```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 <iostream>
#include <iomanip>
using namespace std;

#include "Matica.h"
//---------------------------------------------------------------------------
//Konstruktor inicializuje prvky matice
TMatica::TMatica()
{
   PocetRiadkov = MAXPOCET;
   PocetStlpcov = MAXPOCET;
   for (unsigned i = 0; i<PocetRiadkov; i++)
      for (unsigned j = 0; j<PocetStlpcov; j++)
         Matica[i][j] = 0;
}
//---------------------------------------------------------------------------
TMatica::~TMatica()
{
}
//---------------------------------------------------------------------------
const TMatica operator+(const TMatica& LavaMatica, const TMatica& PravaMatica)
{
   TMatica VyslMatica;

   VyslMatica.PocetRiadkov = LavaMatica.PocetRiadkov;
   VyslMatica.PocetStlpcov = LavaMatica.PocetStlpcov;
   for (unsigned i = 0; i<LavaMatica.PocetRiadkov; i++)
      for (unsigned j = 0; j<LavaMatica.PocetStlpcov; j++)
         VyslMatica.Matica[i][j] = LavaMatica.Matica[i][j]
         + PravaMatica.Matica[i][j];
   return VyslMatica;
}
//---------------------------------------------------------------------------
const TMatica operator*(const TMatica& LavaMatica, const TMatica& PravaMatica)
{
   TMatica VyslMatica;
```

```
54     my_class xx;
55     unsigned i, j, k;
56     float Suma;
57
58     if (LavaMatica.PocetStlpcov != PravaMatica.PocetRiadkov) {
59        cout << "\n  Matice sa nedaju nasobit!";
60        xx.my_getch();
61     }
62     VyslMatica.PocetRiadkov = LavaMatica.PocetRiadkov;
63     VyslMatica.PocetStlpcov = PravaMatica.PocetStlpcov;
64     for (i = 0; i<LavaMatica.PocetRiadkov; i++)
65        for (j = 0; j<PravaMatica.PocetStlpcov; j++) {
66           Suma = 0;
67           for (k = 0; k<LavaMatica.PocetStlpcov; k++)
68              Suma += LavaMatica.Matica[i][k] * PravaMatica.Matica[k][j];
69           VyslMatica.Matica[i][j] = Suma;
70        }
71     return VyslMatica;
72  }
73  //---------------------------------------------------------------------------
74  TMatica TMatica::UmocniMaticu(unsigned k)
75  {
76     TMatica Y, PomMatica0, PomMatica;
77     my_class xx;
78
79     if (PocetStlpcov != PocetRiadkov) {
80        cout << "\n  Matica sa neda umocnit!";
81        xx.my_getch();
82        exit(1);
83     }
84     Y.PocetRiadkov = Y.PocetStlpcov = PocetRiadkov;
85
86     switch (k) {
87     case 0:
88        for (unsigned i = 0; i<PocetRiadkov; i++) {
89           for (unsigned j = 0; j<PocetRiadkov; j++)
90              Y.Matica[i][j] = 0;
91           Y.Matica[i][i] = 1;
92        }
93        break;
94     case 1:
95        Y = *this;
96        break;
97     default:
98        PomMatica0 = PomMatica = *this;
99        k--;
100       while (k--) {
101          Y = PomMatica * PomMatica0;
102          PomMatica = Y;
103       }
104    }
105
106    return Y;
```

```
107 }
108 //---------------------------------------------------------------------------
109 istream& operator>>(istream& is, TMatica& X)
110 {
111    my_class xx;
112
113    is >> X.PocetRiadkov >> X.PocetStlpcov;
114    if (is.fail()) {
115       cout << "\n\n  Vstupny subor sa nepodarilo otvorit!\n";
116       xx.my_getch();
117       exit(1);
118    }
119    for (unsigned i = 0; i<X.PocetRiadkov; i++)
120       for (unsigned j = 0; j<X.PocetStlpcov; j++)
121          is >> X.Matica[i][j];
122    return is;
123 }
124 //---------------------------------------------------------------------------
125 ostream& operator<<(ostream& os, TMatica& X)
126 {
127    os.setf(ios::fixed, ios::floatfield);
128    os.precision(2);
129    for (unsigned i = 0; i<X.PocetRiadkov; i++) {
130       os << "\n  ";
131       for (unsigned j = 0; j<X.PocetStlpcov; j++)
132          os << setw(8) << X.Matica[i][j];
133    }
134    os << "\n";
135    return os;
136 }
137 //---------------------------------------------------------------------------
138 void my_class::my_getch() const
139 {
140 #ifdef _WIN32
141    _getch();
142 #else
143    cout << endl;
144 #endif
145 }
146 //---------------------------------------------------------------------------
147
```