C++ 语 言 程 序 设 计

实
验
报
告

实 验 四

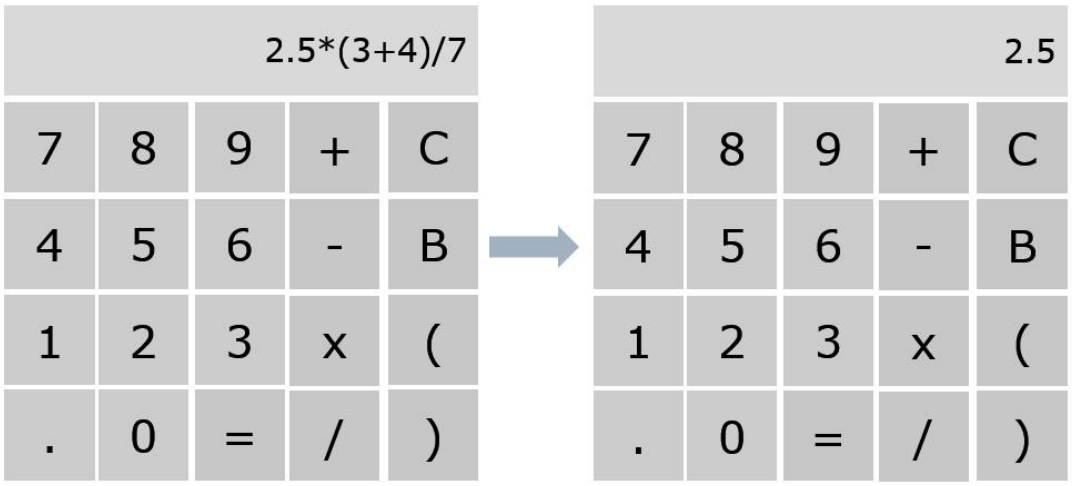
姓名：方尧

学号：190410102

班级：19 自动化 1 班

一 实验项目

1. 封装实验二的表达式计算功能到一个类中。
2. 编写基于 win32 编程的 GUI 界面，实现计算式的交互式输入与结果显示：



3. 实现功能

[1]. 点击数字、运算符号或小数点，上面显示框中即时显示
[2]. 点击 = ，显示框输出运算结果
[3]. 点击 C ，清空显示框
[4]. 点击 B ，实现输入显示的内容退格
[5]. 鼠标移至数字或符号的按键上时，按键变色，移开后恢复
[6]. 鼠标点击时，左键按下，按键颜色变深，左键弹起，恢复。
[7]. 鼠标点击时，不响应右键（按下鼠标右键后，程序界面无反应）
[8]. 支持键盘输入运算表达式
[9]. 键盘输入时，非界面上的数字或符号，则不响应
[10]. 本程序不考虑错误处理，默认输入的表达式为正确的

二 实验原理

1、 简述程序交互界面功能的实现方法

- 接收消息，通过接口传入参数，处理消息
- 按键

实现 0-9、乘号(用 x 代替)、等号、括号(用大括号[ ]代替)

