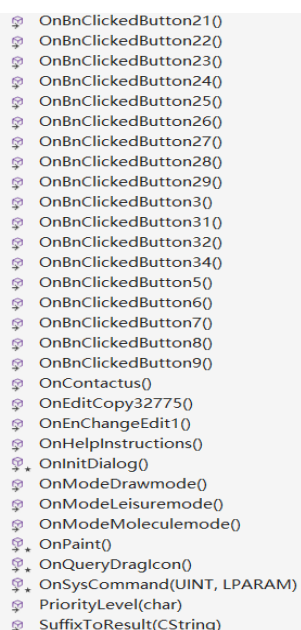
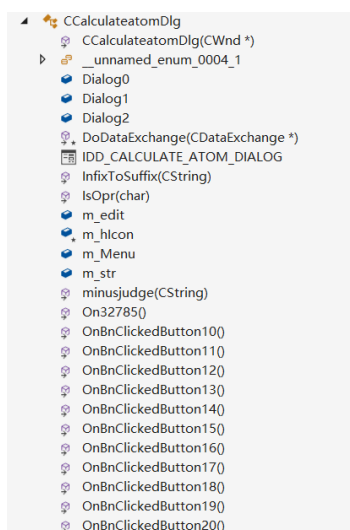
- DC\_EDIT1，计算器的显示屏幕。

命令按键有 28 个，ID、映射变量和功能是：

- DC\_BUTTON8，清除。
- DC\_BUTTON10，左括号。
- DC\_BUTTON14，右括号。
- DC\_BUTTON26，乘方。
- DC\_BUTTON5，delete。
- DC\_BUTTON3，7。
- DC\_BUTTON6，8。
- DC\_BUTTON12，9。
- DC\_BUTTON22，4。

- DC\_BUTTON15, 5。
- DC\_BUTTON17, 6。
- DC\_BUTTON27, 1。
- DC\_BUTTON21, 2。
- DC\_BUTTON28, 3。
- DC\_BUTTON23, 0。
- DC\_BUTTON32, 小数点。
- DC\_BUTTON34, 等于号。
- DC\_BUTTON31, 除法。
- DC\_BUTTON13, 乘法。
- DC\_BUTTON7, 加法。
- DC\_BUTTON11, 减法。
- DC\_BUTTON20, 对数函数。
- DC\_BUTTON19, 正弦函数。
- DC\_BUTTON16, 正切函数。
- DC\_BUTTON18, 余弦函数。
- DC\_BUTTON25, 平方。
- DC\_BUTTON9, 阶乘。
- DC\_BUTTON29, 开根号。
- DC\_BUTTON24, e 的 x 次方。

对话框类设计:





```

public:
    INT PriorityLevel(char op); //运算符分级
    BOOL IsOpr(char c);      //判断是否是运算符
    CString InfixToSuffix(CString szIn); //转为后缀式
    CString minusjudge(CString a);      //判断减号和负号
    double SuffixToResult(CString szSuffix); //算出结果

protected:
    virtual void DoDataExchange(CDataExchange* pDX);
protected:
    HICON m_hIcon;

    // 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()
public:
    afx_msg void OnEnChangeEdit1();
    CEdit m_edit;
    CString m_str;
    afx_msg void OnBnClickedButton7();
    afx_msg void OnBnClickedButton34();
    afx_msg void OnBnClickedButton8();
    afx_msg void OnBnClickedButton5();
    afx_msg void OnBnClickedButton11();
    afx_msg void OnBnClickedButton13();
    afx_msg void OnBnClickedButton31();
    afx_msg void OnBnClickedButton26();
    afx_msg void OnBnClickedButton10();
    afx_msg void OnBnClickedButton14();

```

```

afx_msg void OnBnClickedButton27();
afx_msg void OnBnClickedButton21();
afx_msg void OnBnClickedButton28();
afx_msg void OnBnClickedButton22();
afx_msg void OnBnClickedButton3();
afx_msg void OnBnClickedButton6();
afx_msg void OnBnClickedButton12();
afx_msg void OnBnClickedButton32();
afx_msg void OnBnClickedButton23();
afx_msg void OnBnClickedButton15();
afx_msg void OnBnClickedButton17();
afx_msg void OnBnClickedButton9();
afx_msg void OnBnClickedButton16();
afx_msg void OnBnClickedButton18();
afx_msg void OnBnClickedButton19();
afx_msg void OnBnClickedButton20();
afx_msg void OnBnClickedButton24();
afx_msg void OnBnClickedButton25();
afx_msg void OnBnClickedButton29();
afx_msg void OnHelpInstructions(); //菜单中的 Help
afx_msg void OnModeMoleculemode();//菜单中的 Mode—Molecule mode
afx_msg void OnModeLeisuremode();//菜单中的 Mode—Leisure mode
afx_msg void OnContactus();//菜单中的 Contact us
afx_msg void OnModeDrawmode();//菜单中的 Mode—Draw mode

```

算法说明：

Atom mode 利用了栈空间的算法，将表达式转化为后缀式之后根据运算符的优先级进行运算。兼容括号，负号，小数等。

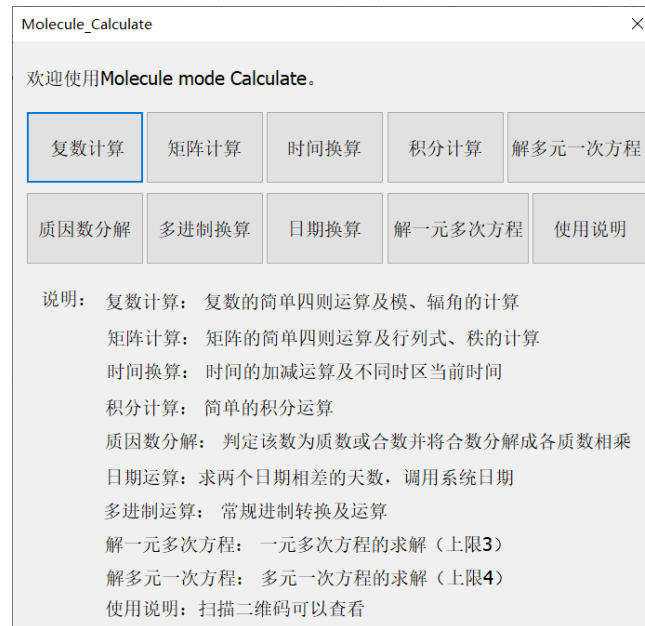
菜单则直接用 Menu 创建。

控制消息流：

可以用鼠标利用命令控件进行操作，也可以用键盘在编辑框中直接输入表达式。

Molecule mode 对话框（IDD\_DIALOG1）：拥有多个按钮控件和对与 Molecule mode 功能的说明。

外观图：



控件说明：

静态文本有 12 个，ID、映射变量和功能是：

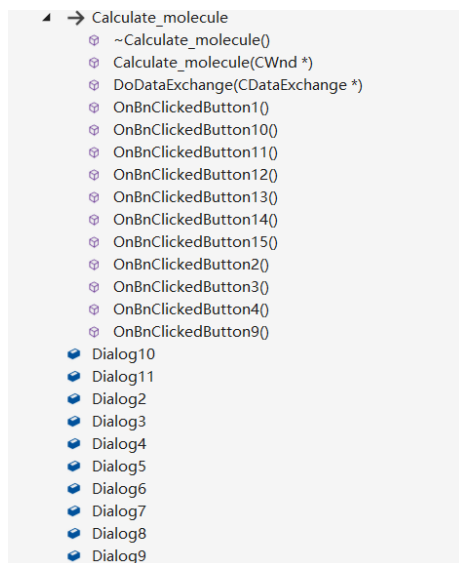
- IDC\_STATIC2，欢迎用户的话。
- IDC\_STATIC1，说明具体功能。
- IDC\_STATIC3，说明具体功能。
- IDC\_STATIC4，说明具体功能。
- IDC\_STATIC5，说明具体功能。
- IDC\_STATIC6，说明具体功能。
- IDC\_STATIC7，说明具体功能。
- IDC\_STATIC8，说明具体功能。
- IDC\_STATIC9，说明具体功能。
- IDC\_STATIC10，说明具体功能。
- IDC\_STATIC11，说明具体功能。
- IDC\_STATIC12，说明具体功能。

命令按键有 10 个，ID、映射变量和功能是：

- DC\_BUTTON1，复数计算。
- DC\_BUTTON2，矩阵计算。
- DC\_BUTTON10，时间换算。

- DC\_BUTTON4, 积分计算。
- DC\_BUTTON9, 解多元一次方程。
- DC\_BUTTON12, 质因数分解。
- DC\_BUTTON14, 多进制换算。
- DC\_BUTTON13, 日期换算。
- DC\_BUTTON11, 解一元多次方程。
- DC\_BUTTON15, 使用说明。

对话框类设计:



```
afx_msg void OnBnClickedButton1();
```

```
afx_msg void OnBnClickedButton4();
```

```
afx_msg void OnBnClickedButton3();
```

```
afx_msg void OnBnClickedButton10();
```

```
afx_msg void OnBnClickedButton9();
```

```
afx_msg void OnBnClickedButton11();
```

```
afx_msg void OnBnClickedButton14();
```

```
afx_msg void OnBnClickedButton2();
```

```
afx_msg void OnBnClickedButton13();
```

```
afx_msg void OnBnClickedButton12();
```

```
afx_msg void OnBnClickedButton15();
```

```
afx_msg void OnContactus();
```

算法说明：

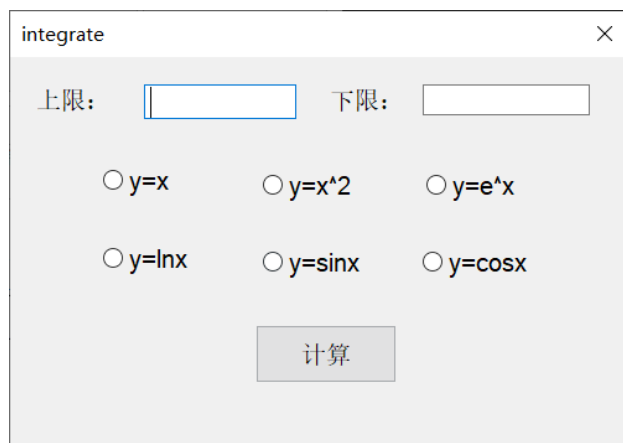
Molecule mode 直接创建了很多命令控件，用鼠标点击这些控件就可以弹出新的对话框进行进一步操作。

控制消息流：

可以用鼠标利用命令控件进行操作。

integrate 对话框（IDD\_DIALOG2）：实现简单积分运算。

外观图：



控件说明：

静态文本有 2 个，ID、映射变量和功能是：

- IDC\_STATIC1：上限。
- IDC\_STATIC2：下限。

命令按键有 1 个，ID、映射变量和功能是：

- IDC\_BUTTON1：计算。

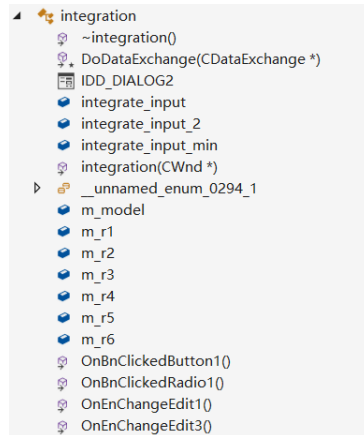
单选按钮有 6 个，ID、映射变量和功能是：

- IDC\_RADIO1：给定的函数 1。
- IDC\_RADIO2：给定的函数 2。
- IDC\_RADIO3：给定的函数 3。
- IDC\_RADIO4：给定的函数 4。
- IDC\_RADIO5：给定的函数 5。
- IDC\_RADIO6：给定的函数 6。

编辑框控件有 2 个，ID、映射变量和功能是：

- IDC\_EDIT1：上限具体值。
- IDC\_EDIT2：下限具体值。

对话框类设计：



算法说明：

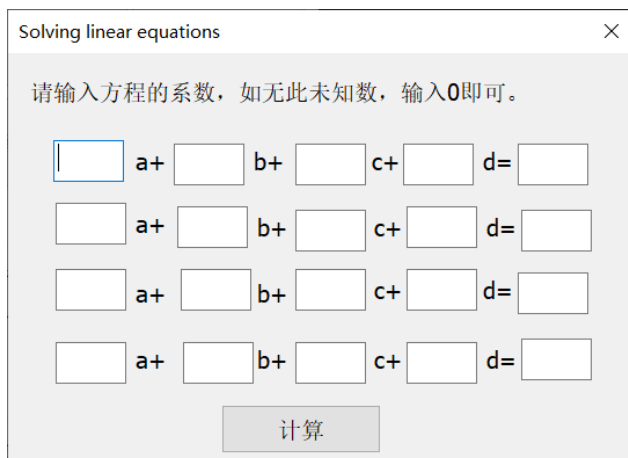
直接写出原函数进行上下限的运算。

控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

Solving linear equations 对话框（IDD\_DIALOG3）：解线性齐次方程。

外观图：



控件说明：

静态文本有 17 个，ID、映射变量和功能是：

- IDC\_STATIC1，说明具体功能。
- IDC\_STATIC2，变量。
- IDC\_STATIC3，变量。
- IDC\_STATIC4，变量。
- IDC\_STATIC5，变量。
- IDC\_STATIC6，变量。
- IDC\_STATIC7，变量。
- IDC\_STATIC8，变量。
- IDC\_STATIC9，变量。
- IDC\_STATIC10，变量。
- IDC\_STATIC11，变量。
- IDC\_STATIC12，变量。
- IDC\_STATIC13，变量。
- IDC\_STATIC14，变量。
- IDC\_STATIC15，变量。
- IDC\_STATIC16，变量。
- IDC\_STATIC17，变量。

编辑框有 20 个，ID、映射变量和功能是：

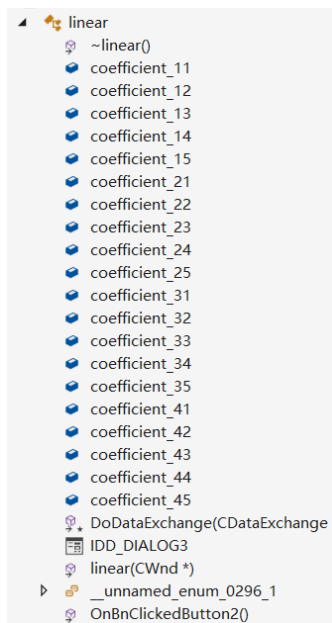
- IDC\_EDIT1：方程的系数。
- IDC\_EDIT2：方程的系数。
- IDC\_EDIT3：方程的系数。
- IDC\_EDIT4：方程的系数。
- IDC\_EDIT5：方程的系数。
- IDC\_EDIT6：方程的系数。
- IDC\_EDIT7：方程的系数。
- IDC\_EDIT8：方程的系数。
- IDC\_EDIT9：方程的系数。
- IDC\_EDIT10：方程的系数。
- IDC\_EDIT11：方程的系数。

- IDC\_EDIT12: 方程的系数。
- IDC\_EDIT13: 方程的系数。
- IDC\_EDIT14: 方程的系数。
- IDC\_EDIT15: 方程的系数。
- IDC\_EDIT16: 方程的系数。
- IDC\_EDIT17: 方程的系数。
- IDC\_EDIT18: 方程的系数。
- IDC\_EDIT19: 方程的系数。
- IDC\_EDIT20: 方程的系数。

命令按键有 1 个，ID、映射变量和功能是：

- IDC\_BUTTON1: 计算。

对话框类设计：



算法说明：

先判断系数矩阵的秩，然后利用增广矩阵的算法来化简问题，最终得到解。

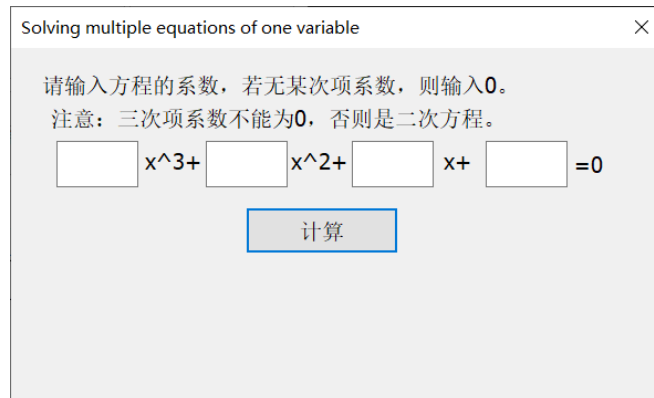
控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。



Solving multiple equations of one variable 对话框（IDD\_DIALOG4）：解一元高次方程。

外观图：



控件说明：

静态文本有 6 个，ID、映射变量和功能是：

- IDC\_STATIC1，说明具体功能。
- IDC\_STATIC2，注意事项。
- IDC\_STATIC3，x 三次方。
- IDC\_STATIC4，x 二次方。
- IDC\_STATIC5，x 一次方。
- IDC\_STATIC6，常数项。

