

```
1  #define UNICODE
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 g1NumberOfActiveComponents=0;// number of active components
51 long g1NumberOfServerLocks=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
73     QueryInterface()
74     InterlockedIncrement(&g1NumberOfActiveComponents);// increment global counter
75 }
76 // Implementation Of CSumSubtract's Destructor Method
77 CSumSubtract::~~CSumSubtract(void)
78 {
79     // code
80     InterlockedDecrement(&g1NumberOfActiveComponents);// decrement global counter
81     if(m_pIMultiplication)
82     {
83         m_pIMultiplication->Release();
84         m_pIMultiplication=NULL;
85     }
86     if(m_pIDivision)
87     {
88         m_pIDivision->Release();
89         m_pIDivision=NULL;
90     }
91 }
92 // Implementation Of CSumSubtract's IUnknown's Methods
93 HRESULT CSumSubtract::QueryInterface(REFIID riid,void **ppv)
94 {
95     // code
96     if(riid==IID_IUnknown)
97         *ppv=static_cast<ISum *>(this);
98     else if(riid==IID_ISum)
99         *ppv=static_cast<ISum *>(this);
100     else if(riid==IID_ISubtract)
101         *ppv=static_cast<ISubtract *>(this);
102     else if(riid==IID_IMultiplication)
103         *ppv=static_cast<IMultiplication *>(this);
104     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
146     *pMultiplication)
147 {
148     // code
149     // delegate to inner component
150     m_pIMultiplication->MultiplicationOfTwoIntegers(num1,num2,pMultiplication);
151     return(S_OK);
152 }
153 // Implementation Of IDivision's Methods
154 HRESULT CSumSubtract::DivisionOfTwoIntegers(int num1,int num2,int *pDivision)
```



```

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
185               QueryInterface()
186 }
187 // Implementation Of CSumSubtractClassFactory's Destructor Method
188 CSumSubtractClassFactory::~CSumSubtractClassFactory(void)
189 {
190     // code
191 }
192 // Implementation Of CSumSubtractClassFactory's IClassFactory's IUnknown's
193 // Methods
194 HRESULT CSumSubtractClassFactory::QueryInterface(REFIID riid,void **ppv)
195 {
196     // code
197     if(riid==IID_IUnknown)
198         *ppv=static_cast<IClassFactory *>(this);
199     else if(riid==IID_IClassFactory)
200         *ppv=static_cast<IClassFactory *>(this);
201     else
202     {
203         *ppv=NULL;
204         return(E_NOINTERFACE);
205     }
206 }

```

```

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
226 riid, void **ppv)
227 {
228     // variable declarations
229     CSumSubtract *pCSumSubtract=NULL;
230     HRESULT hr;
231     // code
232     if(pUnkOuter!=NULL)
233         return(CLASS_E_NOAGGREGATION);
234     // create the instance of component i.e. of CSumSubtract class
235     pCSumSubtract=new CSumSubtract;
236     if(pCSumSubtract==NULL)
237         return(E_OUTOFMEMORY);
238     // initialize the inner component
239     hr=pCSumSubtract->InitializeInnerComponent();
240     if(FAILED(hr))
241     {
242         MessageBox(NULL,TEXT("Failed To Initialize Inner Component"),TEXT
243         ("Error"),MB_OK);
244         pCSumSubtract->Release();
245         return(hr);
246     }
247     // get the requested interface
248     hr=pCSumSubtract->QueryInterface(riid,ppv);
249     pCSumSubtract->Release();// anticipate possible failure of QueryInterface()
250     return(hr);
251 }
252 HRESULT CSumSubtractClassFactory::LockServer(BOOL fLock)
253 {
254     // 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
274         QueryInterface()
275     return(hr);
276 }
277 HRESULT __stdcall DllCanUnloadNow(void)
278 {
279     // code
280     if((glNumberOfActiveComponents==0) && (glNumberOfServerLocks==0))
281         return(S_OK);
282     else
283         return(S_FALSE);
284 }
```