2、 给出实验测试结果

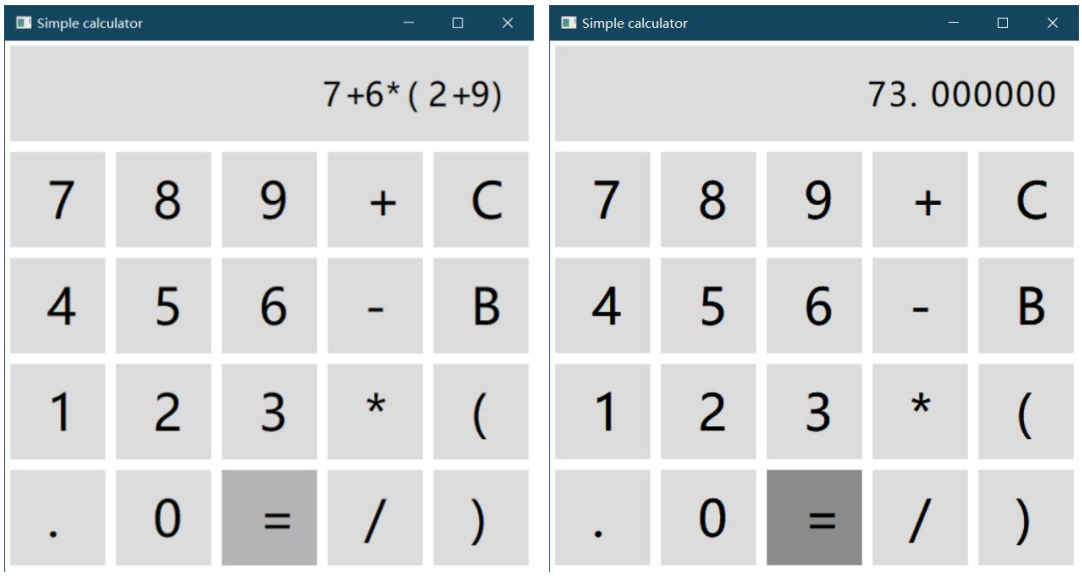


figure1 鼠标输入：运算式及结果、按键显示

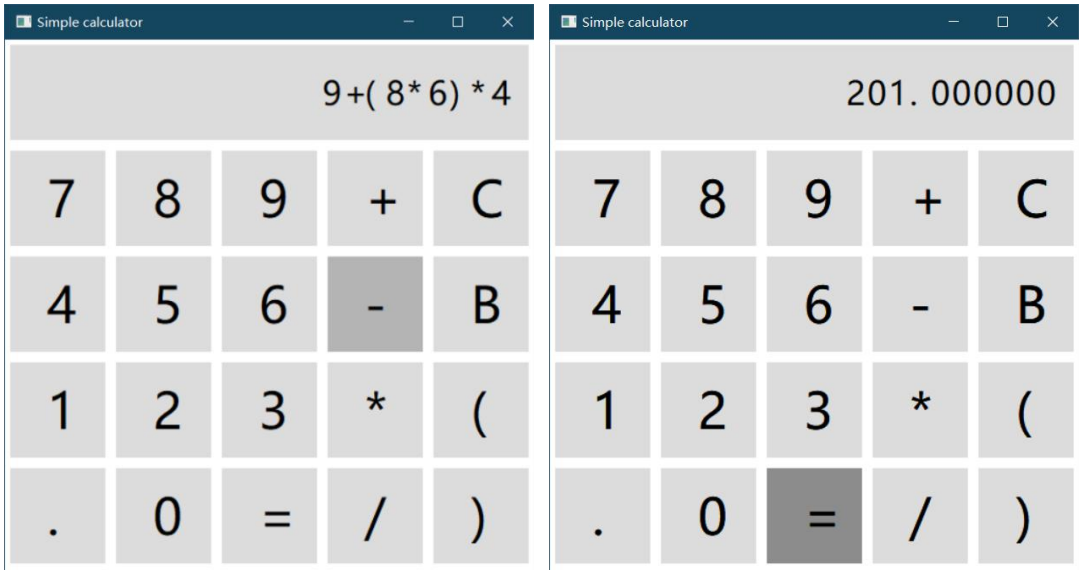


figure2 键盘&鼠标输入：运算式及结果、按键显示

三 实验总结与建议

（总结实验实施过程，说明实验过程中遇到的问题与解决方案；提出实验环节的建议）

实验实施过程：先构思；画框架图；编写伪代码；用实际代码实现；运行调试；debug 直至无错误并得到预期结果。

实验解决方案：实验中主要实现消息和处理消息，编写各个消息的处理代码以及界面函数即可。

附录：源代码

```
#include <windows.h>
#include<iostream>
#include<math.h>
#include<string>
#define length 100
using namespace std;
const char g_szClassName[] = "myWindowClass";
int x,y,key;
int draw;
int p=0;
double ans=0;
char com[length];
int position=-1;
template<class type>
class stack
{
public:
    stack(type a)
    {
        for (int i = 0; i < length; i++)
            link[i] = a;
    }
    type link[length];
};
class cal_class
{
public:
    string command;
    double num[length] ;
    int brackets[length] ; //1-{,2-[,3-(,4-},5-],6-)
    cal_class(char com[]){reset();command = com;}
    double CALCULATE();
private:
    void reset();
    int former(int i);
    int later(int i, int j);
    void spl();
    double calcualte_simple(int n, double a, double b);
    double calcualte_result();
    double calcualte();
};
void cal_class::reset()
{
    for (int i = 0; i < length; i++)
    {
        command = "";
```

```
47.         num[i] = 0;//num，brackets 默认初值为 0,'0'
48.         brackets[i] = 0;
49.     }
50. }
51. int cal_class::former(int i)
52. {
53.     int j;
54.     for (j = i + 1; j < int(command.size()) && j < length; j++)
55.     {
56.         if (int(command[j]) >= 48 && int(command[j]) <= 57)
57.             num[i] = num[i] * 10 + command[j] - '0';
58.         else
59.             break;
60.     }
61.     return j;
62. }
63. int cal_class::later(int i, int j)
64. {
65.     if (int(command[j]) == 46)
66.     {
67.         if (!(command[j + 1] >= 48 && command[j + 1] <= 57))
68.             return 0;
69.         else
70.         {
71.             for (int k = j + 1; k < int(command.size()) && k < length; k++)
72.             {
73.                 if (int(command[k]) >= 48 && int(command[k]) <= 57)
74.                 {
75.                     num[i] = num[i] + (command[k] - '0') / double(pow(10, k - j));
76.                     //最后为数字结尾
77.                     if (k == int(command.size()) - 1) return command.size() - 1;
78.                 }
79.                 else
80.                     return k;
81.             }
82.         }
83.     }
84.     else
85.         return j;
86.     return j;
87. }
88. void cal_class::spl()
89. {
90.     for (int i = 0; i < int(command.size()) && i < length; i++)
91.     {
92.         switch (command[i])
93.         {
94.         case '{':
95.             brackets[i] = 1;
```

```
96.             break;
97.         case '}':
98.             brackets[i] = 2;
99.             break;
100.        case '[':
101.            brackets[i] = 3;
102.            break;
103.        case ']':
104.            brackets[i] = 4;
105.            break;
106.        case '(':
107.            brackets[i] = 5;
108.            break;
109.        case ')':
110.            brackets[i] = 6;
111.            break;
112.        case '+':
113.            brackets[i] = 7;
114.            break;
115.        case '-':
116.            brackets[i] = 8;
117.            break;
118.        case '*':
119.            brackets[i] = 9;
120.            break;
121.        case '/':
122.            brackets[i] = 10;
123.            break;
124.        case '.':
125.            brackets[i] = 11;
126.            break;
