```cpp
// inheritance_multiple.cpp
//
#ifdef _WIN32
    #pragma warning(disable:4996)
    #include <tchar.h>
    #include <windows.h>
    #include <conio.h>
    #include <direct.h>
#elif (defined __linux__) || (defined _AIX) || (defined __APPLE__)
    #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[])
{
    TMaticaStvorcova A, B;
    TMaticaStvorcovaKomplexna KA, KB;
    my_class xx;
    ifstream in;
    char nameoffile[MAXLINE];
    int Exponent;

    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.";
    GetNamesOfIOFiles("MATICE.TXT", nameoffile);
    in.open(nameoffile, ios::in);
    in >> A >> KA;
    in.close();

    cout << "\n\nZ textoveho suboru boli nacitane matice A a KA:";
    cout << A << KA;
    cout << "\nZadajte prirodzene cislo ako exponent matice A!\nExponent=";
    do
        cin >> Exponent;
    while (Exponent < 0);

    B = A.UmocniMaticu(Exponent);  // Overload the function pow(A, Exponent)
                                   // according to our function UmocniMaticu(Exponent)!!!
    cout << endl << Exponent;
    switch (Exponent % 10) {
    case 1: cout << "-va mocnina matice A je:"; break;
    case 2: cout << "-ha mocnina matice A je:"; break;
    case 3: cout << "-tia mocnina matice A je:"; break;
    default: cout << "-ta mocnina matice A je:"; break;
    }
    cout << B;

    B = A + A;
    cout << "\nMatica B = A + A je:" << B;

    B = A * A;
    cout << "\nMatica B = A * A je:" << B;

    B = A;
```

```
74        B += A;
75        cout << "\nMatica A += A je:" << B;
76
77        B = A;
78        B *= A;
79        cout << "\nMatica A *= A je:" << B;
80
81        KB = KA + KA;
82        cout << "\nMatica KB = KA + KA je:" << KB;
83
84        KB = KA * KA;                                          // Error
85        cout << "\nMatica KB = KA * KA je:" << KB;
86
87        xx.my_getch();
88        return 0;
89    }
90    //---------------------------------------------------------------------------
91    void GetNamesOfIOFiles(const char name_of_input_file[], char path_to_input_file[])
92    {
93        char current_path[MAXLINE];
94        my_class xx;
95        current_path[0] = '\0';
96
97    #ifdef _WIN32
98        TCHAR exePath[MAXLINE];
99
100       HMODULE hModule = GetModuleHandle(NULL);
101       if (hModule != NULL) {
102           if (!GetModuleFileName(hModule, exePath, MAXLINE)) {
103               cout << "Nepodarila sa zistit cesta k exe-suboru.\n";
104               xx.my_getch();
105               exit(1);
106           }
107       }
108       else {
109           cout << "Module handle is NULL.\n" << endl;
110           xx.my_getch();
111           exit(1);
112       }
113
114       int iii;
115       bool flag = false;
116       for (iii = (int)wcslen(exePath); iii >= 0; iii--) {
117           if (!flag && exePath[iii] == '\\') {
118               current_path[iii + 1] = '\0';
119               flag = true;
120           }
121           if (flag)
122               current_path[iii] = (char)exePath[iii];
123       }
124   #elif (defined __linux__) || (defined __APPLE__)
125       unsigned iii;
126       char line[MAXLINE];
127       FILE* fp;
128       if ((fp = popen("/bin/pwd", "r")) == NULL) {
129           perror("popen error");
130           exit(1);
131       }
132       if (fgets(line, MAXLINE, fp) == NULL) {
133           perror("fgets error");
134           exit(1);
135       }
136       pclose(fp);
137
138       iii = 0;
139       while (line[iii] != '\r' && line[iii] != '\n') {
140           current_path[iii] = line[iii];
141           iii++;
142       }
143       current_path[iii] = '\0';
144   #elif (defined _AIX)
145       unsigned iii;
146       char line[MAXLINE];
```

```
147         FILE* fp;
148         if ((fp = popen("user/bin/pwd", "r")) == NULL) {
149             perror("popen error");
150             exit(1);
151         }
152         if (fgets(line, MAXLINE, fp) == NULL) {
153             perror("fgets error");
154             exit(1);
155         }
156         pclose(fp);
157 
158         iii = 0;
159         while (line[iii] != '\r' && line[iii] != '\n') {
160             current_path[iii] = line[iii];
161             iii++;
162         }
163         current_path[iii] = '\0';
164     #endif
165 
166         path_to_input_file[0] = '\0';
167         strcat(path_to_input_file, current_path);
168     #if (defined __linux__) || (defined _AIX) || (defined __APPLE__)
169         strcat(path_to_input_file, "/inputs/");
170     #elif (defined _WIN32)
171         strcat(path_to_input_file, "inputs\\");
172     #endif
173         strcat(path_to_input_file, name_of_input_file);
174     }
175     //---------------------------------------------------------------------------
176 
```