

```
1 #ifndef _WIN32
2     #pragma warning(disable:4996)
3     #include <conio.h>
4     #include <direct.h>
5 #elif (defined __linux__ || (defined _AIX)
6     #include <stdlib.h>
7     #include <sys/types.h>
8     #include <sys/stat.h>
9     #include <unistd.h>
10    typedef char _TCHAR;
11    #define _tmain main
12 #endif
13
14 #include <stdio.h>
15 #include <iostream>
16 #include <iomanip>
17 using namespace std;
18
19 #include "MaticaObdlznikova.h"
20 //-----
21 // This is a default constructor to initiate the base class 'TMaticaObdlznikova'
22 // for childs
23 TMaticaObdlznikova::TMaticaObdlznikova()
24 {
25     Matica = NULL; // dolezite
26 }
27 //-----
28 //Konstruktor dynamicky alokuje pamat pre prvky matice a inicializuje ich
29 TMaticaObdlznikova::TMaticaObdlznikova(const unsigned m, const unsigned n)
30 {
31     unsigned i,j;
32
33     PocetRiadkov = m;
34     PocetStlpcov = n;
35
36     Matica = new long double*[PocetRiadkov]; // pocet [] a * je 3
37     for (i=0;i<PocetRiadkov;i++)
38         Matica[i] = new long double[PocetStlpcov]; // pocet [] je 3
39
40     for (i=0; i<PocetRiadkov; i++)
41         for (j=0; j<PocetStlpcov; j++)
42             Matica[i][j]=0;
43 }
44 //-----
45 //Kopirovaci konstruktor
46 TMaticaObdlznikova::TMaticaObdlznikova(TMaticaObdlznikova& X)
47 {
48     unsigned i,j;
49
50     PocetRiadkov = X.PocetRiadkov;
51     PocetStlpcov = X.PocetStlpcov;
52
53     Matica = new long double*[PocetRiadkov]; // pocet [] a * je 3
```

```
54     for (i=0;i<PocetRiadkov;i++)
55         Matica[i] = new long double[PocetStlpcov];    // pocet [] je 3
56
57     for (i=0;i<PocetRiadkov;i++)
58         for (j=0;j<PocetStlpcov;j++)
59             Matica[i][j] = X.Matica[i][j];
60 }
61 //-----
62 //Destruktor upratuje alokovanu pamat
63 TMaticaObdlznikova::~TMaticaObdlznikova()
64 {
65     if (Matica != NULL) {
66         for (unsigned i=0;i<PocetRiadkov;i++)
67             delete[] Matica[i];
68         delete[] Matica;
69         Matica = NULL;
70     }
71 }
72 //-----
73 TMaticaObdlznikova& TMaticaObdlznikova::operator=(const TMaticaObdlznikova& X)
74 {
75     if (this == &X) return *this;
76
77     for (unsigned i=0;i<PocetRiadkov;i++)
78         for (unsigned j=0;j<PocetStlpcov;j++)
79             Matica[i][j] = X.Matica[i][j];
80
81     return *this;
82 }
83 //-----
84 const TMaticaObdlznikova operator+(const TMaticaObdlznikova& LavaMatica,
85     const TMaticaObdlznikova& PravaMatica)
86 {
87     TMaticaObdlznikova VyslMatica(LavaMatica.PocetRiadkov, LavaMatica.PocetStlpcov);
88
89     for (unsigned i=0; i<LavaMatica.PocetRiadkov; i++)
90         for (unsigned j=0; j<LavaMatica.PocetStlpcov; j++)
91             VyslMatica.Matica[i][j] = LavaMatica.Matica[i][j]+PravaMatica.Matica[i][j];
92
93     return VyslMatica;
94 }
95 //-----
96 const TMaticaObdlznikova operator*(const TMaticaObdlznikova& LavaMatica,
97     const TMaticaObdlznikova& PravaMatica)
98 {
99     TMaticaObdlznikova VyslMatica(LavaMatica.PocetRiadkov, LavaMatica.PocetStlpcov);
100     my_class xx;
101
102     unsigned i,j,k;
103     long double Suma;
104
105     if (LavaMatica.PocetStlpcov!=PravaMatica.PocetRiadkov){
106         cout << "\n Matice sa nedaju nasobit!\n";
```

```
107     xx.my_getch();
108 }
109 for (i=0; i<LavaMatica.PocetRiadkov; i++)
110     for (j=0; j<PravaMatica.PocetStlpcov; j++){
111         Suma=0;
112         for (k=0; k<LavaMatica.PocetStlpcov; k++)
113             Suma+=LavaMatica.Matica[i][k]*PravaMatica.Matica[k][j];
114         VyslMatica.Matica[i][j]=Suma;
115     }
116     return VyslMatica;
117 }
118 //-----
119 TMaticaObdlznikova& TMaticaObdlznikova::operator+=(const TMaticaObdlznikova&
120     PravaMatica)
121 {
122     *this = *this + PravaMatica;
123     return *this;
124 }
125 //-----
126 TMaticaObdlznikova& TMaticaObdlznikova::operator*=(const TMaticaObdlznikova&
127     PravaMatica)
128 {
129     *this = *this * PravaMatica;
130     return *this;
131 }
132 //-----
133 istream& operator>>(istream& is, TMaticaObdlznikova& X)
134 {
135     my_class xx;
136
137     for (unsigned i=0; i<X.PocetRiadkov; i++)
138         for (unsigned j=0; j<X.PocetStlpcov; j++)
139             is >> X.Matica[i][j];
140     if (is.fail()){
141         cout << "\n\n Vstupny subor sa nepodarilo otvorit.\n";
142         xx.my_getch();
143         exit(1);
144     }
145
146     return is;
147 }
148 //-----
149 ostream& operator<<(ostream& os, const TMaticaObdlznikova& X)
150 {
151     os.setf(ios::fixed, ios::floatfield);
152     os.precision(2);
153     for (unsigned i=0; i<X.PocetRiadkov; i++) {
154         os << "\n ";
155         for (unsigned j=0; j<X.PocetStlpcov; j++)
156             os << setw(8) << X.Matica[i][j];
157     }
158     os << "\n";
159 }
```

```
160     return os;
161 }
162 //-----
163 void my_class::my_getch() const
164 {
165     #ifdef _WIN32
166         _getch();
167     #else
168         cout << endl;
169     #endif
170 }
171 //-----
172
173
```