127.        }
128.    }
129.    for (int i = 0; i < int(command.size()) && i < length; i++)
130.    {
131.        int j, k;
132.        if (int(command[i]) >= 48 && int(command[i]) <= 57)
133.        {
134.            num[i] = command[i] - '0';
135.            j = former(i);
136.            k = later(i, j);
137.            if (k != 0){i = k;}
138.            else{i = j;cout << "wrong\n";}
139.        }
140.    }
141.}
142.double cal_class::calcualte_simple(int n, double a, double b)
143.{
144.    if (n == 7)return a + b;
```

```
    if (n == 8)return a - b;
    if (n == 9)return a * b;
    if (n == 10 && b != 0)return a / b;
    return 0;
}
double cal_class::calcualte_result()
{
    int order[11] = { 0,3,0,3,0,3,0,1,1,2,2 };//加减乘除 7,8,9,10
    double result = 999.99;
    //操作符栈
    stack <int> oper(0);
    int position_oper = 0;
    //操作数栈
    stack <double> number(0);
    int position_number = 0;
    for (int i = 0; i < int(command.size()) && i < length; i++)
    {
        if (num[i] != 0)//数值数组非 0 存入
        {
            number.link[position_number + 1] = num[i];
            position_number++;
        }
        if (brackets[i] != 0 && brackets[i] != 11)//操作符数组非 0 且非.存入
        {
            if (order[oper.link[position_oper]] < order[brackets[i]] || order[oper.link[po
sition_oper]] == 3)//优先级高于现栈顶优先级，压入
            {
                oper.link[position_oper + 1] = brackets[i];
                position_oper++;
            }
            else
            {
                if (brackets[i] == 2 || brackets[i] == 4 || brackets[i] == 6)
                {
                    for (;;)
                    {
                        if (oper.link[position_oper] == brackets[i] - 1)
                        {
                            position_oper--;
                            break;
                        }
                        else
                        {
                            number.link[position_number - 1] = calcualte_simple(oper.link[
position_oper], \
                                                                number.link[position_number
 - 1], number.link[position_number]);
                            position_number--;
                            position_oper--;
```

```
                        }
                    }
                }
                else
                {
                    if (position_number > 1)
                    {
                        if (position_oper > 0)
                        {
                            number.link[position_number - 1] = calcualte_simple(oper.link[position_oper], \
                                                                                number.link[position_number - 1], number.link[position_number]);
                            position_number--;
                            position_oper--;
                        }
                    }
                    oper.link[position_oper + 1] = brackets[i];
                    position_oper++;
                }
            }
        }
    }
    if (oper.link[position_oper] == 9 || oper.link[position_oper] == 10)
    {
        number.link[position_number - 1] = calcualte_simple(oper.link[position_oper], \
                                            number.link[position_number - 1], number.link[position_number]);
        position_number--;
        position_oper--;
    }
    for (int i = 1; i <= position_oper; i++)
    {
        number.link[i + 1] = calcualte_simple(oper.link[i], \
