```cpp
// dynamic_alocation_of_memory_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 <iostream>
using namespace std;

#define MAXLINE 255
//#define MAXLEN 20    Nie je treba!

void my_getch();
void VynasobMatice(const unsigned, const unsigned, const unsigned, long double**,
   long double**, long double**);
void VypisMaticu(const unsigned, const unsigned, long double**);
void GetNamesOfIOFiles(const char[], char[]);
//---------------------------------------------------------------------------
int _tmain(int argc, _TCHAR* argv[])
{
   int i, j, m, n, r, s;
   long double** A, ** B, ** C;
   FILE* in;
   char nameoffile[MAXLINE];

   cout << "\n  Program nacita z textoveho suboru dve matice A a B"
      << " a maticu C = A * B vypise\nna obrazovku.\n\n";

   GetNamesOfIOFiles("MATICE.TXT", nameoffile);
   if ((in = fopen(nameoffile, "r")) == NULL) {
      cout << "  Subor MATICE.TXT sa nepodarilo otvorit.\n\n";
      my_getch();
      return 1;
   }

   fscanf(in, "%i%i", &m, &n);         // mnemotechnicka pomocka
   A = new long double* [m];          // pocet [] a * je 2
   for (i = 0; i < m; i++)
      A[i] = new long double[n];      // pocet [] je 2

```

```
54     for (i = 0; i < m; i++)
55        for (j = 0; j < n; j++)
56           fscanf(in, "%Lf", &A[i][j]);
57
58     fscanf(in, "%i%i", &r, &s);
59     B = new long double* [r];
60     for (i = 0; i < r; i++)
61        B[i] = new long double[s];
62
63     if (n != r) {
64        cout << "Matice sa nedaju nasobit!";
65        my_getch();
66        return 0;
67     }
68
69     for (i = 0; i < r; i++)
70        for (j = 0; j < s; j++)
71           fscanf(in, "%Lf", &B[i][j]);
72     fclose(in);
73
74     C = new long double* [m];
75     for (i = 0; i < m; i++)
76        C[i] = new long double[s];
77
78     VynasobMatice(m, n, s, A, B, C);
79
80     cout << "  Z textoveho suboru boli nacitane dve matice:\n  A:\n";
81     VypisMaticu(m, n, A);
82     cout << "\n  B:\n";
83     VypisMaticu(r, s, B);
84     cout << "\n  Vysledna matica C=A*B je:\n\n";
85     VypisMaticu(m, s, C);
86
87     for (i = 0; i < m; i++)  // Kazde new ma mat svoje delete
88        delete[] C[i];
89     delete[] C;
90
91     for (i = 0; i < r; i++)
92        delete[] B[i];
93     delete[] B;
94
95     for (i = 0; i < m; i++)
96        delete[] A[i];
97     delete[] A;
98
99     my_getch();
100    return 0;
101 }
102 //---------------------------------------------------------------------------
103 void my_getch()
104 {
105 #ifdef _WIN32
106    _getch();
```

```
107 #else
108    cout << endl;
109 #endif
110 }
111 //---------------------------------------------------------------------------
112 // Matice ako parametre funkcie volame odkazom "by dereference"
113 void VynasobMatice(const unsigned m, const unsigned n, const unsigned s,
114    long double** X, long double** Y, long double** Z)
115 {
116    long double Suma;
117 
118    for (unsigned i = 0; i < m; i++)
119       for (unsigned j = 0; j < s; j++) {
120          Suma = 0;
121          for (unsigned k = 0; k < n; k++)
122             Suma += X[i][k] * Y[k][j];
123          Z[i][j] = Suma;
124       }
125 }
126 //---------------------------------------------------------------------------
127 // Maticu ako parameter funkcie volame odkazom "by dereference"
128 void VypisMaticu(const unsigned m, const unsigned n, long double** X)
129 {
130    for (unsigned i = 0; i < m; i++) {
131       for (unsigned j = 0; j < n; j++)
132          printf("%12.6Lf", X[i][j]);
133       cout << "\n";
134    }
135 }
136 //---------------------------------------------------------------------------
137 void GetNamesOfIOFiles(const char name_of_input_file[], char path_to_input_file[])
138 {
139    char current_path[MAXLINE];
140 
141 #ifdef _WIN32
142    TCHAR exePath[MAXLINE];
143 
144    HMODULE hModule = GetModuleHandle(NULL);
145    if (hModule != NULL) {
146       if (!GetModuleFileName(hModule, exePath, MAXLINE)) {
147          cout << "Nepodarila sa zistit cesta k exe-suboru.\n";
148          my_getch();
149          exit(1);
150       }
151    }
152    else {
153       cout << "Module handle is NULL.\n" << endl;
154       my_getch();
155       exit(1);
156    }
157 
158    int iii;
159    bool flag = false;
```

```
160     for (iii = (int)wcslen(exePath); iii >= 0; iii--) {
161        if (!flag && exePath[iii] == '\\') {
162           current_path[iii + 1] = '\0';
163           flag = true;
164        }
165        if (flag)
166           current_path[iii] = (char)exePath[iii];
167     }
168  #elif (defined __linux__)
169     unsigned iii;
170     char line[MAXLINE];
171     FILE* fp;
172     if ((fp = popen("/bin/pwd", "r")) == NULL) {
173        perror("popen error");
174        exit(1);
175     }
176     if (fgets(line, MAXLINE, fp) == NULL) {
177        perror("fgets error");
178        exit(1);
179     }
180     pclose(fp);
181
182     iii = 0;
183     while (line[iii] != '\r' && line[iii] != '\n') {
184        current_path[iii] = line[iii];
185        iii++;
186     }
187     current_path[iii] = '\0';
188  #elif (defined _AIX)
189     unsigned iii;
190     char line[MAXLINE];
191     FILE* fp;
192     if ((fp = popen("user/bin/pwd", "r")) == NULL) {
193        perror("popen error");
194        exit(1);
195     }
196     if (fgets(line, MAXLINE, fp) == NULL) {
197        perror("fgets error");
198        exit(1);
199     }
200     pclose(fp);
201
202     iii = 0;
203     while (line[iii] != '\r' && line[iii] != '\n') {
204        current_path[iii] = line[iii];
205        iii++;
206     }
207     current_path[iii] = '\0';
208  #endif
209
210     path_to_input_file[0] = '\0';
211     strcat(path_to_input_file, current_path);
212  #if (defined __linux__) || (defined _AIX)
```

```
213     strcat(path_to_input_file, "/inputs/");
214 #elif (defined _WIN32)
215     strcat(path_to_input_file, "inputs\\");
216 #endif
217     strcat(path_to_input_file, name_of_input_file);
218 }
219 //---------------------------------------------------------------------------
220
```