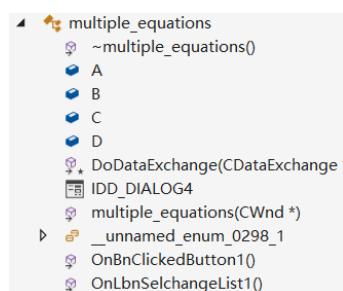
编辑框有 3 个，ID、映射变量和功能是：

- IDC\_EDIT1：方程的系数。
- IDC\_EDIT2：方程的系数。
- IDC\_EDIT3：方程的系数。

命令按键有 1 个，ID、映射变量和功能是：

- IDC\_BUTTON1：计算。

对话框类设计：



算法说明：

算法利用盛金公式为基础进行计算。

控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

Base conversion 对话框（IDD\_DIALOG5）：多进制转换。

外观图：

控件说明：

静态文本有 7 个，ID、映射变量和功能是：

- IDC\_EDIT1：十进制。
- IDC\_EDIT2：二进制。
- IDC\_EDIT3：八进制。
- IDC\_EDIT4：十六进制。
- IDC\_EDIT5：三十二进制。
- IDC\_EDIT6：三十六进制。
- IDC\_EDIT7：说明。

编辑框有 6 个，ID、映射变量和功能是：

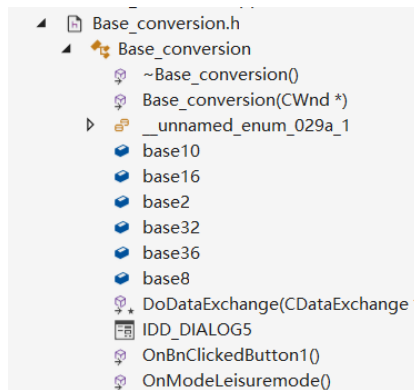
- IDC\_EDIT1：十进制的数字。
- IDC\_EDIT2：二进制的数字。
- IDC\_EDIT3：八进制的数字。
- IDC\_EDIT4：十六进制的数字。

- IDC\_EDIT5: 三十二进制的数字。
- IDC\_EDIT6: 三十六进制的数字。

命令按键有 1 个，ID、映射变量和功能是：

- IDC\_BUTTON1: 换算。

对话框类设计：



算法说明：

利用 C++内置的库函数，将十进制的数字转化为其他进制的数字。

控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

Complex number calculation 对话框（IDD\_DIALOG6）：复数计算。

外观图：

Complex number calculation

×

请输入两个操作数：

操作数1

+

i

操作数2

+

i

加法

减法

乘法

除法

操作数1的模

操作数2的模

操作数1辐角

操作数2辐角

控件说明：

静态文本有 7 个，ID、映射变量和功能是：

- IDC\_EDIT1：请输入俩操作数。
- IDC\_EDIT2：操作数 1。
- IDC\_EDIT3：操作数 2。
- IDC\_EDIT4：+。
- IDC\_EDIT5：+。
- IDC\_EDIT6：i。
- IDC\_EDIT7：i。

编辑框有 4 个，ID、映射变量和功能是：

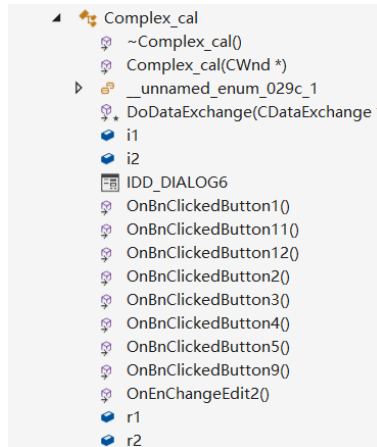
- IDC\_EDIT1：操作数 1 的实部。
- IDC\_EDIT2：操作数 1 的虚部。
- IDC\_EDIT3：操作数 2 的实部。
- IDC\_EDIT4：操作数 2 的虚部。

命令按键有 8 个，ID、映射变量和功能是：

- IDC\_BUTTON1：加法。
- IDC\_BUTTON2：减法。
- IDC\_BUTTON3：乘法。
- IDC\_BUTTON4：除法。
- IDC\_BUTTON5：操作数 1 的模。
- IDC\_BUTTON6：操作数 2 的模。
- IDC\_BUTTON7：操作数 1 的辐角。

- IDC\_BUTTON8: 操作数 2 的辐角。

对话框类设计:



算法说明:

利用高等数学中的复数的知识，进行编程。

控制消息流:

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

Matrix Calculation 对话框 (IDD\_DIALOG7): 矩阵计算。

外观图：

Matrix Calculation

×

请输入矩阵1和矩阵2的系数：

a11

a12

b11

b12

a21

a22

b21

b22

乘法

加法

减法

转置

矩阵1的行列式

矩阵2的行列式

矩阵1的秩

矩阵2的秩

控件说明：

静态文本有 9 个，ID、映射变量和功能是：

- IDC\_EDIT1：操作说明。
- IDC\_EDIT2：矩阵系数。
- IDC\_EDIT3：矩阵系数。
- IDC\_EDIT4：矩阵系数。
- IDC\_EDIT5：矩阵系数。
- IDC\_EDIT6：矩阵系数。
- IDC\_EDIT7：矩阵系数。
- IDC\_EDIT8：矩阵系数。
- IDC\_EDIT9：矩阵系数。

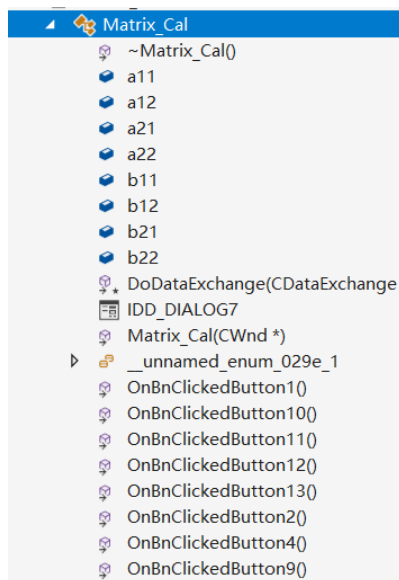
编辑框有 8 个，ID、映射变量和功能是：

- IDC\_EDIT1：矩阵 1 的系数。
- IDC\_EDIT2：矩阵 1 的系数。
- IDC\_EDIT3：矩阵 1 的系数。
- IDC\_EDIT4：矩阵 1 的系数。
- IDC\_EDIT5：矩阵 2 的系数。
- IDC\_EDIT6：矩阵 2 的系数。
- IDC\_EDIT7：矩阵 2 的系数。
- IDC\_EDIT8：矩阵 2 的系数。

命令按键有 8 个，ID、映射变量和功能是：

- IDC\_BUTTON2: 加法。
- IDC\_BUTTON3: 减法。
- IDC\_BUTTON1: 乘法。
- IDC\_BUTTON4: 转置。
- IDC\_BUTTON5: 矩阵 1 的行列式。
- IDC\_BUTTON6: 矩阵 2 的行列式。
- IDC\_BUTTON7: 矩阵 1 的秩。
- IDC\_BUTTON8: 矩阵 2 的秩。

对话框类设计:



算法说明:

利用线性中的矩阵的知识，进行编程。

控制消息流:

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

Time conversion 对话框 (IDD\_DIALOG8): 时间换算，不同时区换算。

外观图：

Time conversion

请输入时间：

时  分  秒

请输入需要加减的时间：

时  分  秒

换算输入的时间    计算加减后的时间    北京时间

东京时间    伦敦时间    华盛顿时间

控件说明：

静态文本有 8 个，ID、映射变量和功能是：

- IDC\_EDIT1：请输入时间：。
- IDC\_EDIT2：时。
- IDC\_EDIT3：分。
- IDC\_EDIT4：秒。
- IDC\_EDIT5：请输入需要加减的时间。
- IDC\_EDIT6：时。
- IDC\_EDIT7：分。
- IDC\_EDIT8：秒。

编辑框有 6 个，ID、映射变量和功能是：

- IDC\_EDIT1：时刻 1 的时。
- IDC\_EDIT2：时刻 1 的分。
- IDC\_EDIT3：时刻 1 的秒。
- IDC\_EDIT4：时刻 2 的时。
- IDC\_EDIT5：时刻 2 的分。
- IDC\_EDIT6：时刻 2 的秒。

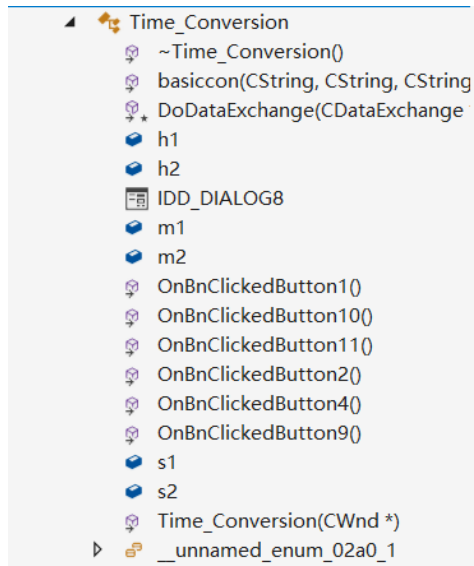
命令按键有 6 个，ID、映射变量和功能是：

- IDC\_BUTTON2：换算输入的时间。
- IDC\_BUTTON3：计算加减后的时间。
- IDC\_BUTTON1：北京时间。



- IDC\_BUTTON4: 东京时间。
- IDC\_BUTTON5: 伦敦时间。
- IDC\_BUTTON6: 华盛顿时间。

对话框类设计:



算法说明:

利用 mod 的算法, 对时间进行处理之后运算。对于系统时间, 则调用 Ctime 库函数进行编程。

控制消息流:

输入需要求解的信息后, 可以用鼠标利用命令控件进行操作。

Date Conversion 对话框 (IDD\_DIALOG9): 日期计算。

外观图：



The image shows a 'Date Conversion' dialog box with a title bar containing a close button (X). The dialog contains two input sections. The first section is labeled '请输入日期：' (Please enter a date:) and has three input fields for year, month, and day, each followed by its respective unit character (年, 月, 日). The second section is labeled '请输入第二个日期：' (Please enter the second date:) and also has three input fields for year, month, and day, each followed by its respective unit character. At the bottom of the dialog, there are two buttons: '计算相差的天数' (Calculate the number of days difference) and '当前日期' (Current date).

控件说明：

静态文本有 8 个，ID、映射变量和功能是：

- IDC\_EDIT1: 请输入日期：。
- IDC\_EDIT2: 年。
- IDC\_EDIT3: 月。
- IDC\_EDIT4: 日。
- IDC\_EDIT5: 请输入第二个日期：。
- IDC\_EDIT6: 年。
- IDC\_EDIT7: 月。
- IDC\_EDIT8: 日。

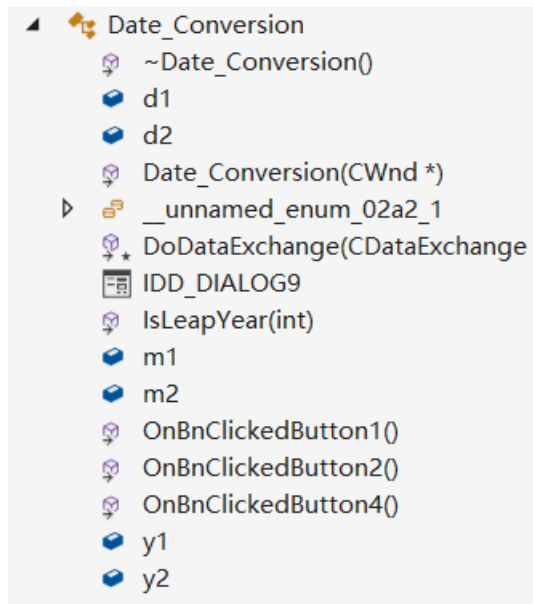
编辑框有 6 个，ID、映射变量和功能是：

- IDC\_EDIT1: 日期 1 的年。
- IDC\_EDIT2: 日期 1 的月。
- IDC\_EDIT3: 日期 1 的日。
- IDC\_EDIT4: 日期 2 的年。
- IDC\_EDIT5: 日期 2 的月。
- IDC\_EDIT6: 日期 2 的日。

