```
 1
 2 #include<windows.h>
 3 #include"ContainmentInnerComponentWithRegFile.h"
 4 #include"ContainmentOuterComponentWithRegFile.h"
 5 // class declarations
 6 class CSumSubtract:public ISum,ISubtract,IMultiplication,IDivision
 7 {
 8 private:
 9     long m_cRef;
10     IMultiplication *m_pIMultiplication;
11     IDivision *m_pIDivision;
12 public:
13     // constructor method declarations
14     CSumSubtract(void);
15     // destructor method declarations
16     ~CSumSubtract(void);
17     // IUnknown specific method declarations (inherited)
18     HRESULT __stdcall QueryInterface(REFIID,void **);
19     ULONG __stdcall AddRef(void);
20     ULONG __stdcall Release(void);
21     // ISum specific method declarations (inherited)
22     HRESULT __stdcall SumOfTwoIntegers(int,int,int *);
23     // ISubtract specific method declarations (inherited)
24     HRESULT __stdcall SubtractionOfTwoIntegers(int,int,int *);
25     // IMultiplication specific method declarations (inherited)
26     HRESULT __stdcall MultiplicationOfTwoIntegers(int,int,int *);
27     // IDivision specific method declarations (inherited)
28     HRESULT __stdcall DivisionOfTwoIntegers(int,int,int *);
29     // custom method for inner component creation
30     HRESULT __stdcall InitializeInnerComponent(void);
31 };
32 class CSumSubtractClassFactory:public IClassFactory
33 {
34 private:
35     long m_cRef;
36 public:
37     // constructor method declarations
38     CSumSubtractClassFactory(void);
39     // destructor method declarations
40     ~CSumSubtractClassFactory(void);
41     // IUnknown specific method declarations (inherited)
42     HRESULT __stdcall QueryInterface(REFIID,void **);
43     ULONG __stdcall AddRef(void);
44     ULONG __stdcall Release(void);
45     // IClassFactory specific method declarations (inherited)
46     HRESULT __stdcall CreateInstance(IUnknown *,REFIID,void **);
47     HRESULT __stdcall LockServer(BOOL);
48 };
49 // global variable declarations
50 long glNumberOfActiveComponents=0;// number of active components
51 long glNumberOfServerLocks=0;// number of locks on this dll
52 // DllMain
```

```
53  BOOL WINAPI DllMain(HINSTANCE hDll,DWORD dwReason,LPVOID Reserved)
54  {
55      // code
56      switch(dwReason)
57      {
58      case DLL_PROCESS_ATTACH:
59          break;
60      case DLL_PROCESS_DETACH:
61          break;
62      }
63  return(TRUE);
64  }
65  // Implementation Of CSumSubtract's Constructor Method
66  CSumSubtract::CSumSubtract(void)
67  {
68      // code
69      // initialization of private data members
70      m_pIMultiplication=NULL;
71      m_pIDivision=NULL;
72      m_cRef=1;// hardcoded initialization to anticipate possible failure of
          QueryInterface()
73      InterlockedIncrement(&glNumberOfActiveComponents);// increment global counter
74  }
75  // Implementation Of CSumSubtract's Destructor Method
76  CSumSubtract::~CSumSubtract(void)
77  {
78      // code
79      InterlockedDecrement(&glNumberOfActiveComponents);// decrement global counter
80      if(m_pIMultiplication)
81      {
82          m_pIMultiplication->Release();
83          m_pIMultiplication=NULL;
84      }
85      if(m_pIDivision)
86      {
87          m_pIDivision->Release();
88          m_pIDivision=NULL;
89      }
90  }
91  // Implementation Of CSumSubtract's IUnknown's Methods
92  HRESULT CSumSubtract::QueryInterface(REFIID riid,void **ppv)
93  {
94      // code
95      if(riid==IID_IUnknown)
96          *ppv=static_cast<ISum *>(this);
97      else if(riid==IID_ISum)
98          *ppv=static_cast<ISum *>(this);
99      else if(riid==IID_ISubtract)
100         *ppv=static_cast<ISubtract *>(this);
101     else if(riid==IID_IMultiplication)
102         *ppv=static_cast<IMultiplication *>(this);
103     else if(riid==IID_IDivision)
```

```
104             *ppv=static_cast<IDivision *>(this);
105         else
106         {
107             *ppv=NULL;
108             return(E_NOINTERFACE);
109         }
110         reinterpret_cast<IUnknown *>(*ppv)->AddRef();
111         return(S_OK);
112     }
113     ULONG CSumSubtract::AddRef(void)
114     {
115         // code
116         InterlockedIncrement(&m_cRef);
117         return(m_cRef);
118     }
119     ULONG CSumSubtract::Release(void)
120     {
121         // code
122         InterlockedDecrement(&m_cRef);
123         if(m_cRef==0)
124         {
125             delete(this);
126             return(0);
127         }
128         return(m_cRef);
129     }
130     // Implementation Of ISum's Methods
131     HRESULT CSumSubtract::SumOfTwoIntegers(int num1,int num2,int *pSum)
132     {
133         // code
134         *pSum=num1+num2;
135         return(S_OK);
136     }
137     // Implementation Of ISubtract's Methods
138     HRESULT CSumSubtract::SubtractionOfTwoIntegers(int num1,int num2,int *pSubtract)
139     {
140         // code
141         *pSubtract=num1-num2;
142         return(S_OK);
143     }
144     // Implementation Of IMultiplication's Methods