                                                number.link[i], number.link[i + 1]);
    }
    result = number.link[position_number];
    return result;
}
double cal_class::calcualte()
{
    int wrong = 0;
    //left 从第二个存储位置开始存
    stack <int> left('0');
    int position = -1;
    //判断括号是否匹配
    for (int i = 0; i < int(command.size()) && i < length; i++)
    {
        if (brackets[i] == 1 || brackets[i] == 3 || brackets[i] == 5)
```

```
        {
            left.link[position + 1] = brackets[i];
            position++;
        }
        if (brackets[i] == 2 || brackets[i] == 4 || brackets[i] == 6)
        {
            if (position == -1){wrong = i;break;}
            if (brackets[i] == 2)
            {
                if (left.link[position] != 1){wrong = i;break;}
                else{left.link[position] = 0;position--;}
            }
            if (brackets[i] == 4)
            {
                if (left.link[position] != 3){wrong = i;break;}
                else{left.link[position] = 0;position--;}
            }
            if (brackets[i] == 6)
            {
                if (left.link[position] != 5){wrong = i;break;}
                else{left.link[position] = 0;position--;}
            }
        }
    }
    if (wrong != 0){return 999.99;cout << "wrong" << endl;}
    else
    {
        if (position == -1)//正确，输出计算结果
        {
            double result = calcualte_result();
            if (result == 999.99){return 999.99;cout << "wrong" << endl;}
            else return result;
        }
        else{return 999.99;cout << "wrong" << endl;}
    }
}
double cal_class:: CALCULATE(){spl();return calcualte();}
void Text(HDC hdc)
{
    //字体设置大小，颜色
    HFONT hFont = CreateFont(60, 0, 0, 0, FW_DONTCARE, 0, 0, 0, GB2312_CHARSET,
                             OUT_DEFAULT_PRECIS, CLIP_DEFAULT_PRECIS, DEFAULT_QUALITY,
                             DEFAULT_PITCH|FF_DONTCARE, "微软雅黑");//创建字体
    SelectObject(hdc, hFont);//选择字体
    SetBkMode(hdc, TRANSPARENT);
    char c[128];
    static int ind=10;
    int len=sprintf(c,"%d",ind);
    //绘制默认书写按钮文字
```

```
    for(int i=0; i<3; i++)
    {
        for(int j=0; j<3; j++)
        {
            ind=3*j+i+1;
            len=sprintf(c,"%d",ind);
            TextOut(hdc,40+i*100, 320-j*100,TEXT(c),len);
        }
    }
    TextOut(hdc,40,420,TEXT("."),1);
    TextOut(hdc,140,420,TEXT("0"),1);
    TextOut(hdc,240,420,TEXT("="),1);
    TextOut(hdc,340,120,TEXT("+"),1);
    TextOut(hdc,340,220,TEXT("-"),1);
    TextOut(hdc,340,320,TEXT("*"),1);
    TextOut(hdc,340,420,TEXT("/"),1);
    TextOut(hdc,440,220,TEXT("B"),1);
    TextOut(hdc,440,420,TEXT(")"),1);
    TextOut(hdc,440,320,TEXT("("),1);
    TextOut(hdc,440,120,TEXT("C"),1);
}
// Step 4: the Window Procedure
LRESULT CALLBACK WndProc(HWND hwnd, UINT msg, WPARAM wParam, LPARAM lParam)
{
    switch(msg)
    {
    case WM_CLOSE:
        cout<<" WM_CLOSE"<<endl;
        DestroyWindow(hwnd);
        break;
    case WM_DESTROY:
        cout<<" WM_DESTROY"<<endl;
        PostQuitMessage(0);
        break;
    case WM_CREATE:
        cout<<" WM_CREATE"<<endl;
        SetTimer(hwnd,1,100,NULL);
        draw=1;
        break;
    case WM_MOVE:
        x=LOWORD(lParam);
        y=HIWORD(lParam);
        cout<<" WM_MOVING at "<<x<<" "<<y<<endl;
        break;
    case WM_SIZE:
        cout<<" WM_SIZE"<<endl;
        break;
    case WM_LBUTTONDOWN:
        x=LOWORD(lParam);
```

```
        y=HIWORD(lParam);
        cout<<" WM_LBUTTONDOWN at "<<x<<" "<<y<<endl;