命令按键有 2 个，ID、映射变量和功能是：

- IDC\_BUTTON1: 计算相差天数。
- IDC\_BUTTON2: 当前日期。

对话框类设计：



算法说明：

采用包含判断是否是闰年的算法，对日期进行处理之后运算。对于系统日期，则调用 Ctime 库函数进行编程。

控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

Prime factor decomposition 对话框（IDD\_DIALOG10）：分解质因数。

外观图：



静态文本有 1 个，ID、映射变量和功能是：

- IDC\_EDIT1：请输入需要分解的数字：。

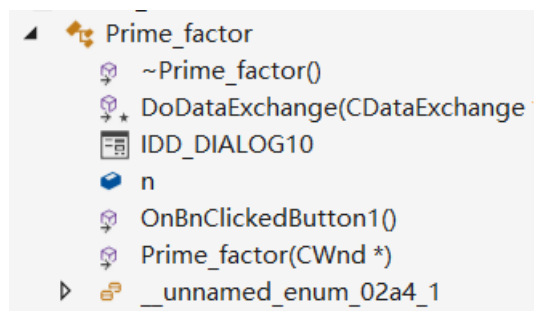
编辑框有 1 个，ID、映射变量和功能是：

- IDC\_EDIT1：用户输入的数字。

命令按键有 1 个，ID、映射变量和功能是：

- IDC\_BUTTON1：分解。

对话框类设计：



算法说明：

采用循环的算法，直接分解。

控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

使用说明对话框（IDD\_DIALOG11）：提供使用说明二维码。

外观图：



控件说明：

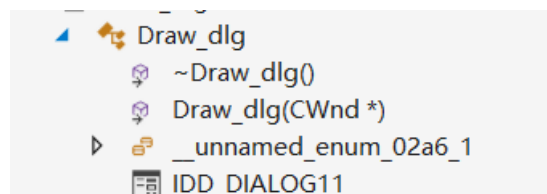
静态文本有 1 个，ID、映射变量和功能是：

- IDC\_EDIT1：请扫描以下二维码，下载并阅读使用说明文件。

Picture control 有 1 个，ID、映射变量和功能是：

- IDC\_STATIC1 (Picture Control)：二维码。

对话框类设计：



算法说明：

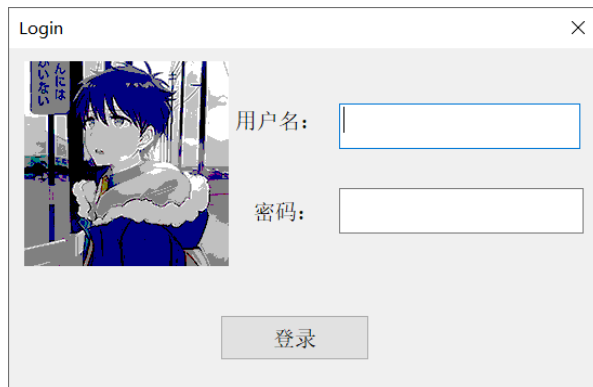
无算法。

控制消息流：

无。

Login 对话框（IDD\_DIALOG12）：用户登录。

外观图：



控件说明：

静态文本有 2 个，ID、映射变量和功能是：

- IDC\_EDIT1：用户名。
- IDC\_EDIT2：登录。

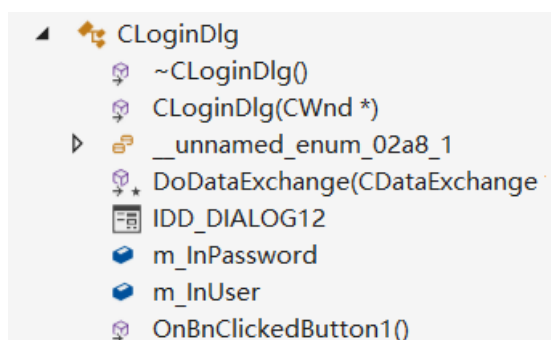
Picture control 有 1 个，ID、映射变量和功能是：

- IDC\_STATIC1 (Picture Control)：图片。

命令按键有 1 个，ID、映射变量和功能是：

- IDC\_BUTTON1：登录。

对话框类设计：



算法说明：

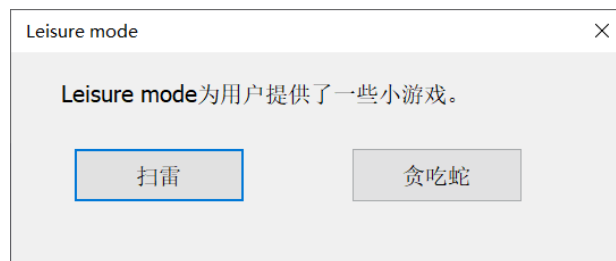
在内部代码中先写好用户的信息，确认后进行比较信息，若成功，则允许进入主程序。

控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

Leisure mode 对话框（IDD\_DIALOG13）：休闲模式。

外观图：



控件说明：

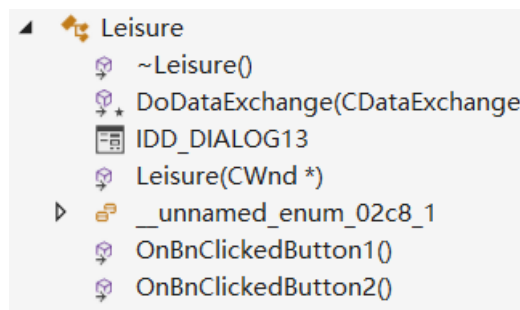
静态文本有 1 个，ID、映射变量和功能是：

- IDC\_EDIT1：功能说明。

命令按键有 2 个，ID、映射变量和功能是：

- IDC\_BUTTON1：扫雷。
- IDC\_BUTTON1：贪吃蛇。

对话框类设计：



算法说明：

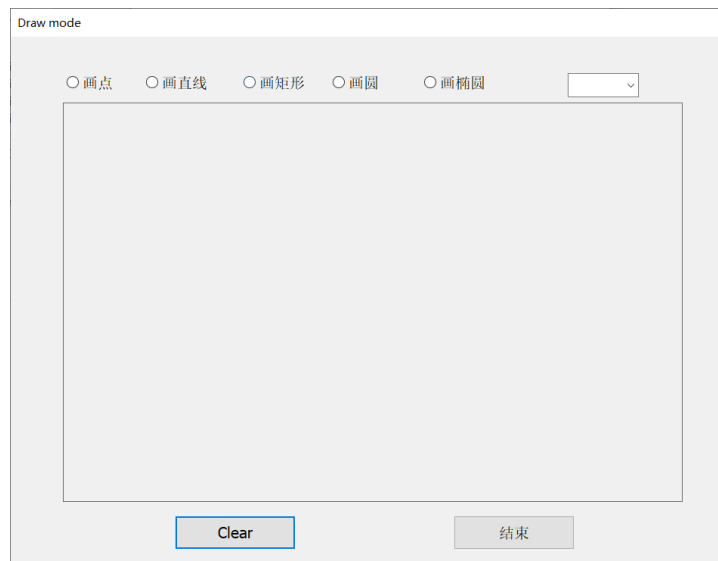
利用了外部的 exe 文件，直接进行连接调用。

控制消息流：

可以用鼠标利用命令控件进行操作。

Draw mode 对话框 (IDD\_MFC\_DRAW\_DIALOG)：绘图模式。

外观图：



控件说明：

命令按键有 2 个，ID、映射变量和功能是：

- IDC\_BUTTON1：Clear。
- IDC\_BUTTON1：结束。

单选按钮有 5 个，ID、映射变量和功能是：

- IDC\_RADIO1：画点。
- IDC\_RADIO2：画直线。
- IDC\_RADIO3：画矩形。
- IDC\_RADIO4：画圆。
- IDC\_RADIO5：画椭圆。

Combo-box Control 有 1 个，ID、映射变量和功能是：

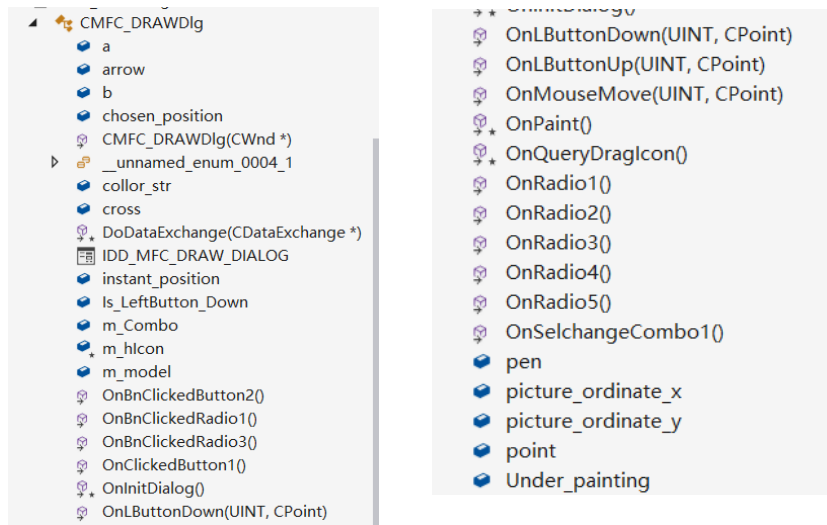
IDC\_COMBO1：画笔颜色。

Picture Control 有 1 个，ID、映射变量和功能是：

IDC\_STATIC1：画板。



对话框类设计:



算法说明:

创建了一些变量, 然后利用坐标进行操作。

控制消息流:

可以用鼠标利用控件进行操作。

#### • 对话框数据组织

这里给出所有的头文件:

```
class Base_conversion : public CDialogEx
{
    DECLARE_DYNAMIC(Base_conversion)

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

    // 对话框数据

#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG5 };
#endif
```

```

#endif

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

    DECLARE_MESSAGE_MAP()
public:
    afx_msg void OnModeLeisuremode();

    CString base10;
    CString base16;
    CString base2;
    CString base32;
    CString base8;
    CString base36;

    afx_msg void OnBnClickedButton1();
};

// Calculate_atom.h: PROJECT_NAME 应用程序的主头文件
//

#pragma once

#ifndef __AFXWIN_H__
    #error "在包含此文件之前包含 'pch.h' 以生成 PCH"
#endif

#include "resource.h"    // 主符号

// CCalculateatomApp:
// 有关此类的实现，请参阅 Calculate_atom.cpp
//

class CCalculateatomApp : public CWinApp

```

```

{
public:
    CCalculateatomApp();

// 重写
public:
    virtual BOOL InitInstance();

// 实现

    DECLARE_MESSAGE_MAP()
};

// Calculate_atomDlg.h: 头文件
//

#pragma once

//////////防止变成 WIN95 风格!!!

#ifdef _M_IX86

#pragma comment(linker, "/manifestdependency:\"type='win32'
name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='x86'
publicKeyToken='6595b64144ccf1df' language='*\\'")

#elif defined _M_IA64

#pragma comment(linker, "/manifestdependency:\"type='win32'
name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='ia64'
publicKeyToken='6595b64144ccf1df' language='*\\'")

#elif defined _M_X64

#pragma comment(linker, "/manifestdependency:\"type='win32'
name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='amd64'
publicKeyToken='6595b64144ccf1df' language='*\\'")

#else

```

```

#pragma comment(linker, "/manifestdependency:\"type='win32'
name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='*'
publicKeyToken='6595b64144ccf1df' language='*\\")
#endif

#include"Calculate_molecule.h"

#include"integration.h"

#include"CLoginDlg.h"

#include"Leisure.h"

// CCalculateatomDlg 对话框

class CCalculateatomDlg : public CDialogEx
{
// 构造

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

    CMenu m_Menu;

    Calculate_molecule Dialog1;

    Leisure Dialog0;

    integration Dialog2;

// 对话框数据

#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_CALCULATE_ATOM_DIALOG };
#endif

public:
    INT PriorityLevel(char op);

    BOOL IsOpr(char c);

    CString InfixToSuffix(CString szIn);

    CString minusjudge(CString a);

    double SuffixToResult(CString szSuffix);

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

```

```

// 实现

protected:
    HICON m_hIcon;

    // 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()

public:
    afx_msg void OnEnChangeEdit1();
    CEdit m_edit;
    CString m_str;
    afx_msg void OnBnClickedButton7();
    afx_msg void OnBnClickedButton34();
    afx_msg void OnBnClickedButton8();
    afx_msg void OnBnClickedButton5();
    afx_msg void OnBnClickedButton11();
    afx_msg void OnBnClickedButton13();
    afx_msg void OnBnClickedButton31();
    afx_msg void OnBnClickedButton26();
    afx_msg void OnBnClickedButton10();
    afx_msg void OnBnClickedButton14();
    afx_msg void OnBnClickedButton27();
    afx_msg void OnBnClickedButton21();
    afx_msg void OnBnClickedButton28();
    afx_msg void OnBnClickedButton22();
    afx_msg void OnBnClickedButton3();
    afx_msg void OnBnClickedButton6();

```

```

afx_msg void OnBnClickedButton12();
afx_msg void OnBnClickedButton32();
afx_msg void OnBnClickedButton23();
afx_msg void OnBnClickedButton15();
afx_msg void OnBnClickedButton17();
afx_msg void OnBnClickedButton9();
afx_msg void OnBnClickedButton16();
afx_msg void OnBnClickedButton18();
afx_msg void OnBnClickedButton19();
afx_msg void OnBnClickedButton20();
afx_msg void OnBnClickedButton24();
afx_msg void OnBnClickedButton25();
afx_msg void OnBnClickedButton29();
afx_msg void OnHelpInstructions();
afx_msg void OnEditCopy32775();
afx_msg void OnModeMoleculemode();
afx_msg void OnModeLeisuremode();
afx_msg void OnContactus();
afx_msg void On32785();
afx_msg void OnModeDrawmode();
};

#include "integration.h"

// Calculate_molecule 对话框

class Calculate_molecule : public CDialog
{
    DECLARE_DYNAMIC(Calculate_molecule)

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

```

```

        integration dialog2;

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG1 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    CMenu m2_Menu;
    afx_msg void OnBnClickedButton1();
    afx_msg void OnBnClickedButton4();
    afx_msg void OnBnClickedButton3();
    afx_msg void OnBnClickedButton10();
    afx_msg void OnBnClickedButton9();
    afx_msg void OnBnClickedButton11();
    afx_msg void OnBnClickedButton14();
    afx_msg void OnBnClickedButton2();
    afx_msg void OnBnClickedButton13();
    afx_msg void OnBnClickedButton12();
    afx_msg void OnBnClickedButton15();
    afx_msg void OnContactus();

};

// CLoginDlg 对话框

class CLoginDlg : public CDialogEx

```

```

{
    DECLARE_DYNAMIC(CLoginDlg)

public:
    CLoginDlg(CWnd* pParent = nullptr);    // 标准构造函数
    virtual ~CLginDlg();
// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG12 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    CString m_InUser;
    CString m_InPassword;
    afx_msg void OnBnClickedButton1();
    //afx_msg void OnPaint();
};

// Complex_cal 对话框

class Complex_cal : public CDialogEx
{
    DECLARE_DYNAMIC(Complex_cal)

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

```



```

// 对话框数据

#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG6 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    CString r1;
    CString r2;
    afx_msg void OnEnChangeEdit2();
    CString i1;
    CString i2;
    afx_msg void OnBnClickedButton4();
    afx_msg void OnBnClickedButton9();
    afx_msg void OnBnClickedButton1();
    afx_msg void OnBnClickedButton2();
    afx_msg void OnBnClickedButton3();
    afx_msg void OnBnClickedButton5();
    afx_msg void OnBnClickedButton11();
    afx_msg void OnBnClickedButton12();
};

// Date_Conversion 对话框

class Date_Conversion : public CDialogEx
{
    DECLARE_DYNAMIC(Date_Conversion)

```

```

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

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG9 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    afx_msg void OnBnClickedButton2();
    bool IsLeapYear(int year);
    CString y1;
    CString m1;
    CString d1;
    CString y2;
    CString m2;
    CString d2;
    afx_msg void OnBnClickedButton1();
    afx_msg void OnBnClickedButton4();
};

// Draw_dlg 对话框

class Draw_dlg : public CDialogEx
{

```

```

        DECLARE_DYNAMIC(Draw_dlg)

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

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG11 };
#endif

protected:
    DECLARE_MESSAGE_MAP()
public:

};

// integration 对话框

class integration : public CDialogEx
{
    DECLARE_DYNAMIC(integration)

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

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG2 };
#endif

```

```

protected:

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

    DECLARE_MESSAGE_MAP()
public:

    CString integrate_input;
    afx_msg void OnBnClickedButton1();
    afx_msg void OnEnChangeEdit1();
    CString integrate_input_2;
    afx_msg void OnEnChangeEdit3();
    CString integrate_input_min;
    BOOL m_r1;
    BOOL m_r2;
    BOOL m_r3;
    BOOL m_r4;
    BOOL m_r5;
    BOOL m_r6;
    int m_model;
    afx_msg void OnBnClickedRadio1();
};

// Leisure 对话框

class Leisure : public CDialogEx
{
    DECLARE_DYNAMIC(Leisure)

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

```

```

// 对话框数据

#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG13 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    afx_msg void OnBnClickedButton1();
    afx_msg void OnBnClickedButton2();
};

// linear 对话框

class linear : public CDialogEx
{
    DECLARE_DYNAMIC(linear)

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

// 对话框数据

#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG3 };
#endif

protected:

```

```

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

DECLARE_MESSAGE_MAP()
public:
    CString coefficient_11;
    CString coefficient_12;
    CString coefficient_13;
    CString coefficient_14;
    CString coefficient_15;
    CString coefficient_21;
    CString coefficient_22;
    CString coefficient_23;
    CString coefficient_24;
    CString coefficient_25;
    CString coefficient_31;
    CString coefficient_32;
    CString coefficient_33;
    CString coefficient_34;
    CString coefficient_35;
    CString coefficient_41;
    CString coefficient_42;
    CString coefficient_43;
    CString coefficient_44;
    CString coefficient_45;
    afx_msg void OnBnClickedButton2();
};

// Matrix_Cal 对话框

class Matrix_Cal : public CDialogEx
{

```

```

DECLARE_DYNAMIC(Matrix_Cal)

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

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG7 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    CString a11;
    CString a12;
    CString a21;
    CString a22;
    CString b21;
    CString b11;
    CString b12;
    CString b22;
    afx_msg void OnBnClickedButton1();
    afx_msg void OnBnClickedButton2();
    afx_msg void OnBnClickedButton4();
    afx_msg void OnBnClickedButton13();
    afx_msg void OnBnClickedButton9();
    afx_msg void OnBnClickedButton10();
    afx_msg void OnBnClickedButton11();

```

```

        afx_msg void OnBnClickedButton12();
};

// multiple_equations 对话框

class multiple_equations : public CDialogEx
{
    DECLARE_DYNAMIC(multiple_equations)

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

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG4 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    afx_msg void OnBnClickedButton1();
    CString A;
    CString B;
    CString C;
    CString D;
    afx_msg void OnLbnSelchangeList1();
};

```



```

// Prime_factor 对话框

class Prime_factor : public CDialogEx
{
    DECLARE_DYNAMIC(Prime_factor)

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

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG10 };
#endif

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

    DECLARE_MESSAGE_MAP()
public:
    CString n;
    afx_msg void OnBnClickedButton1();
};

// MFC_DRAWDlg.h : 头文件
//
//////////防止变成 WIN95 风格!!!

#ifdef defined _M_IX86

#pragma comment(linker, "/manifestdependency:\"type='win32' name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='x86' publicKeyToken='6595b64144ccf1df' language='*'\")

#elif defined _M_IA64

```

```

#pragma                                comment(linker,"/manifestdependency:\"type='win32'
name='Microsoft.Windows.Common-Controls' version='6.0.0.0' processorArchitecture='ia64'
publicKeyToken='6595b64144ccf1df' language='*\\'")
#elif defined _M_X64
#pragma                                comment(linker,"/manifestdependency:\"type='win32'
name='Microsoft.Windows.Common-Controls'                                version='6.0.0.0'
processorArchitecture='amd64' publicKeyToken='6595b64144ccf1df' language='*\\'")
#else
#pragma                                comment(linker,"/manifestdependency:\"type='win32'
name='Microsoft.Windows.Common-Controls'    version='6.0.0.0'    processorArchitecture='*\\'
publicKeyToken='6595b64144ccf1df' language='*\\'")
#endif
#pragma once
#include "afxwin.h"

// Time_Conversion 对话框

class Time_Conversion : public CDialogEx
{
    DECLARE_DYNAMIC(Time_Conversion)

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

// 对话框数据
#ifdef AFX_DESIGN_TIME
    enum { IDD = IDD_DIALOG8 };
#endif

protected:

```

```

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

    DECLARE_MESSAGE_MAP()
public:
    afx_msg void OnBnClickedButton2();

    CString h1;

    CString m1;

    CString s1;

    CString h2;

    CString m2;

    CString s2;

    void Time_Conversion::basiccon(CString a, CString b, CString c);

    afx_msg void OnBnClickedButton1();

    afx_msg void OnBnClickedButton4();

    afx_msg void OnBnClickedButton9();

    afx_msg void OnBnClickedButton10();

    afx_msg void OnBnClickedButton11();

};

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

// 对话框数据
    enum { IDD = IDD_MFC_DRAW_DIALOG };

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

