```
 1
 2  #include<windows.h>
 3  #include<process.h>
 4  #include"HeaderForClientOfContainmentComponentWithRegFile.h"
 5  // global function declarations
 6  LRESULT CALLBACK WndProc(HWND,UINT,WPARAM,LPARAM);
 7  // global variable declarations
 8  ISum *pISum=NULL;
 9  ISubtract *pISubtract=NULL;
10  IMultiplication *pIMultiplication=NULL;
11  IDivision *pIDivision=NULL;
12  // WinMain
13  int WINAPI WinMain(HINSTANCE hInstance,HINSTANCE hPrevInstance,
14                     LPSTR lpCmdLine,int nCmdShow)
15  {
16      // variable declarations
17      WNDCLASSEX wndclass;
18      HWND hwnd;
19      MSG msg;
20      TCHAR AppName[]=TEXT("ComClient");
21      HRESULT hr;
22      // code
23      // COM Initialization
24      hr=CoInitialize(NULL);
25      if(FAILED(hr))
26      {
27          MessageBox(NULL,TEXT("COM Library Can Not Be Initialized.\nProgram Will
              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                          CW_USEDEFAULT,
53                          NULL,
54                          NULL,
55                          hInstance,
56                          NULL);
57      ShowWindow(hwnd,nCmdShow);
58      UpdateWindow(hwnd);
59      // message loop
60      while(GetMessage(&msg,NULL,0,0))
61      {
62          TranslateMessage(&msg);
63          DispatchMessage(&msg);
64      }
65      // COM Un-initialization
66      CoUninitialize();
67      return((int)msg.wParam);
68  }
69  // Window Procedure
70  LRESULT CALLBACK WndProc(HWND hwnd,UINT iMsg,WPARAM wParam,LPARAM lParam)
71  {
72      // function declarations
73      void SafeInterfaceRelease(void);
74      // variable declarations
75      HRESULT hr;
76      int iNum1,iNum2,iSum,iSubtraction,iMultiplication,iDivision;
77      TCHAR str[255];
78      // code
79      switch(iMsg)
80      {
81      case WM_CREATE:
82          hr=CoCreateInstance(CLSID_SumSubtract,NULL,CLSCTX_INPROC_SERVER,
83                              IID_ISum,(void **)&pISum);
84          if(FAILED(hr))
85          {
86              MessageBox(hwnd,TEXT("ISum Interface Can Not Be Obtained"),TEXT
                    ("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
                    ("Error"),MB_OK);
```

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

```
147         pIDivision->DivisionOfTwoIntegers(iNum1,iNum2,&iDivision);
148         // display the result
149         wsprintf(str,TEXT("Division Of %d And %d = %d"),iNum1,iNum2,iDivision);
150         MessageBox(hwnd,str,TEXT("Result"),MB_OK);
151         // finally release IDivision
152         pIDivision->Release();
153         pIDivision=NULL;// make relesed interface NULL
154         // exit the application
155         DestroyWindow(hwnd);
156         break;
157     case WM_DESTROY:
158         SafeInterfaceRelease();
159         PostQuitMessage(0);
160         break;
161     }
162     return(DefWindowProc(hwnd,iMsg,wParam,lParam));
163 }
164 void SafeInterfaceRelease(void)
165 {
166     // code
167     if(pISum)
168     {
169         pISum->Release();
170         pISum=NULL;
171     }
172     if(pISubtract)
173     {
174         pISubtract->Release();
175         pISubtract=NULL;
176     }
177     if(pIMultiplication)
178     {
179         pIMultiplication->Release();
180         pIMultiplication=NULL;
181     }
182     if(pIDivision)
183     {
184         pIDivision->Release();
185         pIDivision=NULL;
186     }
187 }
188
```