```cpp
// inheritance_multiple_new.cpp
//
#ifdef _WIN32
   #pragma warning(disable:4996)
   #include <tchar.h>
   #include <windows.h>
   #include <conio.h>
   #include <direct.h>
#elif (defined __linux__) || (defined _AIX)
   #include <stdlib.h>
   #include <sys/types.h>
   #include <sys/stat.h>
   #include <unistd.h>
   typedef char _TCHAR;
   #define _tmain main
#endif

#include <stdio.h>
#include <string.h>
#include <fstream>
#include <iostream>
#include <iomanip>
using namespace std;

#include "MaticaObdlznikova.h"
#include "MaticaStvorcova.h"
#include "CisloKomplexne.h"
#include "MaticaStvorcovaKomplexna.h"

void GetNamesOfIOFiles(const char[], char[]);
//---------------------------------------------------------------------------
int _tmain(int argc, _TCHAR* argv[])
{
   my_class xx;
   ifstream in;
   char nameoffile[MAXLINE];
   int Exponent;
   unsigned m, n;

   cout << "\n  Tento program nacita z textoveho suboru stvorcovu maticu A,\n"
      << "z klavesnice prirodzene cislo k a na obrazovku a do textoveho\n"
      << "suboru vypise k-tu mocninu matice A spolu s vysledkami dalsich\n"
      << "operacii s realnymi a komplexnymi maticami.";
   GetNamesOfIOFiles("MATICE.TXT", nameoffile);

   in.open(nameoffile, ios::in);
   in >> m >> n;
   if (m != n) {
      cout << "\n  Prva nacitana matica nie je stvorcova a neda sa umocnit!";
      xx.my_getch();
      exit(1);
   }
   TMaticaStvorcova* A = new TMaticaStvorcova(n), * B = new TMaticaStvorcova(n);
```

```
54     in >> *A;
55
56     in >> m >> n;
57     if (m != n) {
58        cout << "\n  Druha nacitana matica nie je stvorcova a neda sa umocnit!";
59        xx.my_getch();
60        exit(1);
61     }
62     TMaticaStvorcovaKomplexna* KA = new TMaticaStvorcovaKomplexna(n),
63        * KB = new TMaticaStvorcovaKomplexna(n);
64     in >> *KA;
65     in.close();
66
67     cout << "\n  Z textoveho suboru boli nacitane matice A a KA:";
68     cout << *A << *KA;
69     cout << "\n  Zadajte prirodzene cislo ako exponent matice A!\n    Exponent=";
70     do
71        cin >> Exponent;
72     while (Exponent < 0);
73
74     *B = A->UmocniMaticu(Exponent);  // Try to overload the function pow(A, Exponent)
75                                      // according to our function UmocniMaticu(Exponent)!!!
76     cout << endl << Exponent;
77     switch (Exponent % 10) {
78     case 1: cout << "-va mocnina matice A je:"; break;
79     case 2: cout << "-ha mocnina matice A je:"; break;
80     case 3: cout << "-tia mocnina matice A je:"; break;
81     default: cout << "-ta mocnina matice A je:"; break;
82     }
83     cout << *B;
84
85     *B = *A + *A;
86     cout << "\nMatica B = A + A je:" << *B;
87
88     *B = *A * *A;
89     cout << "\nMatica B = A * A je:" << *B;
90
91     *B = *A;
92     *B += *A;
93     cout << "\nMatica A += A je:" << *B;
94
95     *B = *A;
96     *B *= *A;
97     cout << "\nMatica A *= A je:" << *B;
98
99     *KB = *KA + *KA;
100    cout << "\nMatica KB = KA + KA je:" << *KB;
101
102    *KB = *KA * *KA;
103    cout << "\nMatica KB = KA * KA je:" << *KB;
104
105    *KB = *KA;
106    *KB += *KA;
```

```
107     cout << "\nMatica KA += KA je:" << *KB;
108
109     *KB = *KA;
110     *KB *= *KA;
111     cout << "\nMatica KA *= KA je:" << *KB;
112
113     delete KB;
114     delete KA;
115     delete B;
116     delete A;
117
118     xx.my_getch();
119     return 0;
120  }
121  //------------------------------------------------------------------------
122  void GetNamesOfIOFiles(const char name_of_input_file[], char path_to_input_file[])
123  {
124     char current_path[MAXLINE];
125     my_class xx;
126     current_path[0] = '\0';
127
128  #ifdef _WIN32
129     TCHAR exePath[MAXLINE];
130
131     HMODULE hModule = GetModuleHandle(NULL);
132     if (hModule != NULL) {
133        if (!GetModuleFileName(hModule, exePath, MAXLINE)) {
134           cout << "Nepodarila sa zistit cesta k exe-suboru.\n";
135           xx.my_getch();
136           exit(1);
137        }
138     }
139     else {
140        cout << "Module handle is NULL.\n" << endl;
141        xx.my_getch();
142        exit(1);
143     }
144
145     int iii;
146     bool flag = false;
147     for (iii = (int)wcslen(exePath); iii >= 0; iii--) {
148        if (!flag && exePath[iii] == '\\') {
149           current_path[iii + 1] = '\0';
150           flag = true;
151        }
152        if (flag)
153           current_path[iii] = (char)exePath[iii];
154     }
155  #elif (defined __linux__)
156     unsigned iii;
157     char line[MAXLINE];
158     FILE* fp;
159     if ((fp = popen("/bin/pwd", "r")) == NULL) {
```

```
160         perror("popen error");
161         exit(1);
162      }
163      if (fgets(line, MAXLINE, fp) == NULL) {
164         perror("fgets error");
165         exit(1);
166      }
167      pclose(fp);
168
169      iii = 0;
170      while (line[iii] != '\r' && line[iii] != '\n') {
171         current_path[iii] = line[iii];
172         iii++;
173      }
174      current_path[iii] = '\0';
175   #elif (defined _AIX)
176      unsigned iii;
177      char line[MAXLINE];
178      FILE* fp;
179      if ((fp = popen("user/bin/pwd", "r")) == NULL) {
180         perror("popen error");
181         exit(1);
182      }
183      if (fgets(line, MAXLINE, fp) == NULL) {
184         perror("fgets error");
185         exit(1);
186      }
187      pclose(fp);
188
189      iii = 0;
190      while (line[iii] != '\r' && line[iii] != '\n') {
191         current_path[iii] = line[iii];
192         iii++;
193      }
194      current_path[iii] = '\0';
195   #endif
196
197      path_to_input_file[0] = '\0';
198      strcat(path_to_input_file, current_path);
199   #if (defined __linux__) || (defined _AIX)
200      strcat(path_to_input_file, "/inputs/");
201   #elif (defined _WIN32)
202      strcat(path_to_input_file, "inputs\\");
203   #endif
204      strcat(path_to_input_file, name_of_input_file);
205   }
206   //---------------------------------------------------------------------------
207
```