        p=1;
        //输入表达字符
        {
            for(int i=0; i<5; i++)
            {
                for(int j=0; j<4; j++)
                {
                    if(x>=5+i*100&&x<=95+i*100)
                    {
                        if(y>=105+j*100&&y<=195+j*100)
                        {
                            //1---9
                            if(i<=2&&j<=2)
                            {
                                char temp[2];
                                if(j==0)
                                    snprintf(temp, sizeof(temp), "%d", 3*j+i+1+6);
                                if(j==2)
                                    snprintf(temp, sizeof(temp), "%d", 3*j+i+1-6);
                                if(j==1)
                                    snprintf(temp, sizeof(temp), "%d", 3*j+i+1);
                                com[++position]=temp[0];
                            }
                            //第四排
                            if(j==3)
                            {
                                if(i==0)
                                    com[++position]='.';
                                if(i==1)
                                    com[++position]='0';
                                if(i==2)
                                {
                                    com[++position]='=';
                                    cal_class a(com);
                                    ans=a.CALCULATE();
                                    for(int i=0; i<length; i++)
                                        com[i]='0';
                                    position=-1;
                                }
                                if(i==3)
                                    com[++position]='/';
                                if(i==4)
                                    com[++position]=')';
                            }
                            //第四列
                            if(i==3)
                            {
```

```
                                if(j==0)
                                    com[++position]='+';
                                if(j==1)
                                    com[++position]='-';
                                if(j==2)
                                    com[++position]='*';
                            }
                            //第五列
                            if(i==4)
                            {
                                if(j==0)
                                {
                                    position=-1;
                                    ans=0;
                                    break;
                                }
                                if(j==1)
                                {
                                    position--;
                                    break;
                                }
                                if(j==2)
                                    com[++position]='(';
                            }

                        }
                    }
                }
            }

        }

        break;
    case WM_LBUTTONUP:
        x=LOWORD(lParam);
        y=HIWORD(lParam);
        cout<<" WM_LBUTTONUP at "<<x<<" "<<y<<endl;
        p=0;
        break;
    case WM_MOUSEMOVE:
        cout<<" WM_MOUSEMOVE "<<endl;
        x=LOWORD(lParam);
        y=HIWORD(lParam);
        cout<<" WM_MOUSEMOVE at "<<x<<" "<<y<<endl;
        break;
    case WM_RBUTTONDOWN:
        x=LOWORD(lParam);