```

```

// 实现
protected:
    HICON m_hIcon;

    // 生成的消息映射函数
    virtual BOOL OnInitDialog();
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    DECLARE_MESSAGE_MAP()
public:
    afx_msg void OnMouseMove(UINT nFlags, CPoint point);
    int m_model;
    bool Is_LeftButton_Down;
    bool Under_painting;
    int picture_ordinate_x;
    int picture_ordinate_y;
    int a,b;                //画椭圆的长半轴和短半轴
    CPoint chosen_position;
    CPoint instant_position;
    CPoint point;
    HCURSOR cross;
    HCURSOR arrow;
    CString collar_str;
    CPen pen;

    afx_msg void OnRadio1();
    afx_msg void OnRadio2();
    // afx_msg void OnNcLButtonDown(UINT nHitTest, CPoint point);
    afx_msg void OnLButtonDown(UINT nFlags, CPoint point);

```

```
afx_msg void OnLButtonUp(UINT nFlags, CPoint point);  
afx_msg void OnRadio3();  
afx_msg void OnRadio4();  
afx_msg void OnRadio5();  
// afx_msg BOOL OnEraseBkgnd(CDC* pDC);  
// afx_msg BOOL OnEraseBkgnd(CDC* pDC);  
// afx_msg BOOL OnEraseBkgnd(CDC* pDC);  
// afx_msg BOOL OnEraseBkgnd(CDC* pDC);  
afx_msg void OnClickedButton1();  
afx_msg void OnSelchangeCombo1();  
CComboBox m_Combo;  
afx_msg void OnBnClickedRadio1();  
afx_msg void OnBnClickedRadio3();  
afx_msg void OnBnClickedButton2();  
};
```

以上就是所有的头文件了，数据储存情况可以自行查看。

#### IV.课程设计日志

2020 年 8 月 31 号      在教室上课

孙寒石：学习 MFC 程序的基本内容，了解一些 MFC 常识；

张扬：学习一些关于 mfc 的基本内容，了解一些 mfc 常识；

陶星宇：学习有关 MFC 的基本内容，了解一些 MFC 常识，并在课后进行了初步资料查询；

车旭明：学习一些关于 mfc 的基本内容，了解一些 mfc 常识；

2020 年 9 月 3 号      上机课                      机房

孙寒石：上机学习视频资料以及电子书的一些基本知识，看 MFC 视频与相关书籍资料，系统学习 MFC 编程的大致流程，包括窗口的实现与应用，消息映射机制等初步尝试建立 mfc 程序以及尝试一些简单的 MFC 操作。

张扬：上机学习视频资料以及电子书的一些基本知识，初步尝试建立 mfc 程序以及尝试一些简单的 mfc 操作。

陶星宇：上机通过视频资料以及电子书初步了解 MFC，初步尝试建立 MFC 程序以及尝试一些简单的 MFC 操作。

车旭明：上机学习视频资料以及电子书的一些基本知识，看 MFC 视频与相关书籍资料，系统学习 MFC 编程的大致流程，包括窗口的实现与应用，消息映射机制等初步尝试建立 mfc 程序以及尝试一些简单的 mfc 操作。

2020 年 9 月 5 号      宿舍集体商讨交流

孙寒石：开题，与组员进行交流，确定了本组的开题项目，并且开始分配任务。

张扬：交流一些关于学习的知识困难之处，交流各自疑惑等等，确定 mfc 项目为计算机项目。

陶星宇：交流 MFC 有关内容，交流各自疑惑等等，确定计算器为小组 MFC 项目。

车旭明：交流一些关于学习的知识困难之处，交流各自疑惑等等，确定 mfc 项目为计算机项目。

2020 年 9 月 6 号      宿舍

孙寒石：开始撰写任务建议书，并设想系统。

2020 年 9 月 7 号      上机课                      机房

孙寒石：续学习视频资料，上机练手 MFC， 上网寻找资料，借鉴学习别人计算机程序，初步与舍友合作建立 MFC 计算器程序。

张扬：继续学习视频资料，上机练手 mfc， 上网寻找资料，借鉴学习别人计算机程序，初步与舍友合作建立 mfc 计算器程序。

陶星宇：继续学习视频资料，上机练手 MFC， 上网寻找资料，借鉴学习别人计算机程序，初步与舍友合作建立 MFC 计算器程序。

车旭明：继续学习视频资料，上机练手 mfc， 上网寻找资料，借鉴学习别人计算机程序，初步与舍友合作建立 mfc 计算器程序。

2020 年 9 月 8-13 号 集体制作

孙寒石：先将组员所负责模块的对话框界面和框架设计完成，随后完成了 Atom mode 的代码，开始探究 Draw mode 和 Molecule mode。

任务建议书的提交，开始构思系统分析报告和系统设计报告。

张扬：帮助检测 BUG 情况，并着手开始做自己负责部分的 Molecule 的代码。

陶星宇：帮助检测 BUG 情况，并着手开始做自己负责部分的 Molecule 的代码。

车旭明：帮助检测 BUG 情况，并着手开始做自己负责部分的 Molecule 的代码。

2020 年 9 月 14 号 上机课 机房

孙寒石：维护界面，完善自己负责部分的 Molecule 的代码。开始编写 Draw mode 和 Leisure mode 的代码。

张扬：开始实现计算器的附加功能，并编写完善计算器的日期换算和时间换算的部分

陶星宇：实现计算器的额外附加功能，并编写完善计算器的多元一次函数的部分，及画图中椭圆部分

车旭明：开始实现计算器的附加功能，并编写完善积分运算和多进制换算的部分

2020 年 9 月 15-20 号 集体制作及商讨

孙寒石：测试程序鲁棒性，并对 BUG 部分进行了修改，比如：Atom mode 中，无法正确判断减号和负号；齐次线性方程无法解决不满秩的情况等等。维护报告，提交系统分析报告和系统设计报告。

张扬：完成并完善计算器的时间换算和日期换算部分，修复一些 bug（无法正常显示负数结果输出，无法实现小日期减去大日期）

陶星宇：完成并完善计算器的多元一次函数部分，修复些许 bug（如出现部分错误解和乱码情况），绘制了计算器程序简单图标。

车旭明：完成并完善积分运算和多进制换算部分，修复一些 bug

2020 年 9 月 21 号            上机课            机房

孙寒石：准备 ppt，准备答辩材料，准备答辩稿子，审阅组员的报告

张扬：完成并完善个人日志报告和个人报告

陶星宇：完成并完善个人日志报告和个人报告

车旭明：完成并完善个人日志报告和个人报告



V.程序说明书。

## Atom-Molecule Calculate 计算器使用说明书

### 一、设计说明：

**1.1 功能简介：**该计算机可以实现加减乘除，乘方，对数，正余弦等基本运算和混合运算。还加入了复数计算，矩阵计算，时间换算，积分计算，质因数分解等功能，可以实现此类运算的简单计算。同时计算机还附加了 **Draw mode**（画画功能）和 **Leisure mode**（休闲小游戏），可以画一些简单的几何图形，游玩扫雷和贪吃蛇小游戏。计算机功能较为全面，用户友好性搞，且具有一定的趣味性。

**1.2 运行环境：**该程序基于 windows 系统，需要有 MFC 组件的编译器来启动。

### 二、登录操作及主界面介绍

#### 2.1 登录界面

运行程序后，弹出登陆界面如下：



## 2.2 计算机主界面

登录以后进入计算机主界面，如下示：

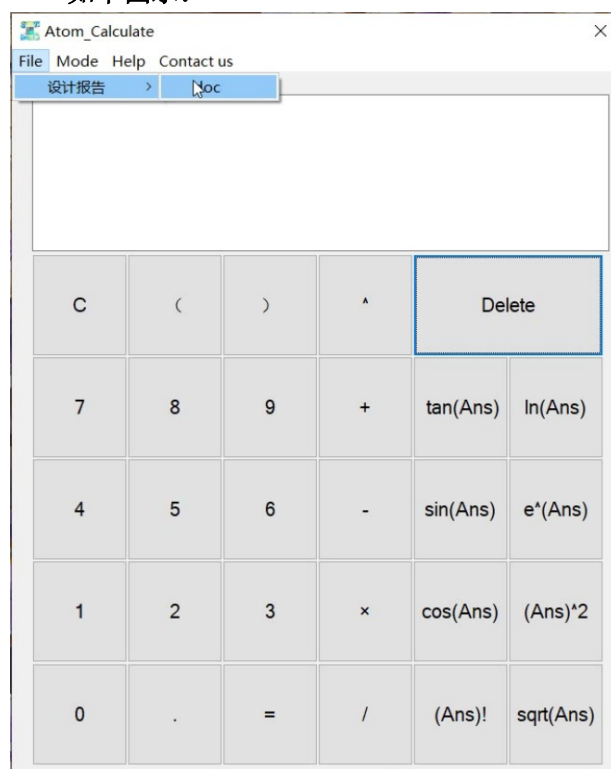


计算机主界面可以加减乘除，乘方，对数，正余弦等基本运算和混合运算。

## 2.3 计算机功能及属性菜单栏

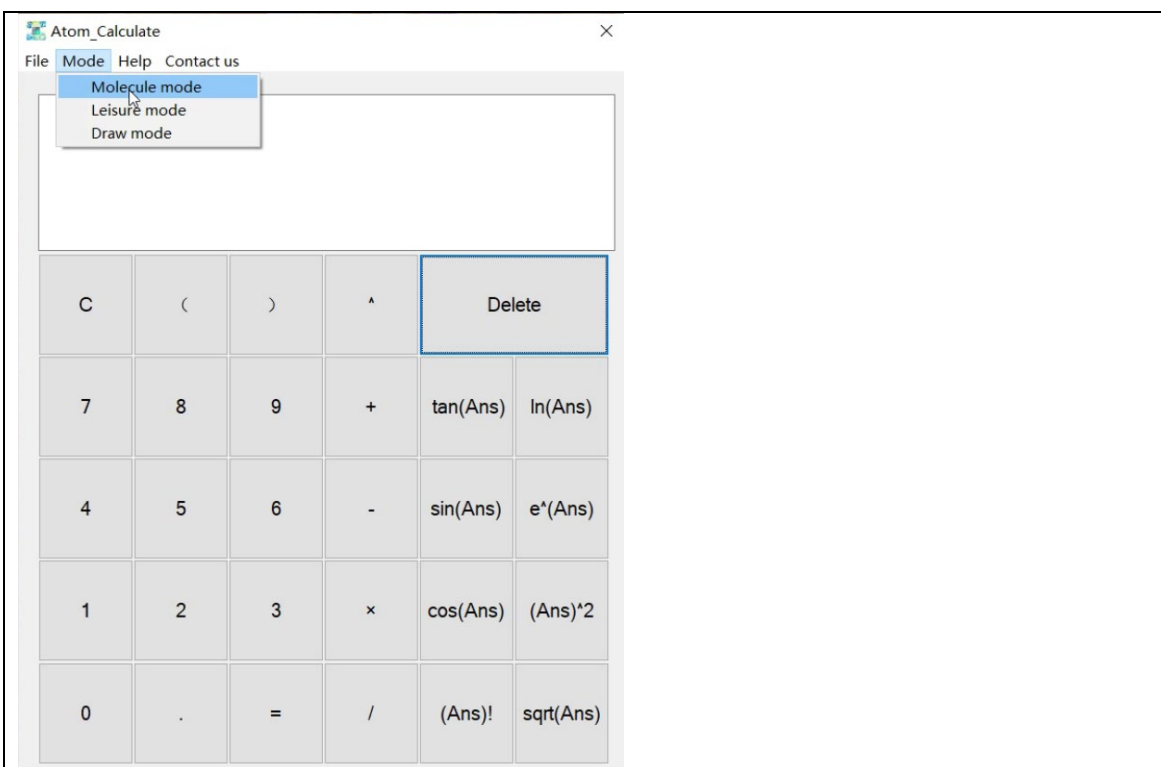
左键“File”，出现“设计报告 doc 文件”，点击 doc 可查看该计算机的设计报告。

