```cpp
#define UNICODE
#include<windows.h>
#include"AggregationInnerComponentWithRegFile.h"
#include"AggregationOuterComponentWithRegFile.h"
// class declarations
class CSumSubtract:public ISum,ISubtract
{
private:
    long m_cRef;
    IUnknown *m_pIUnknownInner;
    IMultiplication *m_pIMultiplication;
    IDivision *m_pIDivision;
public:
    // constructor method declarations
    CSumSubtract(void);
    // destructor method declarations
    ~CSumSubtract(void);
    // IUnknown specific method declarations (inherited)
    HRESULT __stdcall QueryInterface(REFIID,void **);
    ULONG __stdcall AddRef(void);
    ULONG __stdcall Release(void);
    // ISum specific method declarations (inherited)
    HRESULT __stdcall SumOfTwoIntegers(int,int,int *);
    // ISubtract specific method declarations (inherited)
    HRESULT __stdcall SubtractionOfTwoIntegers(int,int,int *);
    // custom method for inner component creation
    HRESULT __stdcall InitializeInnerComponent(void);
};
class CSumSubtractClassFactory:public IClassFactory
{
private:
    long m_cRef;
public:
    // constructor method declarations
    CSumSubtractClassFactory(void);
    // destructor method declarations
    ~CSumSubtractClassFactory(void);
    // IUnknown specific method declarations (inherited)
    HRESULT __stdcall QueryInterface(REFIID,void **);
    ULONG __stdcall AddRef(void);
    ULONG __stdcall Release(void);
    // IClassFactory specific method declarations (inherited)
    HRESULT __stdcall CreateInstance(IUnknown *,REFIID,void **);
    HRESULT __stdcall LockServer(BOOL);
};
// global variable declarations
long glNumberOfActiveComponents=0;// number of active components
long glNumberOfServerLocks=0;// number of locks on this dll
// DllMain
BOOL WINAPI DllMain(HINSTANCE hDll,DWORD dwReason,LPVOID Reserved)
{
    // code
```

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

```
104     else if(riid==IID_IMultiplication)
105         return(m_pIUnknownInner->QueryInterface(riid,ppv));
106     else if(riid==IID_IDivision)
107         return(m_pIUnknownInner->QueryInterface(riid,ppv));
108     else
109     {
110         *ppv=NULL;
111         return(E_NOINTERFACE);
112     }
113     reinterpret_cast<IUnknown *>(*ppv)->AddRef();
114     return(S_OK);
115 }
116 ULONG CSumSubtract::AddRef(void)
117 {
118     // code
119     InterlockedIncrement(&m_cRef);
120     return(m_cRef);
121 }
122 ULONG CSumSubtract::Release(void)
123 {
124     // code
125     InterlockedDecrement(&m_cRef);
126     if(m_cRef==0)
127     {
128         delete(this);
129         return(0);
130     }
131     return(m_cRef);
132 }
133 // Implementation Of ISum's Methods
134 HRESULT CSumSubtract::SumOfTwoIntegers(int num1,int num2,int *pSum)
135 {
136     // code
137     *pSum=num1+num2;
138     return(S_OK);
139 }
140 // Implementation Of ISubtract's Methods
141 HRESULT CSumSubtract::SubtractionOfTwoIntegers(int num1,int num2,int *pSubtract)
142 {
143     // code
144     *pSubtract=num1-num2;
145     return(S_OK);
146 }
147 HRESULT CSumSubtract::InitializeInnerComponent(void)
148 {
149     // variable declarations
150     HRESULT hr;
151     // code
152     hr=CoCreateInstance(CLSID_MultiplicationDivision,
153                         reinterpret_cast<IUnknown *>(this),
154                         CLSCTX_INPROC_SERVER,
155                         IID_IUnknown,
```

```
156                             (void **)&m_pIUnknownInner);
157     if(FAILED(hr))
158     {
159         MessageBox(NULL,TEXT("IUnknown Interface Can Not Be Obtained From Inner
              Component."),TEXT("Error"),MB_OK);
160         return(E_FAIL);
161     }
162     hr=m_pIUnknownInner->QueryInterface(IID_IMultiplication,(void **)
          &m_pIMultiplication);
163     if(FAILED(hr))
164     {
165         MessageBox(NULL,TEXT("IMultiplication Interface Can Not Be Obtained From
              Inner Component."),TEXT("Error"),MB_OK);
166         m_pIUnknownInner->Release();
167         m_pIUnknownInner=NULL;
168         return(E_FAIL);
169     }
170     hr=m_pIUnknownInner->QueryInterface(IID_IDivision,(void **)&m_pIDivision);
171     if(FAILED(hr))
172     {
173         MessageBox(NULL,TEXT("IDivision Interface Can Not Be Obtained From Inner
              Component."),TEXT("Error"),MB_OK);
174         m_pIUnknownInner->Release();
175         m_pIUnknownInner=NULL;
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
```