        y=HIWORD(lParam);
        cout<<" WM_RBUTTONDOWN at "<<x<<" "<<y<<endl;
```

```
433.         break;
434.     case WM_KEYDOWN:
435.         key=wParam;
436.         if((key>=48&&key<=57)||key==88||key==191||key==187||key==219||key==221)
437.         {
438.             if(key>=48&&key<=57)
439.                 com[++position]=key-48+'0';
440.             if(key==88)
441.                 com[++position]='*';
442.             if(key==191)
443.                 com[++position]='/';
444.             if(key==187)
445.             {
446.                 com[++position]='=';
447.                 cal_class a(com);
448.                 ans=a.CALCULATE();
449.                 for(int i=0; i<length; i++)
450.                     com[i]='0';
451.                 position=-1;
452.             }
453.             if(key==219)
454.                 com[++position]='(';
455.             if(key==221)
456.                 com[++position]=')';
457.         }
458.         cout<<" WM_KEYDOWN : "<<key<<endl;
459.         break;
460.     case WM_KEYUP:
461.         key=wParam;
462.         cout<<" WM_KEYUP : "<<key<<endl;
463.         break;
464.     case WM_TIMER:
465.         cout<<" WM_TIMER"<<endl;
466.         if(draw)
467.         {
468.             HDC hdc=GetDC(hwnd);
469.             HBRUSH hBrush;
470.             RECT rect;
471.             //背景白色的最大矩形框
472.             hBrush = CreateSolidBrush(RGB(255,255,255));
473.             SetRect(&rect, 0, 0,500,500);
474.             FillRect(hdc, &rect, hBrush);
475.             //绘制默认输入显示区域以及按钮区域
476.             hBrush = CreateSolidBrush(RGB(220,220,220));
477.             SetRect(&rect, 5,5,495,95);
478.             FillRect(hdc, &rect, hBrush);
479.             for(int i=0; i<5; i++)
480.             {
481.                 for(int j=0; j<4; j++)
```

```
482.                 {
483.                     SetRect(&rect, 5+i*100, 105+j*100,95+i*100,195+j*100);
484.                     FillRect(hdc, &rect, hBrush);
485.                 }
486.             }
487.             //鼠标在上方
488.             hBrush = CreateSolidBrush(RGB(180,180,180));
489.             for(int i=0; i<5; i++)
490.             {
491.                 for(int j=0; j<4; j++)
492.                 {
493.                     if(x>=5+i*100&&x<=95+i*100)
494.                     {
495.                         if(y>=105+j*100&&y<=195+j*100)
496.                         {
497.                             SetRect(&rect, 5+i*100, 105+j*100,95+i*100,195+j*100);
498.                             FillRect(hdc, &rect, hBrush);
499.                         }
500.                     }
501.                 }
502.             }
503.             //左键 down
504.             if(p==1)
505.             {
506.                 //按下绘制加深矩形
507.                 {
508.                     HDC hdc=GetDC(hwnd);
509.                     HBRUSH hBrush;
510.                     RECT rect;
511.                     hBrush = CreateSolidBrush(RGB(140,140,140));
512.                     for(int i=0; i<5; i++)
513.                     {
514.                         for(int j=0; j<4; j++)
515.                         {
516.
517.                             if(x>=5+i*100&&x<=95+i*100)
518.                             {
519.                                 if(y>=105+j*100&&y<=195+j*100)
520.                                 {
521.
522.                                     SetRect(&rect, 5+i*100, 105+j*100,95+i*100,195+j*100);