如下图示：

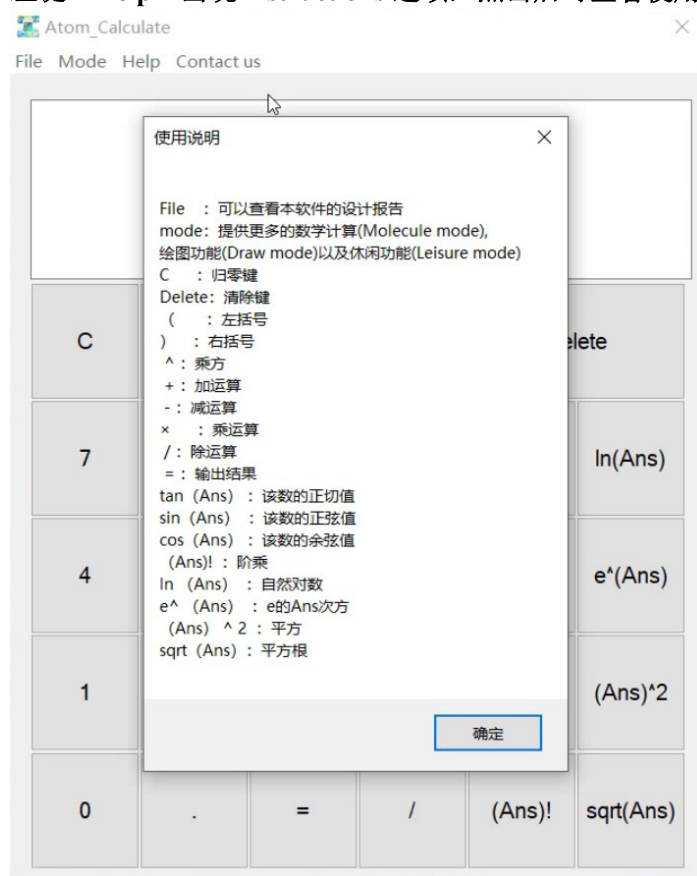


左键“mode”，出现“Molecule mode”（其他运算功能）、“Leisure mode”（休闲小游戏）及“Draw mode”（画画功能）选项。

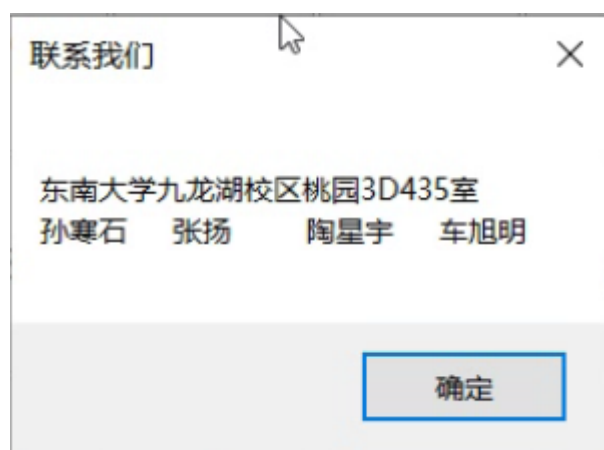
如下图所示：



左键“Help”出现 instructions 选项，点击后可查看使用说明，如下图示：



左键“Contact us”后弹出窗口如下图所示：



### 三、其他运算及功能的介绍

#### Draw mode:

绘画功能进入方式：Mode->Draw mode

进阶操作：点击 **Mode** 后，点击 **Molecule mode** 后（图 1-1），得到以下额外操作主界面（图 1-2）。

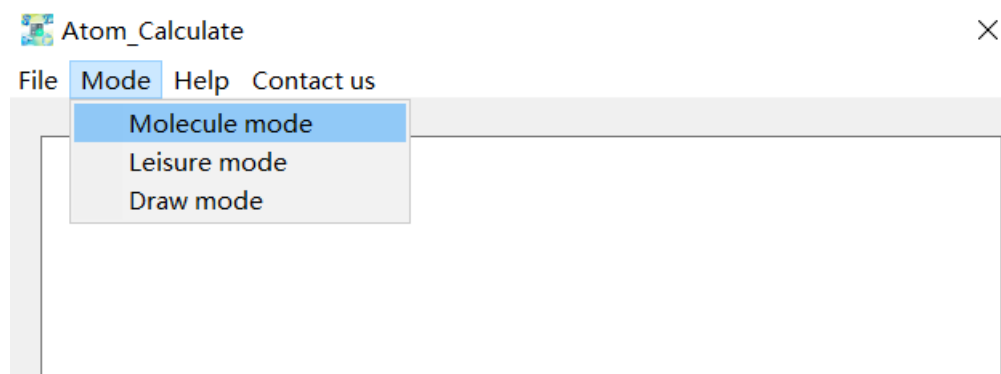


图 1-1

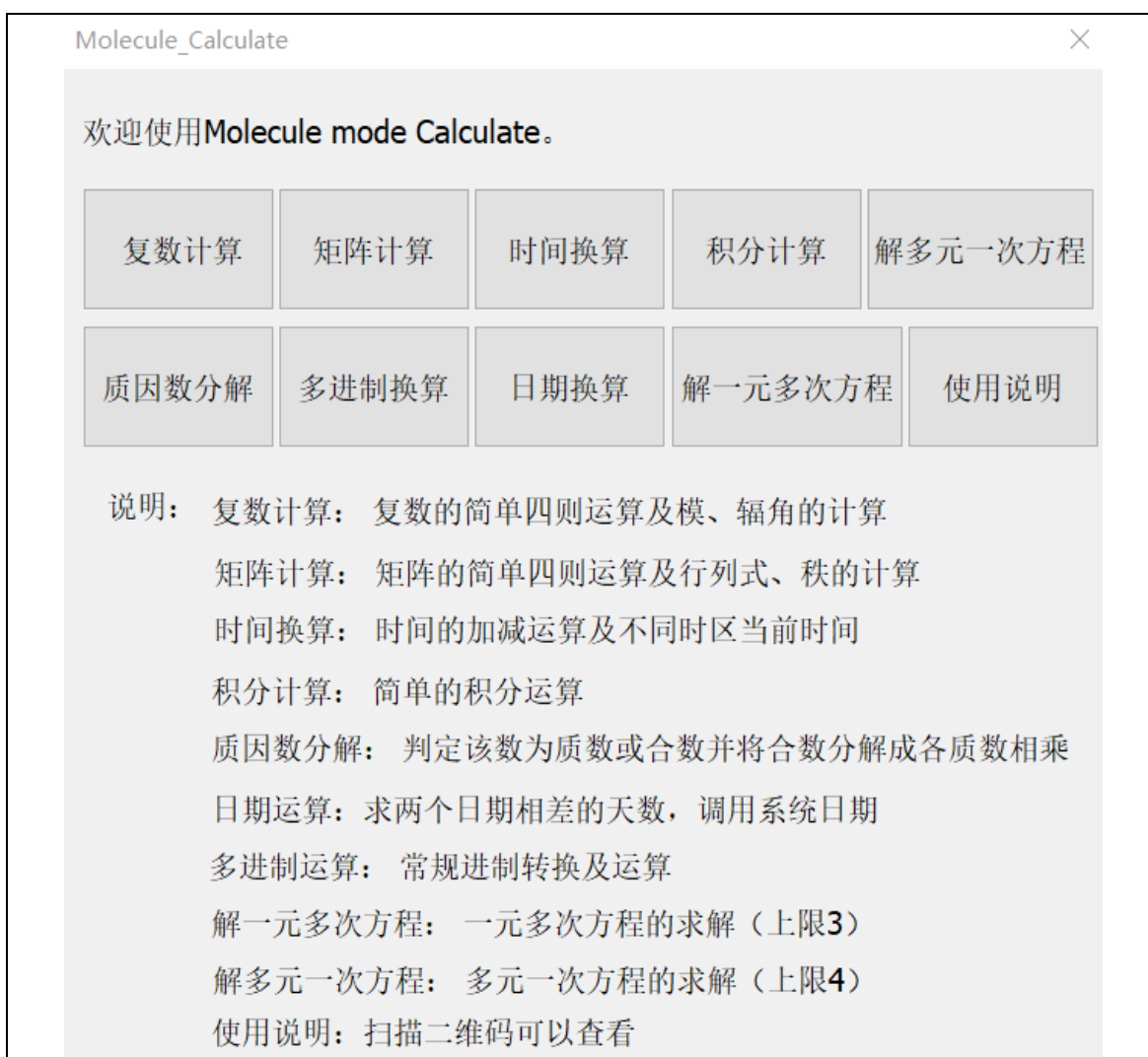


图 1-2

（一）单击复数计算，得到以下操作界面（图 2-1），分别填入操作数 1 的实数系数、虚数系数，操作 2 的实数系数、虚数系数。

Complex number calculation

请输入两个操作数：

操作数1  +  i

操作数2  +  i

加法	减法	乘法	除法
操作数1的模	操作数2的模	操作数1辐角	操作数2辐角

图 2-1

加法：两操作数相加结果                  减法：两操作数相减结果  
乘法：两操作数相乘结果                  除法：两操作数相除结果  
操作数 1 的模：对 1 进行模运算      操作数 2 的模：对 2 进行模运算  
操作数 1 的辐角：求出操作数 1 对应辐角  
操作数 2 的辐角：求出操作数 2 对应辐角

（二）单击矩阵计算，得到以下操作界面（图 3-1），分别填入二阶矩阵 1、2 的各项系数。

Matrix Calculation
×

请输入矩阵1和矩阵2的系数：

a11	<input type="text"/>	a12	<input type="text"/>	b11	<input type="text"/>	b12	<input type="text"/>
a21	<input type="text"/>	a22	<input type="text"/>	b21	<input type="text"/>	b22	<input type="text"/>

乘法

加法

减法

转置

矩阵1的行列式

矩阵2的行列式

矩阵1的秩

矩阵2的秩

图 3-1

乘法：两矩阵相乘结果                  加法：两矩阵相加结果  
减法：两矩阵相减结果                  转置：求出的矩阵的转置矩阵  
矩阵 1 的行列式：计算出矩阵 1 的行列式  
矩阵 2 的行列式：计算出矩阵 2 的行列式  
矩阵 1 的秩：计算出矩阵 1 的秩  
矩阵 2 的秩：计算出矩阵 2 的秩

（三）单击时间换算，得出以下操作界面（图 4-1），填入对应的时间和加减时间。

Time conversion
×

请输入时间：

<input type="text"/>	时	<input type="text"/>	分	<input type="text"/>	秒
----------------------	---	----------------------	---	----------------------	---

请输入需要加减的时间：

<input type="text"/>	时	<input type="text"/>	分	<input type="text"/>	秒
----------------------	---	----------------------	---	----------------------	---

换算输入的时间

计算加减后的时间

北京时间

东京时间

伦敦时间

华盛顿时间

图 4-1

换算输入的时间：对时间进行规范运算  
 计算加减后的时间：得出计算的时间结果  
 北京时间：当前时刻的北京当地时间  
 东京时间：当前时刻的东京当地时间  
 伦敦时间：当前时刻的伦敦当地时间  
 华盛顿时间：当前时刻的华盛顿当地时间

（四）单击积分计算，得出以下操作界面（图 5-1），选择对应函数，填入上下限，以得出结果。

integrate
×

上限：
下限：

☒  $y=x$

☐  $y=x^2$

☐  $y=e^x$

☐  $y=\ln x$

☐  $y=\sin x$

☐  $y=\cos x$

图 5-1

（五）单击解多元一次方程，得出以下操作界面（图 6-1），分别填入各项式系数及结果，判定是否有解，并求出解。

Solving linear equations
×

请输入方程的系数，如无此未知数，输入0即可。

a +  b +  c +  d =

a +  b +  c +  d =

a +  b +  c +  d =

a +  b +  c +  d =

图 6-1

（六）单击质因数分解，得出以下操作界面（图 7-1），输入数字以判定该数为质数或者合数，并将合数进行质因数分解。

Prime factor decomposition ×

请输入要分解的数字：

分解

图 7-1

示例： $12=2\times 2\times 3$

（七）单击多进制换算，得出以下操作界面（图 8-1），根据需求对数进行相对应的进制转换。

Base conversion ×

十进制	<input type="text"/>	十六进制	<input type="text"/>
二进制	<input type="text"/>	三十二进制	<input type="text"/>
八进制	<input type="text"/>	三十六进制	<input type="text"/>

换算

说明：请输入要换算的十进制数字之后点击“换算”键即可。

图 8-1

示例：十进制 25 转换成二进制为 11001



（八）单击日期换算，得出以下操作界面（图 9-1），输入两对应日期进行计算。

Date Conversion

×

请输入日期：

年月日

请输入第二个日期：

年月日

计算相差的天数

当前日期

图 9-1

计算相差的天数：根据两日期计算对应相差的天数

当前日期：当前的日期

（九）单击解一元多次方程，得出以下操作界面（图 10-1），输入各项式系数以求解。

Solving multiple equations of one variable

×

请输入方程的系数，若无某次项系数，则输入0。

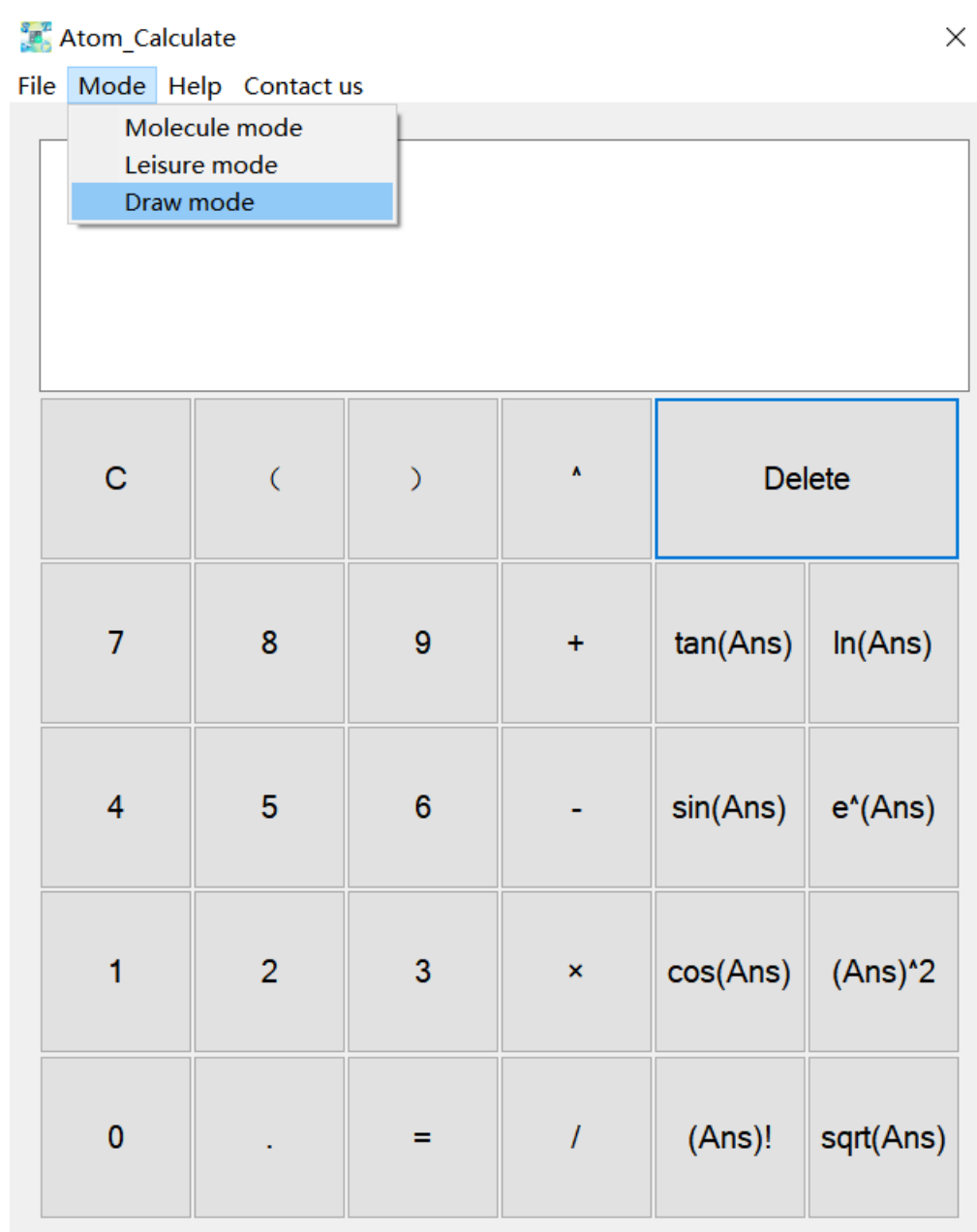
注意：三次项系数不能为0，否则是二次方程。

$x^3+$  $x^2+$  $x+$  $=0$

计算

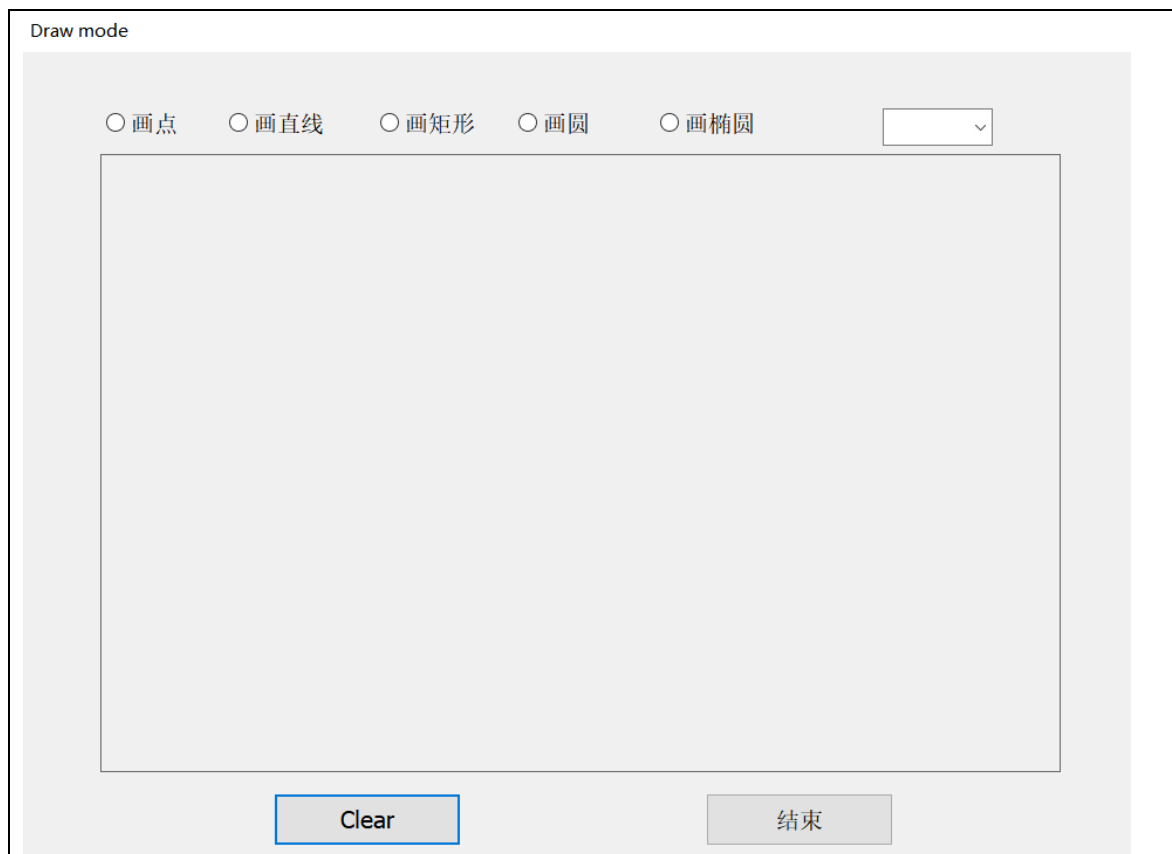
图 10-1

(十) 单击使用说明，可获得说明文件二维码，可下载并阅读使用说明文件。



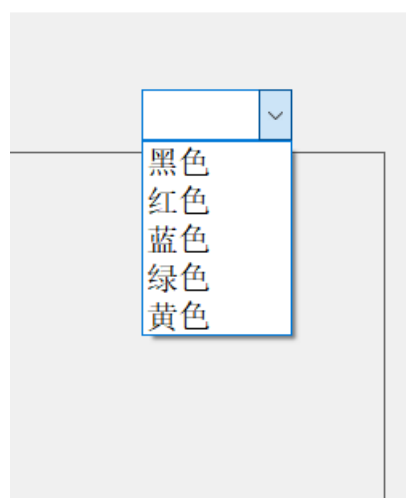
我们对绘画有点，直线，矩形，圆，椭圆等图形的实现。解放了绘画的限制性，用户可根据基本图形，塑造自己喜欢的图案。同时也增添了清除（Clear）功能和结束功能。

