

```

1  #define UNICODE
2  #include<windows.h>
3  #include"HeaderForClientOfAggregationComponentWithRegFile.h"
4  // global function declarations
5  LRESULT CALLBACK WndProc(HWND,UINT,WPARAM,LPARAM);
6  // global variable declarations
7  ISum *pISum=NULL;
8  ISubtract *pISubtract=NULL;
9  IMultiplication *pIMultiplication=NULL;
10 IDivision *pIDivision=NULL;
11 // WinMain
12 int WINAPI WinMain(HINSTANCE hInstance,HINSTANCE hPrevInstance,
13                   LPSTR lpCmdLine,int nCmdShow)
14 {
15     // variable declarations
16     WNDCLASSEX wndclass;
17     HWND hwnd;
18     MSG msg;
19     TCHAR AppName[]=TEXT("ComClient");
20     HRESULT hr;
21     // code
22     // COM Initialization
23     hr=CoInitialize(NULL);
24     if(FAILED(hr))
25     {
26         MessageBox(NULL,TEXT("COM Library Can Not Be Initialized.\nProgram Will
27             Now Exit."),TEXT("Program Error"),MB_OK);
28         exit(0);
29     }
30     // WNDCLASSEX initialization
31     wndclass.cbSize=sizeof(wndclass);
32     wndclass.style=CS_HREDRAW|CS_VREDRAW;
33     wndclass.cbClsExtra=0;
34     wndclass.cbWndExtra=0;
35     wndclass.lpfnWndProc=WndProc;
36     wndclass.hIcon=LoadIcon(NULL,IDI_APPLICATION);
37     wndclass.hCursor=LoadCursor(NULL,IDC_ARROW);
38     wndclass.hbrBackground=(HBRUSH)GetStockObject(WHITE_BRUSH);
39     wndclass.hInstance=hInstance;
40     wndclass.lpszClassName=AppName;
41     wndclass.lpszMenuName=NULL;
42     wndclass.hIconSm=LoadIcon(NULL,IDI_APPLICATION);
43     // register window class
44     RegisterClassEx(&wndclass);
45     // create window
46     hwnd=CreateWindow(AppName,
47                     TEXT("Client Of COM Dll Server"),
48                     WS_OVERLAPPEDWINDOW,
49                     CW_USEDEFAULT,
50                     CW_USEDEFAULT,
51                     CW_USEDEFAULT,

```

```

52         NULL,
53         NULL,
54         hInstance,
55         NULL);
56     ShowWindow(hwnd,nCmdShow);
57     UpdateWindow(hwnd);
58     // message loop
59     while(GetMessage(&msg,NULL,0,0))
60     {
61         TranslateMessage(&msg);
62         DispatchMessage(&msg);
63     }
64     // COM Un-initialization
65     CoUninitialize();
66     return((int)msg.wParam);
67 }
68 // Window Procedure
69 LRESULT CALLBACK WndProc(HWND hwnd,UINT iMsg,WPARAM wParam,LPARAM lParam)
70 {
71     // function declarations
72     void SafeInterfaceRelease(void);
73     // variable declarations
74     HRESULT hr;
75     int iNum1,iNum2,iSum,iSubtraction,iMultiplication,iDivision;
76     TCHAR str[255];
77     // code
78     switch(iMsg)
79     {
80     case WM_CREATE:
81         hr=CoCreateInstance(CLSID_SumSubtract,NULL,CLSCTX_INPROC_SERVER,
82                             IID_ISum,(void **)&pISum);
83         if(FAILED(hr))
84         {
85             MessageBox(hwnd,TEXT("ISum Interface Can Not Be Obtained"),TEXT
86                         ("Error"),MB_OK);
87             DestroyWindow(hwnd);
88         }
89         // initialize arguments hardcoded
90         iNum1=65;
91         iNum2=45;
92         // call SumOfTwoIntegers() of ISum to get the sum
93         pISum->SumOfTwoIntegers(iNum1,iNum2,&iSum);
94         // display the result
95         wsprintf(str,TEXT("Sum Of %d And %d = %d"),iNum1,iNum2,iSum);
96         MessageBox(hwnd,str,TEXT("Result"),MB_OK);
97         // call QueryInterface() on ISum,to get ISubtract's pointer
98         hr=pISum->QueryInterface(IID_ISubtract,(void **)&pISubtract);
99         if(FAILED(hr))