145     HRESULT CSumSubtract::MultiplicationOfTwoIntegers(int num1,int num2,int
          *pMultiplication)
146     {
147         // code
148         // delegate to inner component
149         m_pIMultiplication->MultiplicationOfTwoIntegers(num1,num2,pMultiplication);
150         return(S_OK);
151     }
152     // Implementation Of IDivision's Methods
153     HRESULT CSumSubtract::DivisionOfTwoIntegers(int num1,int num2,int *pDivision)
154     {
```

```
155     // code
156     // delegate to inner component
157     m_pIDivision->DivisionOfTwoIntegers(num1,num2,pDivision);
158     return(S_OK);
159 }
160 HRESULT CSumSubtract::InitializeInnerComponent(void)
161 {
162     // variable declarations
163     HRESULT hr;
164     // code
165     hr=CoCreateInstance(CLSID_MultiplicationDivision,NULL,CLSCTX_INPROC_SERVER,
166                         IID_IMultiplication,(void **)&m_pIMultiplication);
167     if(FAILED(hr))
168     {
169         MessageBox(NULL,TEXT("IMultiplication Interface Can Not Be Obtained From
              Inner Component."),TEXT("Error"),MB_OK);
170         return(E_FAIL);
171     }
172     hr=m_pIMultiplication->QueryInterface(IID_IDivision,(void **)&m_pIDivision);
173     if(FAILED(hr))
174     {
175         MessageBox(NULL,TEXT("IDivision Interface Can Not Be Obtained From Inner
              Component."),TEXT("Error"),MB_OK);
176         return(E_FAIL);
177     }
178     return(S_OK);
179 }
180 // Implementation Of CSumSubtractClassFactory's Constructor Method
181 CSumSubtractClassFactory::CSumSubtractClassFactory(void)
182 {
183     // code
184     m_cRef=1;// hardcoded initialization to anticipate possible failure of
            QueryInterface()
185 }
186 // Implementation Of CSumSubtractClassFactory's Destructor Method
187 CSumSubtractClassFactory::~CSumSubtractClassFactory(void)
188 {
189     // code
190 }
191 // Implementation Of CSumSubtractClassFactory's IClassFactory's IUnknown's
      Methods
192 HRESULT CSumSubtractClassFactory::QueryInterface(REFIID riid,void **ppv)
193 {
194     // code
195     if(riid==IID_IUnknown)
196         *ppv=static_cast<IClassFactory *>(this);
197     else if(riid==IID_IClassFactory)
198         *ppv=static_cast<IClassFactory *>(this);
199     else
200     {
201         *ppv=NULL;
202         return(E_NOINTERFACE);
```

```
203     }
204     reinterpret_cast<IUnknown *>(*ppv)->AddRef();
205     return(S_OK);
206 }
207 ULONG CSumSubtractClassFactory::AddRef(void)
208 {
209     // code
210     InterlockedIncrement(&m_cRef);
211     return(m_cRef);
212 }
213 ULONG CSumSubtractClassFactory::Release(void)
214 {
215     // code
216     InterlockedDecrement(&m_cRef);
217     if(m_cRef==0)
218     {
219         delete(this);
220         return(0);
221     }
222     return(m_cRef);
223 }
224 // Implementation Of CSumSubtractClassFactory's IClassFactory's Methods
225 HRESULT CSumSubtractClassFactory::CreateInstance(IUnknown *pUnkOuter,REFIID
      riid,void **ppv)
226 {
227     // variable declarations
228     CSumSubtract *pCSumSubtract=NULL;
229     HRESULT hr;
230     // code
231     if(pUnkOuter!=NULL)
232         return(CLASS_E_NOAGGREGATION);
233     // create the instance of component i.e. of CSumSubtract class
234     pCSumSubtract=new CSumSubtract;
235     if(pCSumSubtract==NULL)
236         return(E_OUTOFMEMORY);
237     // initialize the inner component
238     hr=pCSumSubtract->InitializeInnerComponent();
239     if(FAILED(hr))
240     {
241         MessageBox(NULL,TEXT("Failed To Initialize Inner Component"),TEXT
              ("Error"),MB_OK);
242         pCSumSubtract->Release();
243         return(hr);
244     }
245     // get the requested interface
246     hr=pCSumSubtract->QueryInterface(riid,ppv);
247     pCSumSubtract->Release();// anticipate possible failure of QueryInterface()
248     return(hr);
249 }
250 HRESULT CSumSubtractClassFactory::LockServer(BOOL fLock)
251 {
252     // code
```

```
253     if(fLock)
254         InterlockedIncrement(&glNumberOfServerLocks);
255     else
256         InterlockedDecrement(&glNumberOfServerLocks);
257     return(S_OK);
258 }
259 // Implementation Of Exported Functions From This Dll
260 HRESULT __stdcall DllGetClassObject(REFCLSID rclsid,REFIID riid,void **ppv)
261 {
262     // variable declaraions
263     CSumSubtractClassFactory *pCSumSubtractClassFactory=NULL;
264     HRESULT hr;
265     // code
266     if(rclsid!=CLSID_SumSubtract)
267         return(CLASS_E_CLASSNOTAVAILABLE);
268     // create class factory
269     pCSumSubtractClassFactory=new CSumSubtractClassFactory;
270     if(pCSumSubtractClassFactory==NULL)
271         return(E_OUTOFMEMORY);
272     hr=pCSumSubtractClassFactory->QueryInterface(riid,ppv);
273     pCSumSubtractClassFactory->Release();// anticipate possible failure of
        QueryInterface()
274     return(hr);
275 }
276 HRESULT __stdcall DllCanUnloadNow(void)
277 {
278     // code
279     if((glNumberOfActiveComponents==0) && (glNumberOfServerLocks==0))
280         return(S_OK);
281     else
282         return(S_FALSE);
283 }
284
```