界面如下：



为防止绘画单调，我们又增添了颜色选择功能，可为用户提供黑色，红色，蓝色，绿色，黄色等五种颜色的选择。

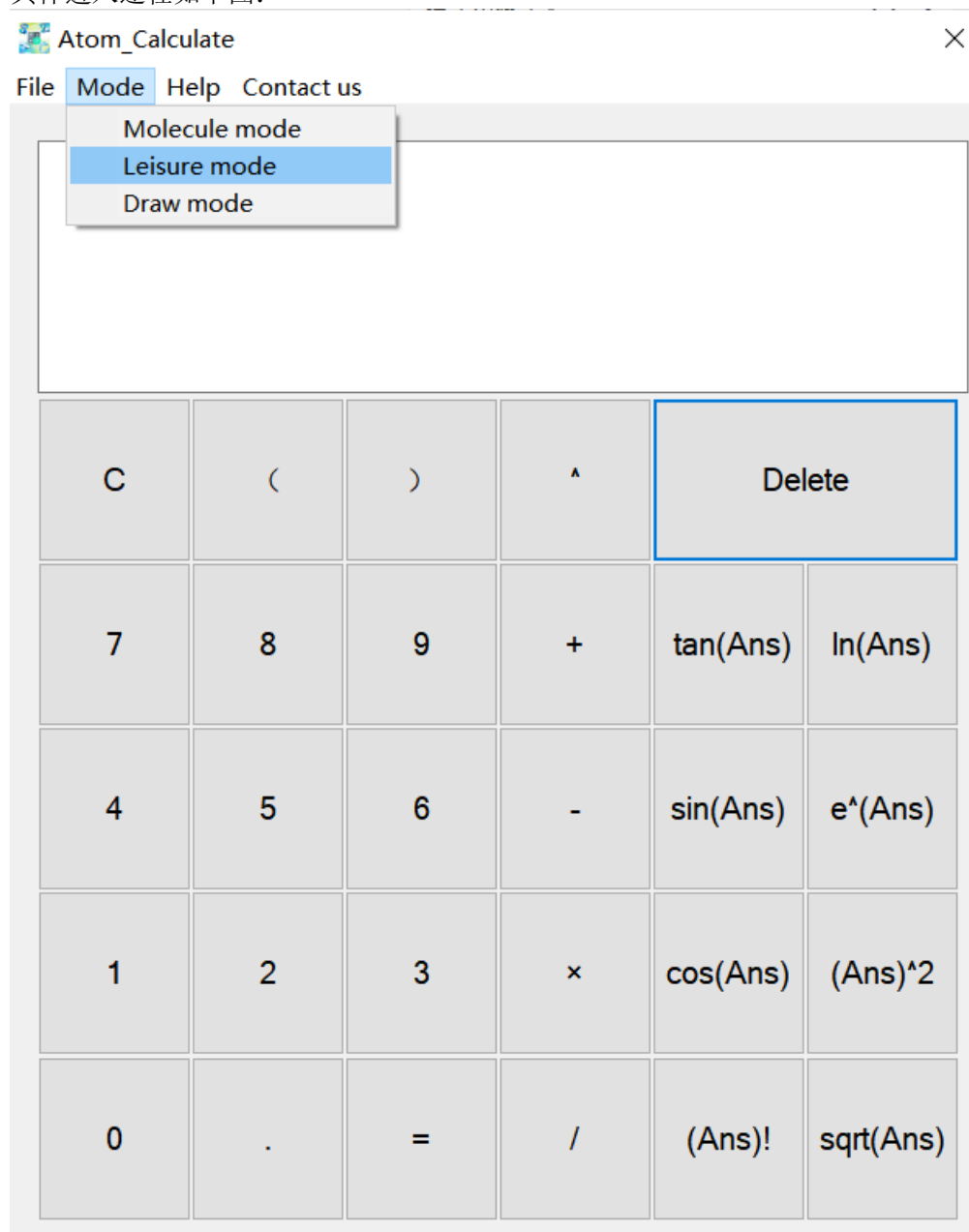
实现颜色选择的功能如下：



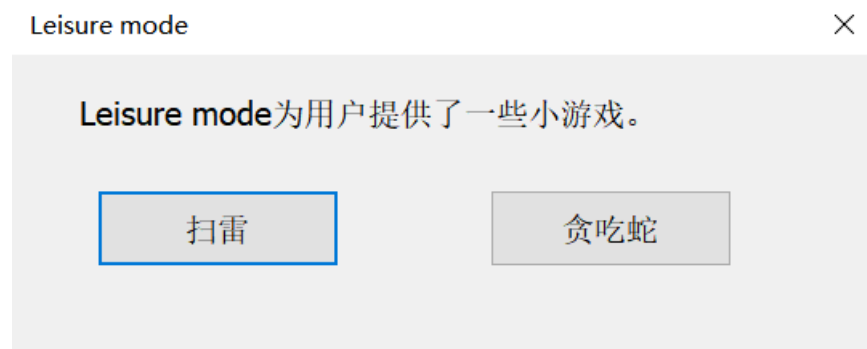
### 趣味性游戏 (Leisure mode)

本着为使用用户增加软件体验感，我们特意从网上借鉴了扫雷，贪吃蛇趣味性小游戏程序。并融入了我们的计算器 Mode->Leisure mode 程序中。

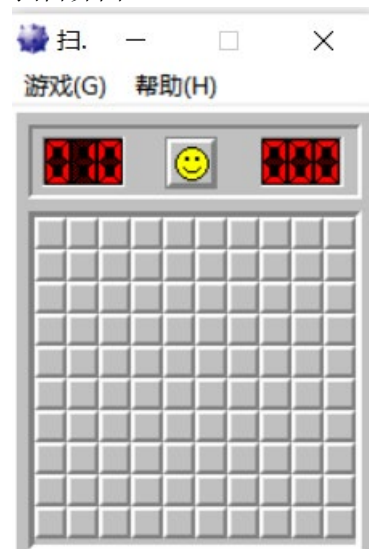
具体进入途径如下图：



趣味小游戏模式选择界面：



扫雷界面：



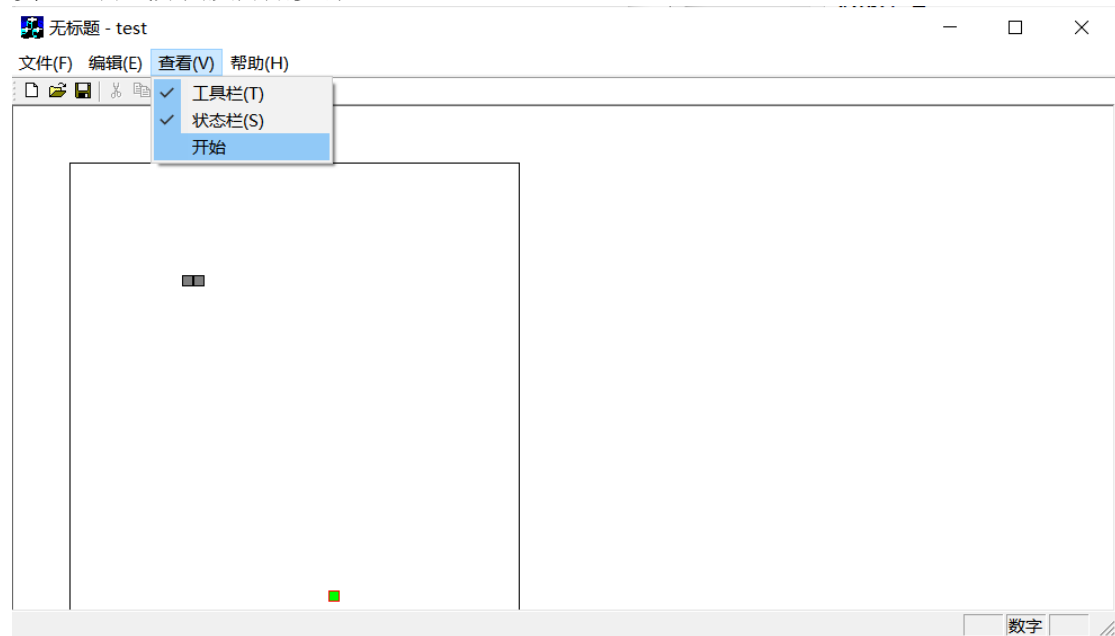
在一个初级，中级，高级，或自定义大小的方块矩阵中随机布置一定量的地雷(初级为 10 个，中级为 40 个，高级为 99 个)。由玩家逐个翻开方块，以找出所有地雷为最终游戏目标。如果玩家翻开的方块有地雷，则游戏结束。

游戏主区域由很多个方格组成。使用鼠标左键随机点击一个方格，方格即被打开并显示出方格中的数字；方格中数字则表示其周围的 8 个方格隐藏了几颗雷。

游戏菜单选择界面：



贪吃蛇游戏界面及开始步骤：



使用键盘的方向键进行贪吃蛇的控制，每吃到一个食物（小绿点），贪吃蛇的长度就会增加一个格子；在保证贪吃蛇不撞到墙壁和咬到自身的情况下，尽可能增加贪吃蛇的长度。

#### 维护及其异常处理：

矩阵计算异常处理：数字计算正常；如果输入字母，程序仍正常计算，但字母默认为数字 0 参与计算；

找不到设计报告：可能是由于计算器程序未与设计报告放在同一文件夹中；

**其他问题：**

如果出现用户不可修复的问题，可通过 QQ 联系我们。我们的 QQ 号为：

孙寒石：1264720735

张扬：1830714675

陶星宇：1102735220

车旭明：2388848373

### 三、成员分工情况（每个成员承担的任务、工作量所占比例(%)）

孙寒石：学习 MFC 程序知识，对程序的所有界面部分进行开发和完善，并编写 Atom mode 的全部代码。编写复数计算，矩阵计算，解一元多次方程，质因数分解，用户登录界面程序代码，并参与编写 Draw mode 程序，撰写课程设计报告内容简介，任务建议书，系统分析报告，系统设计报告，准备答辩材料，最后进行答辩。**31%**


张扬：学习 MFC 程序知识，编写时间换算、日期换算程序代码，参与编写 Draw mode 程序，参与编写 Leisure mode 程序。参与为了提高对用户友好性的界面美化工作，参与最后的视频制作和剪辑工作，参与撰写使用说明书。**23%**

陶星宇：学习 MFC 程序知识，PS 知识，编写解多元一次方程程序代码，参与编写 Draw mode 程序，参与编写 Leisure mode 程序。同时，为程序图标和图片利用 PS 程序进行制作。参与为了提高对用户友好性的界面美化工作，参与最后的视频制作和剪辑工作，参与撰写使用说明书。**23%**

车旭明：学习 MFC 程序知识，编写积分计算、多进制换算程序代码，参与编写 Draw mode 程序，参与编写 Leisure mode 程序。参与为了提高对用户友好性的界面美化工作，参与最后的视频制作和剪辑工作，参与撰写使用说明书。**23%**



个人报告	组员 1 学号：D2219118	姓名：张扬
------	------------------	-------

<p>个人完成分工详述：详述完成的分工具体内容，设计了什么功能函数？或者完成了什么功能模块的设计？流程图？完成情况如何？书写空间自行拓展，可贴图。</p> <p>本人负责分工具体内容：</p> <p>学习 MFC 程序知识，编写时间换算、日期换算程序代码，参与编写 Draw mode 程序，参与编写 Leisure mode 程序。参与为了提高对用户友好性的界面美化工作，参与录屏工作。</p> <p>日期换算函数的实现。</p> <div> <div> Date_Conversion.h </div> <div> （头文件中所示位置） </div> </div> <div> <div> Date_Conversion.cpp </div> <div> （cpp 文件中所示位置，代码主要区域） </div> </div> <p>对话框界面如下图：</p>  <p>在编辑框中分别输入日期一的年，月，日和日期二的年，月，日；左下角按钮为计算日期一和日期二相差的天数，计算结果为两者相差天数的绝对值，对两者大小无特殊要求。右下角为显示当前日期按钮，单击会弹出当前日期的对话框，方便用户读取当前日期。</p> <p>以下为整理后的 cpp 代码部分（代码较长）：</p> <pre> // Date_Conversion.cpp: 实现文件 //  #include "pch.h" #include "Calculate_atom.h" #include "Date_Conversion.h" #include "afxdialogex.h"  // Date_Conversion 对话框  IMPLEMENT_DYNAMIC(Date_Conversion, CDialogEx) </pre>
---

```

Date_Conversion::Date_Conversion(CWnd* pParent /*=nullptr*/)
    : CDialogEx(IDD_DIALOG9, pParent)
    , y1(_T(""))
    , m1(_T(""))
    , d1(_T(""))
    , y2(_T(""))
    , m2(_T(""))
    , d2(_T(""))
{
}

Date_Conversion::~Date_Conversion()
{
}

void Date_Conversion::DoDataExchange(CDataExchange* pDX)
{
    CDialogEx::DoDataExchange(pDX);
    DDX_Text(pDX, IDC_EDIT1, y1);
    DDX_Text(pDX, IDC_EDIT2, m1);
    DDX_Text(pDX, IDC_EDIT4, d1);
    DDX_Text(pDX, IDC_EDIT5, y2);
    DDX_Text(pDX, IDC_EDIT6, m2);
    DDX_Text(pDX, IDC_EDIT7, d2);
}

BEGIN_MESSAGE_MAP(Date_Conversion, CDialogEx)
    ON_BN_CLICKED(IDC_BUTTON2, &Date_Conversion::OnBnClickedButton2)
    ON_BN_CLICKED(IDC_BUTTON1, &Date_Conversion::OnBnClickedButton1)
    ON_BN_CLICKED(IDC_BUTTON4, &Date_Conversion::OnBnClickedButton4)
END_MESSAGE_MAP()

// Date_Conversion 消息处理程序

class date
{
public:

    date(int year = 1990, int month = 1, int day = 1)
        : _year(year)
        , _month(month)
        , _day(day)
    {

```

```

}

date(const date& d)
{
    _year = d._year;
    _month = d._month;
    _day = d._day;
}

bool isLeapyear(int year)
{
    if ((year % 4 == 0) && (year % 100 != 0))
    {
        return true;
    }
    else
    {
        if (year % 400 == 0)
            return true;
        else
            return false;
    }
}

int Getmonthday(int year, int month)
{
    int a[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
    if (isLeapyear(year) && month == 2)
        return a[month - 1] + 1;
    else

```

```

        return a[month - 1];

    }

    bool isinvalid(int year, int month, int day)
    {
        if (year >= 1900)
        {
            if (month > 0 && month <= 12)
            {
                if (day > 0 && (day <= Getmonthday(year, month)))

                    return true;

                else

                    return false;

            }

            else

                return false;

        }

        else

            return false;

    }

    bool operator ==(const date& d)
    {
        if (_year == d._year)
        {
            if (_month == d._month)
            {
                if (_day == d._day)

                    return true;

```

```

        else

            return false;

        }

        return false;

    }

    else

        return false;

}

bool operator !=(const date& d)

{

    return !(*this == d);

}

bool operator >(const date& d)

{

    if (_year > d._year)

    {

        return true;

    }

    else

    {

        if (_month > d._month)

            return true;

        else

        {

            if (_day > d._day)

                return true;

            else

                return false;

        }

    }

}

```

```

    }

    }

}

date operator +(int day)
{

    int nowday = _day + day;

    while (nowday >= Getmonthday(_year, _month))
    {

        nowday = nowday - Getmonthday(_year, _month);

        if (_month == 12)
        {

            _year++;

            _month = 1;

        }

        else
        {

            _month++;

        }

    }

    _day = nowday;

    return (*this);
}

```

```

}

date operator -(int day)

{
    int sumday = _day;
    while (sumday < day)
    {
        sumday = sumday + Getmonthday(_year, _month);

        if (_month == 1)
        {
            _month = 12;
            _year--;
        }
        else
        {
            _month--;
        }
    }

    _day = sumday - day;

    return (*this);
}

int operator -(date const& d)

{
    int count = 0;

    if (*this > d)
    {
        while (*this != d)
        {
            count++;
        }
    }
}

```

```

        *this = *this - 1;

    }

}

else

{

    while (*this != d)

    {

        count++;

        *this = *this + 1;

    }

}

return count;

}

private:

    int _year;

    int _month;

    int _day;

};

bool IsLeapYear(int year)
{
    return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
}

void Date_Conversion::OnBnClickedButton2()
{
    // TODO: 在此添加控件通知处理程序代码
    CString str;
    //获取系统时间
    CTime tm;

```



```

        tm = CTime::GetCurrentTime(); //获取系统日期
        str = tm.Format("现在北京时间是%Y 年%m 月%d 日");
        MessageBox(str, CString("当前日期"), MB_OK);
    }



void Date_Conversion::OnBnClickedButton1()
{
    // TODO: 在此添加控件通知处理程序代码
    UpdateData(TRUE);
    int count=0;
    int year = int(atof(y1));
    int month = int(atof(m1));
    int day = int(atof(d1));
    int year2 = int(atof(y2));
    int month2 = int(atof(m2));
    int day2 = int(atof(d2));
    date A(year, month, day);
    date B(year2, month2, day2);
    count = abs(A - B);
    CString m;
    m.Format(_T("%d"), count);
    MessageBox(CString("第一个日期与第二个日期的差值为: ") + m, CString("日期差值"));
    UpdateData(FALSE);
}

void Date_Conversion::OnBnClickedButton4()
{
    // TODO: 在此添加控件通知处理程序代码
}