100         {
101             MessageBox(hwnd,TEXT("ISubtract Interface Can Not Be Obtained"),TEXT
102                         ("Error"),MB_OK);
103             DestroyWindow(hwnd);

```

```
102     }
103     // as ISum is now not needed onwords, release it
104     pISum->Release();
105     pISum=NULL;// make relese interface NULL
106     // again initialize arguments hardcoded
107     iNum1=155;
108     iNum2=55;
109     // call SubtractionOfTwoIntegers() of ISubtract to get the subtraction
110     pISubtract->SubtractionOfTwoIntegers(iNum1,iNum2,&iSubtraction);
111     // display the result
112     wsprintf(str,TEXT("Subtraction Of %d And %d = %d"),iNum1,iNum2,iSubtraction);
113     MessageBox(hwnd,str,TEXT("Result"),MB_OK);
114     // call QueryInterface() on ISubtract,to get IMultiplication's pointer
115     hr=pISubtract->QueryInterface(IID_IMultiplication,(void **)&pIMultiplication);
116     if(FAILED(hr))
117     {
118         MessageBox(hwnd,TEXT("IMultiplication Interface Can Not Be
119         Obtained"),TEXT("Error"),MB_OK);
120         DestroyWindow(hwnd);
121     }
122     // as ISubtract is now not needed onwords, release it
123     pISubtract->Release();
124     pISubtract=NULL;// make relese interface NULL
125     // again initialize arguments hardcoded
126     iNum1=30;
127     iNum2=25;
128     // call MultiplicationOfTwoIntegers() of IMultiplication to get the
129     Multiplication
130     pIMultiplication->MultiplicationOfTwoIntegers
131     (iNum1,iNum2,&iMultiplication);
132     // display the result
133     wsprintf(str,TEXT("Multiplication Of %d And %d = %d"),iNum1,iNum2,iMultiplication);
134     MessageBox(hwnd,str,TEXT("Result"),MB_OK);
135     // call QueryInterface() on IMultiplication's to get IDivision pointer
136     hr=pIMultiplication->QueryInterface(IID_IDivision,(void **)&pIDivision);
137     if(FAILED(hr))
138     {
139         MessageBox(hwnd,TEXT("IDivision Interface Can Not Be Obtained"),TEXT
140         ("Error"),MB_OK);
141         DestroyWindow(hwnd);
142     }
143     // as IMultiplication is now not needed onwords, release it
144     pIMultiplication->Release();
145     pIMultiplication=NULL;// make relese interface NULL
146     // again initialize arguments hardcoded
147     iNum1=200;
148     iNum2=25;
149     // call DivisionOfTwoIntegers() of IDivision to get the Division
150     pIDivision->DivisionOfTwoIntegers(iNum1,iNum2,&iDivision);
```

```
147     // display the result
148     wsprintf(str,TEXT("Division Of %d And %d = %d"),iNum1,iNum2,iDivision);
149     MessageBox(hwnd,str,TEXT("Result"),MB_OK);
150     // finally release IDivision
151     pIDivision->Release();
152     pIDivision=NULL;// make relesead interface NULL
153     // exit the application
154     DestroyWindow(hwnd);
155     break;
156 case WM_DESTROY:
157     SafeInterfaceRelease();
158     PostQuitMessage(0);
159     break;
160 }
161 return(DefWindowProc(hwnd,iMsg,wParam,lParam));
162 }
163 void SafeInterfaceRelease(void)
164 {
165     // code
166     if(pISum)
167     {
168         pISum->Release();
169         pISum=NULL;
170     }
171     if(pISubtract)
172     {
173         pISubtract->Release();
174         pISubtract=NULL;
175     }
176     if(pIMultiplication)
177     {
178         pIMultiplication->Release();
179         pIMultiplication=NULL;
180     }
181     if(pIDivision)
182     {
183         pIDivision->Release();
184         pIDivision=NULL;
185     }
186 }
187
```