523.                                     FillRect(hdc, &rect, hBrush);
524.                                 }
525.                             }
526.                         }
527.                     }
528.                     Text(hdc);
529.                     Sleep(50);
```

```
                }
            }
            Text(hdc);
            {
                //字体设置大小，颜色
                HFONT hFont = CreateFont(40, 0, 0, 0, FW_DONTCARE, 0, 0, 0, GB2312_CHARSET,
                                         OUT_DEFAULT_PRECIS, CLIP_DEFAULT_PRECIS, DEFAULT_QUALITY,
                                         DEFAULT_PITCH|FF_DONTCARE, "微软雅黑");//创建字体
                SelectObject(hdc, hFont);//选择字体
                SetBkMode(hdc, TRANSPARENT);

                if(position==-1)
                {
                    string a=to_string(ans);
                    char b[30];
                    for(int i=0; i<int(a.size()); i++)
                        b[i]=a[i];
                    for(int i=int(a.size()); i>=0; i--)
                    {
                        char c[2];
                        c[0]=b[i];
                        TextOut(hdc,480+(i-int(a.size()))*20,30,c,1);
                    }
                }
                else
                {
                    char str[30];
                    for(int i=position; i>=0; i--)
                    {
                        str[0]=com[i];
                        TextOut(hdc,460+(i-position)*20,30,TEXT(str),1);
                    }
                }
            }
            hBrush = NULL;
            ReleaseDC(hwnd,hdc);
        }
        break;
    default:
        return DefWindowProc(hwnd, msg, wParam, lParam);
    }
    return 0;
}
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,LPSTR lpCmdLine, int nCmdShow)
{
    WNDCLASSEX wc;
```

```
576.    HWND hwnd;
577.    MSG Msg;
578.    //Step 1: Registering the Window Class
579.    wc.cbSize        = sizeof(WNDCLASSEX);
580.    wc.style         = 0;
581.    wc.lpfnWndProc   = WndProc;
582.    wc.cbClsExtra    = 0;
583.    wc.cbWndExtra    = 0;
584.    wc.hInstance     = hInstance;
585.    wc.hIcon         = LoadIcon(NULL, IDI_APPLICATION);
586.    wc.hCursor       = LoadCursor(NULL, IDC_ARROW);
587.    wc.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
588.    wc.lpszMenuName  = NULL;
589.    wc.lpszClassName = g_szClassName;
590.    wc.hIconSm       = LoadIcon(NULL, IDI_APPLICATION);
591.    if(!RegisterClassEx(&wc))
592.    {
593.        MessageBox(NULL, "Window Registration Failed!", "Error!",
594.                   MB_ICONEXCLAMATION | MB_OK);
595.        return 0;
596.    }
597.    // Step 2: Creating the Window
598.    hwnd = CreateWindowEx(
599.               WS_EX_CLIENTEDGE,
600.               g_szClassName,
601.               "Simple calculator",
602.               WS_OVERLAPPEDWINDOW,
603.               CW_USEDEFAULT, CW_USEDEFAULT, 500, 500,
604.               NULL, NULL, hInstance, NULL);
605.    SetWindowPos(hwnd,HWND_NOTOPMOST,500,100,520,545,SWP_SHOWWINDOW );
606.    if(hwnd == NULL)
607.    {
608.        MessageBox(NULL, "Window Creation Failed!", "Error!",
609.                   MB_ICONEXCLAMATION | MB_OK);
610.        return 0;
611.    }
612.    ShowWindow(hwnd, nCmdShow);
613.    UpdateWindow(hwnd);
614.    // Step 3: The Message Loop
615.    while(GetMessage(&Msg, NULL, 0, 0) > 0)
616.    {
617.        TranslateMessage(&Msg);
618.        DispatchMessage(&Msg);
619.    }
620.    return Msg.wParam;
621.}
```