```

代码部分主要包括：闰年判断函数，不同月份的天数储存函数，运算符重载函数，获取系统当前日期函数，计算日期时间差函数等等

#### 时间换算程序：

- ▷  Time\_Conversion.h （时间换算在头文件中所示位置）
- ▷  Time\_Conversion.cpp （时间换算程序在 cpp 文件中所示位置）

时间换算对话框如下：



对话框包括时间一的时，分，秒；和时间二的时，分，秒；也包括换算输入时间按钮（格式化）；计算加减后时间按钮；显示北京时间按钮；显示东京时间按钮；显示伦敦时间按钮；显示华盛顿时间按钮。方便使用者可以具体查看主流地区的时间。

具体代码如下：

```
// Time_Conversion.cpp: 实现文件
//

#include "pch.h"
#include "Calculate_atom.h"
#include "Time_Conversion.h"
#include "afxdialogex.h"

// Time_Conversion 对话框

IMPLEMENT_DYNAMIC(Time_Conversion, CDialogEx)

Time_Conversion::Time_Conversion(CWnd* pParent /*=nullptr*/)
: CDialogEx(IDD_DIALOG8, pParent)
, h1(_T(""))
, m1(_T(""))
, s1(_T(""))
, h2(_T(""))
, m2(_T(""))
, s2(_T(""))
{
}

Time_Conversion::~Time_Conversion()
{
}

void Time_Conversion::DoDataExchange(CDataExchange* pDX)
```

```

{
    CDialogEx::DoDataExchange(pDX);
    DDX_Text(pDX, IDC_EDIT2, h1);
    DDX_Text(pDX, IDC_EDIT4, m1);
    DDX_Text(pDX, IDC_EDIT1, s1);
    DDX_Text(pDX, IDC_EDIT6, h2);
    DDX_Text(pDX, IDC_EDIT7, m2);
    DDX_Text(pDX, IDC_EDIT8, s2);
}

BEGIN_MESSAGE_MAP(Time_Conversion, CDialogEx)
    ON_BN_CLICKED(IDC_BUTTON2, &Time_Conversion::OnBnClickedButton2)
    ON_BN_CLICKED(IDC_BUTTON1, &Time_Conversion::OnBnClickedButton1)
    ON_BN_CLICKED(IDC_BUTTON4, &Time_Conversion::OnBnClickedButton4)
    ON_BN_CLICKED(IDC_BUTTON9, &Time_Conversion::OnBnClickedButton9)
    ON_BN_CLICKED(IDC_BUTTON10, &Time_Conversion::OnBnClickedButton10)
    ON_BN_CLICKED(IDC_BUTTON11, &Time_Conversion::OnBnClickedButton11)
END_MESSAGE_MAP()

// Time_Conversion 消息处理程序
void Time_Conversion::basiccon(CString a, CString b, CString c)
{
    UpdateData(TRUE);
    double h = atof(a);
    double m = atof(b);
    double s = atof(c);
    double sum = 3600 * h + 60 * m + s;
    h = sum / 3600;
    h = int(h) % 24;
    sum = int(sum) % 3600;
    m = int(sum / 60);
    sum = int(sum) % 60;
    s = sum;
    h1.Format(_T("%d"), int(h));
    m1.Format(_T("%d"), int(m));
    s1.Format(_T("%d"), int(s));
    UpdateData(FALSE);
}

void Time_Conversion::OnBnClickedButton2()
{
    // TODO: 在此添加控件通知处理程序代码
    UpdateData(TRUE);
    basiccon(h1, m1, s1);
    double h = atof(h2);
    double m = atof(m2);
    double s = atof(s2);
    double sum = 3600 * h + 60 * m + s;
    h = sum / 3600;
    h = int(h) % 24;
    sum = int(sum) % 3600;
    m = int(sum / 60);
    sum = int(sum) % 60;
    s = sum;
    h2.Format(_T("%d"), int(h));

```

```

        m2.Format(_T("%d"), int(m));
        s2.Format(_T("%d"), int(s));
        UpdateData(FALSE);
    }

void Time_Conversion::OnBnClickedButton1()
{
    // TODO: 在此添加控件通知处理程序代码
    UpdateData(TRUE);
    basiccon(h1, m1, s1);
    CString a = h1 + h2;
    CString b = m1 + m2;
    CString c = s1 + s2;

    double h = atof(a);
    double m = atof(b);
    double s = atof(c);
    double sum = 3600 * h + 60 * m + s;
    h = sum / 3600;
    h = int(h) % 24;
    sum = int(sum) % 3600;
    m = int(sum / 60);
    sum = int(sum) % 60;
    s = sum;
    a.Format(_T("%d"), int(h));
    b.Format(_T("%d"), int(m));
    c.Format(_T("%d"), int(s));
    MessageBox(a+CString(":")+b+ CString(":")+c, _T("The answer"));
    UpdateData(FALSE);
}

void Time_Conversion::OnBnClickedButton4()
{
    // TODO: 在此添加控件通知处理程序代码
    CString str;
    //获取系统时间
    CTime tm;
    tm = CTime::GetCurrentTime();//获取系统日期
    str = tm.Format("现在北京时间是%Y年%m月%d日 %X");
    MessageBox(str, _T("The answer"));
}

void Time_Conversion::OnBnClickedButton9()
{
    // TODO: 在此添加控件通知处理程序代码
    CTime t = CTime::GetCurrentTime();
    int h = t.GetHour(); //获取当前为几时
    int mm = t.GetMinute(); //获取分钟
    int s = t.GetSecond(); //获取秒
    h = h + 1;
    h = h % 24;
    CString a, b, c;
    a.Format(_T("%d"), int(h));

```

```

        b.Format(_T("%d"), int(mm));
        c.Format(_T("%d"), int(s));
        MessageBox(CString("现在东京时间是")+a + CString(":") + b + CString(":") + c, _T("The
answer"));
    }

void Time_Conversion::OnBnClickedButton10()
{
    // TODO: 在此添加控件通知处理程序代码
    CTime t = CTime::GetCurrentTime();
    int h = t.GetHour(); //获取当前为几时
    int mm = t.GetMinute(); //获取分钟
    int s = t.GetSecond(); //获取秒
    h = h - 7;
    h = h % 24;
    CString a, b, c;
    a.Format(_T("%d"), int(h));
    b.Format(_T("%d"), int(mm));
    c.Format(_T("%d"), int(s));
    MessageBox(CString("现在伦敦时间是")+a + CString(":") + b + CString(":") + c, _T("The
answer"));
}

void Time_Conversion::OnBnClickedButton11()
{
    // TODO: 在此添加控件通知处理程序代码
    CTime t = CTime::GetCurrentTime();
    int h = t.GetHour(); //获取当前为几时
    int mm = t.GetMinute(); //获取分钟
    int s = t.GetSecond(); //获取秒
    h = h - 12;
    h = h % 24;
    CString a, b, c;
    a.Format(_T("%d"), int(h));
    b.Format(_T("%d"), int(mm));
    c.Format(_T("%d"), int(s));
    MessageBox(CString("现在华盛顿时间是")+a + CString(":") + b + CString(":") + c,
_T("The answer"));
}

```

#### 程序代码主要包括：

格式化时间函数；时间换算函数；显示换算结果函数；显示北京时间函数；显示东京时间函数；

显示伦敦时间函数；显示华盛顿时间函数。

Leisure mode 函数菜单负责圆的绘制工作。

#### 圆的核心部分代码如下：

#### 核心代码一：

```

if(m_model==3)
{
    if ((nFlags == MK_LBUTTON) && Is_LeftButton_Down)
    {
        ::SetCursor(cross);
    }
}

```

```

        CClientDC dc(this);
        CBrush *OldBrush;
        OldBrush=(CBrush*)dc.SelectStockObject(NULL_BRUSH); //创建一个不填充的画刷
        int r; //圆的半径

        dc.SetROP2(R2_NOT);
        dc.MoveTo(chosen_position);
        dc.LineTo(instant_position.x, instant_position.y);
        r =
sqrt((chosen_position.x-instant_position.x)*(chosen_position.x-instant_position.x)
+(chosen_position.y-instant_position.y)*(chosen_position.y-instant_position.y));

        dc.Ellipse(chosen_position.x-r, chosen_position.y-r, chosen_position.x+r, chosen_posit
ion.y+r);

        dc.MoveTo(chosen_position);
        dc.LineTo(point);
        r = sqrt((chosen_position.x-point.x)*(chosen_position.x-point.x)
+(chosen_position.y-point.y)*(chosen_position.y-point.y));

        dc.Ellipse(chosen_position.x-r, chosen_position.y-r, chosen_position.x+r, chosen_posit
ion.y+r);
        instant_position=point;

        dc.SelectObject(OldBrush);
    }
    else
        ::SetCursor(arrow);
}

```

#### 核心代码二:

```

if(m_model==3)
{
    if (Is_LeftButton_Down)
    {
        Is_LeftButton_Down = false;
        ::SetCursor(arrow);
        CClientDC dc(this);
        CPen* pOldPen=(CPen*)dc.SelectObject(&pen);

        CBrush *OldBrush;
        OldBrush=(CBrush*)dc.SelectStockObject(NULL_BRUSH); //创建一个不填充的画刷

        int r; //圆的半径

        r =
sqrt((chosen_position.x-instant_position.x)*(chosen_position.x-instant_position.x)
+(chosen_position.y-instant_position.y)*(chosen_position.y-instant_position.y));

        dc.Ellipse(chosen_position.x-r, chosen_position.y-r, chosen_position.x+r, chosen_posit
ion.y+r);
    }
}

```

```

        dc.SelectObject(OldBrush);
    }
}

```

#### 鼠标显示对话框代码:

```

ScreenToClient(rect_ctr); //获取 Picture 控件相对对话框客户区左上角的坐标
instant_position.x = point.x;
instant_position.y = point.y;
picture_ordinate_x = point.x - rect_ctr.left;
picture_ordinate_y = point.y - rect_ctr.top; //point 获取的是鼠标相对对话框客户区左上
角的坐标, 减去 rect_ctr.left 和 rect_ctr.top 后, 即为鼠标相对 Picture 控件左上角的坐标

```

```

//获取状态栏上的文字, 并将鼠标的坐标显示在状态栏上
char    szString[100]={0};
char    szText[512];

HWND hWnd = ::GetActiveWindow();
::GetWindowText(hWnd, szText, 200);
memset(szText+strlen(szText), ' ', 20);

//如果鼠标位于 Picture 控件之外, 在对话框状态栏上不显示其坐标
if( picture_ordinate_x>(rect_ctr.right-rect_ctr.left) ||
picture_ordinate_y>(rect_ctr.bottom-rect_ctr.top )
    || picture_ordinate_x<0 || picture_ordinate_y<0 );
else
    sprintf(szString, "(%4i, %4i)", picture_ordinate_x, picture_ordinate_y);

//将鼠标坐标显示到对话框状态栏上

strcpy(szText+20, szString);
::SetWindowText(hWnd, szText);

```

#### 颜色代码如下:

```

void CMFC_DRAWDlg::OnSelchangeCombo1 ()
{
    // TODO: Add your control notification handler code here

    int pos = m_Combo.GetCurSel();
    m_Combo.GetLBText(pos, collor_str);

    pen.CreatePen(0, 1, RGB(0, 0, 0));

    if(collor_str=="黑色")
    {
        pen.CreatePen(0, 1, RGB(0, 0, 0));
    }

    if(collor_str=="红色")
    {
        pen.CreatePen(0, 1, RGB(255, 0, 0));
    }
}

```

```

if(collor_str=="蓝色")
{
    pen.CreatePen(0,1,RGB(0,0,255));
}

if(collor_str=="绿色")
{
    pen.CreatePen(0,1,RGB(0,255,0));
}

if(collor_str=="黄色")
{
    pen.CreatePen(0,1,RGB(255,255,0));
}

}

```

#### 完成情况:

本人负责部分已完成计划书上所有相关内容，功能也已经实现，使用过程中出现的 bug 也已经修复。

#### 流程图:

##### 日期换算:

输入日期一的年,月,日-->数入日期二的年,月,日-->计算日期一和日期二的时间差-->显示当前日期（此部分可在页面随时实现）

##### 时间换算:

输入时间一的时,分,秒-->数入时间二的时,分,秒-->计算时间一和时间二的时间差-->显示北京,东京,伦敦,华盛顿当前时间（此部分可在页面随时实现）

##### 画图函数:

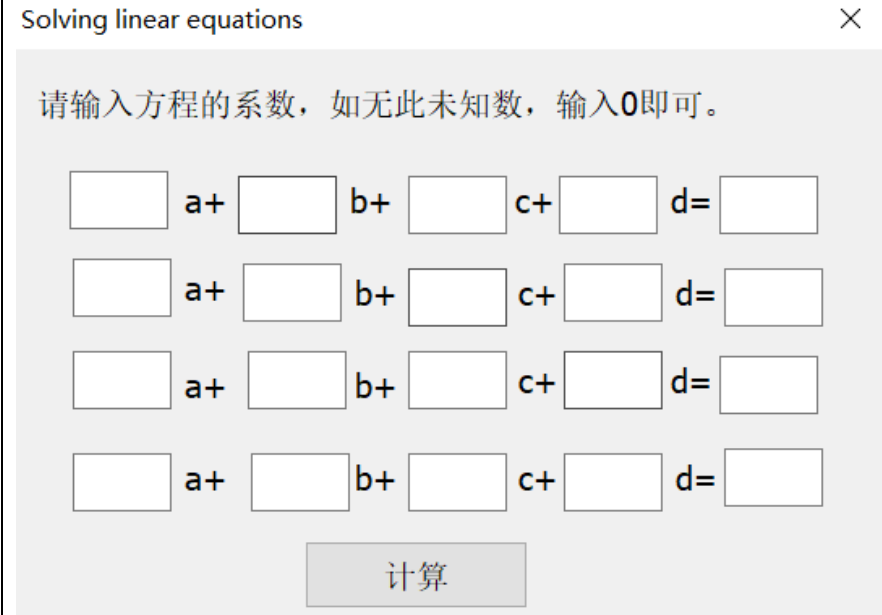
选择所需图形（点,直线,圆,矩形,椭圆）-->选择颜色（黑色,红色,黄色,蓝色,绿色）-->选取圆心,再选取半径。



个人报告	组员 2 学号：D2219113	姓名：陶星宇
------	------------------	--------

个人完成分工详述：详述完成的分工具体内容，设计了什么功能函数？或者完成了什么功能模块的设计？流程图？完成情况如何？书写空间自行拓展，可贴图。

自学 MFC 有关知识，PS 知识，完成了多元一次方程，椭圆的代码。  
以下为外观视图：



可完成四元及以下方程的求解。

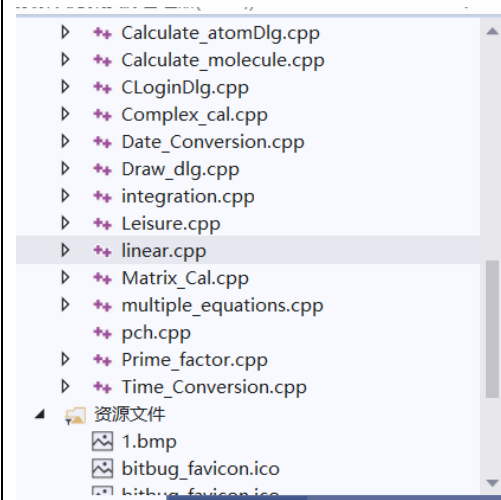
算法说明：

判断系数矩阵的秩，利用增广矩阵的算法化简，最终得到解。

控制消息流：

输入需要求解的信息后，可以用鼠标利用命令控件进行操作。

以下为代码及视图：



```
#include "pch.h"
#include "Calculate_atom.h"
#include "linear.h"
```

```

#include "afxdialogex.h"

// linear 对话框

IMPLEMENT_DYNAMIC(linear, CDialogEx)

linear::linear(CWnd* pParent /*=nullptr*/)
    : CDialogEx(IDD_DIALOG3, pParent)
    , coefficient_11(_T(""))
    , coefficient_12(_T(""))
    , coefficient_13(_T(""))
    , coefficient_14(_T(""))
    , coefficient_15(_T(""))
    , coefficient_21(_T(""))
    , coefficient_22(_T(""))
    , coefficient_23(_T(""))
    , coefficient_24(_T(""))
    , coefficient_25(_T(""))
    , coefficient_31(_T(""))
    , coefficient_32(_T(""))
    , coefficient_33(_T(""))
    , coefficient_34(_T(""))
    , coefficient_35(_T(""))
    , coefficient_41(_T(""))
    , coefficient_42(_T(""))
    , coefficient_43(_T(""))
    , coefficient_44(_T(""))
    , coefficient_45(_T(""))
{
}

linear::~linear()
{
}

void linear::DoDataExchange(CDataExchange* pDX)
{
    CDialogEx::DoDataExchange(pDX);
    DDX_Text(pDX, IDC_EDIT1, coefficient_11);
    DDX_Text(pDX, IDC_EDIT6, coefficient_12);
    DDX_Text(pDX, IDC_EDIT10, coefficient_13);
    DDX_Text(pDX, IDC_EDIT14, coefficient_14);
    DDX_Text(pDX, IDC_EDIT18, coefficient_15);
    DDX_Text(pDX, IDC_EDIT4, coefficient_21);
    DDX_Text(pDX, IDC_EDIT7, coefficient_22);
    DDX_Text(pDX, IDC_EDIT11, coefficient_23);
    DDX_Text(pDX, IDC_EDIT15, coefficient_24);
    DDX_Text(pDX, IDC_EDIT19, coefficient_25);
    DDX_Text(pDX, IDC_EDIT2, coefficient_31);
}

```

```

DDX_Text(pDX, IDC_EDIT8, coefficient_32);
DDX_Text(pDX, IDC_EDIT12, coefficient_33);
DDX_Text(pDX, IDC_EDIT16, coefficient_34);
DDX_Text(pDX, IDC_EDIT20, coefficient_35);
DDX_Text(pDX, IDC_EDIT5, coefficient_41);
DDX_Text(pDX, IDC_EDIT9, coefficient_42);
DDX_Text(pDX, IDC_EDIT13, coefficient_43);
DDX_Text(pDX, IDC_EDIT17, coefficient_44);
DDX_Text(pDX, IDC_EDIT21, coefficient_45);
}

BEGIN_MESSAGE_MAP(linear, CDialogEx)
    ON_BN_CLICKED(IDC_BUTTON2, &linear::OnBnClickedButton2)
END_MESSAGE_MAP() // linear 消息处理程序
void linear::OnBnClickedButton2()
{
    UpdatedData(TRUE);
    int n, m;
    double a11, a12, a13, a14, a15, a21, a22, a23, a24, a25, a31, a32, a33, a34,
a35, a41, a42, a43, a44, a45;
    double s[4];
    a11 = atof(coefficient_11);
    a12 = atof(coefficient_12);
    a13 = atof(coefficient_13);
    a14 = atof(coefficient_14);
    a15 = atof(coefficient_15);
    a21 = atof(coefficient_21);
    a22 = atof(coefficient_22);
    a23 = atof(coefficient_23);
    a24 = atof(coefficient_24);
    a25 = atof(coefficient_25);
    a31 = atof(coefficient_31);
    a32 = atof(coefficient_32);
    a33 = atof(coefficient_33);
    a34 = atof(coefficient_34);
    a35 = atof(coefficient_35);
    a41 = atof(coefficient_41);
    a42 = atof(coefficient_42);
    a43 = atof(coefficient_43);
    a44 = atof(coefficient_44);
    a45 = atof(coefficient_45);
    double det = a11*a22 * a33 * a44 - a11 * a22 * a34 * a43 - a11 * a23 * a32 *
a44 + a11 * a23 * a34 * a42
        + a11 * a24 * a32 * a43 - a11 * a24 * a33 * a42 - a12 * a21 * a33 * a44
+ a12 * a21 * a34 * a43
        + a12 * a23 * a31 * a44 - a12 * a23 * a34 * a41 - a12 * a24 * a31 * a43
+ a12 * a24 * a33 * a41
        + a13 * a21 * a32 * a44 - a13 * a21 * a34 * a42 - a13 * a22 * a31 * a44
+ a13 * a22 * a34 * a41
        + a13 * a24 * a31 * a42 - a13 * a24 * a32 * a41 - a14 * a21 * a32 * a43

```

```

+ a14 * a21 * a33 * a42
+ a14 * a22 * a31 * a43 - a14 * a22 * a33 * a41 - a14 * a23 * a31 * a42
+ a14 * a23 * a32 * a41;
double a[4][5] = {
    {a11, a12, a13, a14, a15},
    {a21, a22, a23, a24, a25},
    {a31, a32, a33, a34, a35},
    {a41, a42, a43, a44, a45},
}; //第四列是增广矩阵
int i, j;
n = 4;
for (j = 0; j < n; j++)
{
    double max = 0;
    double imax = 0;
    for (i = j; i < n; i++)
    {
        if (imax < fabs(a[i][j])) {
            imax = fabs(a[i][j]);
            max = a[i][j]; //得到各行中所在列最大元素
            m = i;
        }
    }
    if (fabs(a[j][j]) != max)
    {
        double b = 0;
        for (int k = j; k < n + 1; k++) {
            b = a[j][k];
            a[j][k] = a[m][k];
            a[m][k] = b;
        }
    }
    for (int r = j; r < n + 1; r++)
    {
        a[j][r] = a[j][r] / max; //让该行的所在列除以所在列的第一个元素，目的是让首元素为1
    }
    for (i = j + 1; i < n; i++)
    {
        double c = a[i][j];
        if (c == 0) continue;
        for (int s = j; s < n + 1; s++) {
            double tempdata = a[i][s];
            a[i][s] = a[i][s] - a[j][s] * c; //前后行数相减，使下一行或者上一行的首元素为0
        }
    }
}
}

```

```

for (i = n - 2; i >= 0; i--)
{
    for (j = i + 1; j < n; j++)
    {
        double tempData = a[i][j];
        double data1 = a[i][n];
        double data2 = a[j][n];
        a[i][n] = a[i][n] - a[j][n] * a[i][j];
    }
}

for (int k = 0; k < n; k++) {
    s[k] = a[k][n];
}

CString S1, S2, S3, S4, S5;
S1.Format(_T("%.4f"), s[0]);
S2.Format(_T("%.4f"), s[1]);
S3.Format(_T("%.4f"), s[2]);
S4.Format(_T("%.4f"), s[3]);
S1 = CString("解为:\n") + CString("a=") + S1 + CString("\n") + CString("b=")
+
    S2 + CString("\n") + CString("c=") + S3 + CString("\n") + CString("d=")
+ S4;
if(det)MessageBox(S1, _T("The answer"));
if(det==0)MessageBox(_T("不满秩"), _T("Wrong"));
UpdateData(FALSE);
}

```

参与了画图椭圆的部分编程，以下为部分代码：

```

if(m_model==4)
{
    if(!Under_painting)
    {
        if ((nFlags == MK_LBUTTON) && Is_LeftButton_Down)
        {
            ::SetCursor(cross);
            CClientDC dc(this);
            dc.SetROP2(R2_NOT);

            dc.MoveTo(chosen_position);
            dc.LineTo(chosen_position.x+1, instant_position.y);

            dc.MoveTo(chosen_position);
            dc.LineTo(chosen_position.x+1, point.y);

            instant_position=point;
        }
    }
    else if(Under_painting)
    {

```

```

        ::SetCursor(cross);
        CClientDC dc(this);
        b = point.x - chosen_position.x;
        int instant_b = instant_position.x - chosen_position.x;

        dc.SetROP2(R2_NOT);
        CBrush *OldBrush;
        OldBrush=(CBrush*)dc.SelectStockObject(NULL_BRUSH);

        dc.Ellipse(chosen_position.x-instant_b,chosen_position.y-a,chosen_position
.x+instant_b,chosen_position.y+a);

        dc.Ellipse(chosen_position.x-b,chosen_position.y-a,chosen_position.x+b,cho
sen_position.y+a);

        dc.SelectObject(OldBrush);

        instant_position=point;
    }
    else
        ::SetCursor(arrow);
}

CDialogEx::OnMouseMove(nFlags, point);
}

```

另外进行了简单的 PS 图标设计：



程序运行时可能出现数据异常、乱码等错误，已在后续调试中修复。

个人报告	组员 3 学号: D2219114	姓名: 车旭明
------	-------------------	---------

个人完成分工详述: 详述完成的分工具体内容, 设计了什么功能函数? 或者完成了什么功能模块的设计? 流程图? 完成情况如何? 书写空间自行拓展, 可贴图。

本人主要负责积分运算与进制换算程序的编写, 同时还参与 Draw mode 与 Leisure mode 功能的部分程序编写。

积分运算程序主要实现了几种简单函数的积分运算, 设计了积分上下限输入和函数选择窗口, 具体如图示:

进行积分运算时选择要被积分的函数, 同时输入积分上下限, 点击计算星星积分运算。

功能实现函数如下:

```
void integration::OnBnClickedButton1()
{
    UpdateData(TRUE);
    CString result;
    if (((CButton*)GetDlgItem(IDC_RADIO1))->GetCheck())
    {
        double max = atof(integrate_input); //上限
        double min = atof(integrate_input_min); //下限
        double res = ((max * max ) - (min * min )) / 2;
        result.Format(_T("%.4f"), res);
        MessageBox(_T(result), _T("The answer"));
    }
    if (((CButton*)GetDlgItem(IDC_RADIO2))->GetCheck())
    {
        double max = atof(integrate_input); //上限
        double min = atof(integrate_input_min); //下限
        double res = ((max * max * max) - (min * min * min)) / 3;
        result.Format(_T("%.4f"), res);
        MessageBox(_T(result), _T("The answer"));
    }
    if (((CButton*)GetDlgItem(IDC_RADIO3))->GetCheck())
    {
        double max = atof(integrate_input); //上限
```



```

        double min = atof(integrate_input_min); //下限
        double res = exp(max) - exp(min);
        result.Format(_T("%.4f"), res);
        MessageBox(_T(result), _T("The answer"));
    }
    if (((CButton*)GetDlgItem(IDC_RADIO4))->GetCheck())
    {
        double max = atof(integrate_input); //上限
        double min = atof(integrate_input_min); //下限
        double res = max*(log(max)-1)- min * (log(min) - 1);
        result.Format(_T("%.4f"), res);
        MessageBox(_T(result), _T("The answer"));
    }
    if (((CButton*)GetDlgItem(IDC_RADIO5))->GetCheck())
    {
        double max = atof(integrate_input); //上限
        double min = atof(integrate_input_min); //下限
        double res = sin(max) - sin(min);
        result.Format(_T("%.4f"), res);
        MessageBox(_T(result), _T("The answer"));
    }
    if (((CButton*)GetDlgItem(IDC_RADIO6))->GetCheck())
    {
        double max = atof(integrate_input); //上限
        double min = atof(integrate_input_min); //下限
        double res = cos(max) - cos(min);
        result.Format(_T("%.4f"), res);
        MessageBox(_T(result), _T("The answer"));
    }
    UpdateData(FALSE);
}

```

多进制运算部分实现了多种常用进制之间的转换，设计的换算窗口具体如下：

Base conversion ×

十进制 <input style="width: 100px;" type="text"/>	十六进制 <input style="width: 100px;" type="text"/>
二进制 <input style="width: 100px;" type="text"/>	三十二进制 <input style="width: 100px;" type="text"/>
八进制 <input style="width: 100px;" type="text"/>	三十六进制 <input style="width: 100px;" type="text"/>

说明：请输入要换算的十进制数字之后点击“换算”键即可。

进行计算时，输入要换算的十进制数字才有效，同时进行换算要点击按钮，按 enter 键会退出。

功能实现函数如下：

```

Base_conversion::Base_conversion(CWnd* pParent /*=nullptr*/)
: CDialogEx(IDD_DIALOG5, pParent)
, base10(_T(""))

```

```

        , base16(_T(""))
        , base2(_T(""))
        , base32(_T(""))
        , base8(_T(""))
        , base36(_T(""))
    {
    }

Base_conversion::~Base_conversion()
{
}

void Base_conversion::DoDataExchange(CDataExchange* pDX)
{
    CDialogEx::DoDataExchange(pDX);
    DDX_Text(pDX, IDC_EDIT1, base10);
    DDX_Text(pDX, IDC_EDIT5, base16);
    DDX_Text(pDX, IDC_EDIT2, base2);
    DDX_Text(pDX, IDC_EDIT6, base32);
    DDX_Text(pDX, IDC_EDIT4, base8);
    DDX_Text(pDX, IDC_EDIT7, base36);
}

BEGIN_MESSAGE_MAP(Base_conversion, CDialogEx)
    //ON_COMMAND(ID_MODE_LEISUREMODE, &Base_conversion::OnModeLeisuremode)
    ON_BN_CLICKED(IDC_BUTTON1, &Base_conversion::OnBnClickedButton1)
END_MESSAGE_MAP()

// Base_conversion 消息处理程序

void Base_conversion::OnBnClickedButton1()
{
    // TODO: 在此添加控件通知处理程序代码
    UpdateData(TRUE);
    int k;
    char m[1000];
    if (base10)
    {
        k = int(atoi(base10));
        _itoa_s(k, m, 2);
        base2 = CString(m);
        _itoa_s(k, m, 16);
        base16 = CString(m);
        _itoa_s(k, m, 32);
        base32 = CString(m);
        _itoa_s(k, m, 8);
        base8 = CString(m);
        _itoa_s(k, m, 36);
        base36 = CString(m);
    }
}

```

```

    }
    UpdateData(FALSE);
}

此外还实现了 Draw mode 的矩形部分的功能，具体程序如下：
CMFC_DRAWDlg::CMFC_DRAWDlg(CWnd* pParent /*=NULL*/)
: CDialogEx(CMFC_DRAWDlg::IDD, pParent)
, m_model(EOF)
, Is_LeftButton_Down(false)
, Under_painting(false)
, collor_str("黑色")
{
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
    // Load two cursors
    cross = AfxGetApp()->LoadStandardCursor(IDC_CROSS);
    arrow = AfxGetApp()->LoadStandardCursor(IDC_ARROW);
}

void CMFC_DRAWDlg::DoDataExchange(CDataExchange* pDX)
{
    CDialogEx::DoDataExchange(pDX);
    DDX_Control(pDX, IDC_COMBO1, m_Combo);
}

BEGIN_MESSAGE_MAP(CMFC_DRAWDlg, CDialogEx)
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_WM_MOUSEMOVE()
    ON_COMMAND(IDC_RADIO1, &CMFC_DRAWDlg::OnRadio1)
    ON_COMMAND(IDC_RADIO2, &CMFC_DRAWDlg::OnRadio2)
    // ON_WM_NCLBUTTONDOWN()
    ON_WM_LBUTTONDOWN()
    ON_WM_LBUTTONUP()
    ON_COMMAND(IDC_RADIO3, &CMFC_DRAWDlg::OnRadio3)
    ON_COMMAND(IDC_RADIO4, &CMFC_DRAWDlg::OnRadio4)
    ON_COMMAND(IDC_RADIO5, &CMFC_DRAWDlg::OnRadio5)
    //ON_WM_ERASEBKGD()
    //ON_WM_ERASEBKGD()
    //ON_WM_ERASEBKGD()
    //ON_WM_ERASEBKGD()
    ON_BN_CLICKED(IDC_BUTTON1, &CMFC_DRAWDlg::OnClickedButton1)
    ON_CBN_SELCHANGE(IDC_COMBO1, &CMFC_DRAWDlg::OnSelchangeCombo1)
    ON_BN_CLICKED(IDC_RADIO1, &CMFC_DRAWDlg::OnBnClickedRadio1)
    ON_BN_CLICKED(IDC_RADIO3, &CMFC_DRAWDlg::OnBnClickedRadio3)
    ON_BN_CLICKED(IDC_BUTTON2, &CMFC_DRAWDlg::OnBnClickedButton2)
END_MESSAGE_MAP()

// CMFC_DRAWDlg 消息处理程序

BOOL CMFC_DRAWDlg::OnInitDialog()
{
    CDialogEx::OnInitDialog();

    // 设置此对话框的图标。当应用程序主窗口不是对话框时，框架将自动
    // 执行此操作
    SetIcon(m_hIcon, TRUE); // 设置大图标

```

```

SetIcon(m_hIcon, FALSE);      // 设置小图标

// TODO: 在此添加额外的初始化代码
//this->UpdateWindow();

return TRUE; // 除非将焦点设置到控件，否则返回 TRUE
}

// 如果向对话框添加最小化按钮，则需要下面的代码
// 来绘制该图标。对于使用文档/视图模型的 MFC 应用程序，
// 这将由框架自动完成。

void CMFC_DRAWDlg::OnPaint()
{
    if (IsIconic())
    {
        CPaintDC dc(this); // 用于绘制的设备上下文

        SendMessage(WM_ICONERASEBKGND, reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

        // 使图标在工作区矩形中居中
        int cxIcon = GetSystemMetrics(SM_CXICON);
        int cyIcon = GetSystemMetrics(SM_CYICON);
        CRect rect;
        GetClientRect(&rect);
        int x = (rect.Width() - cxIcon + 1) / 2;
        int y = (rect.Height() - cyIcon + 1) / 2;

        // 绘制图标
        dc.DrawIcon(x, y, m_hIcon);

        //dc.MoveTo(chosen_position[0],chosen_position[1]);
        //dc.LineTo(instant_position[0],instant_position[1]);
        //Invalidate();
    }
    else
    {
        CDialogEx::OnPaint();
    }
}

```

**Leisure mode** 主要是为了增添计算机的趣味性与可玩性，而引进外部程序实现了小游戏功能。