

```
1 // process_text_ansi.cpp
2 //
3 #ifdef _WIN32
4     #pragma warning(disable:4996)
5     #include <tchar.h>
6     #include <windows.h>
7     #include <conio.h>
8     #include <direct.h>
9 #elif (defined __linux__) || (defined _AIX)
10    #include <stdlib.h>
11    #include <sys/types.h>
12    #include <sys/stat.h>
13    #include <unistd.h>
14    typedef char _TCHAR;
15    #define _tmain main
16 #endif
17
18 #include <stdio.h>
19 #include <stdlib.h>
20 #include <string.h>
21 #include <iostream>
22 using namespace std;
23
24 #define MAXLINE 255
25
26 void my_getch();
27 void GetNamesOfIOFiles(const char[], char[]);
28 //-----
29 int _tmain(int argc, _TCHAR* argv[])
30 {
31     unsigned i, j, pocet_slov;
32     char c, Slova[50][20], temp[20];
33     FILE* in;
34     char nameoffile[MAXLINE];
35
36     cout << "\n Zo suboru Veta.txt nacistajte vetu ukoncenu bodkou, v ktorej su"
37          << " slova\noddelene jednou medzerou bez interpunkcie. Slova vety vypiste"
38          << " pod seba\nusporiadane podla dlzky slov tak, ze slova rovnakej dlzky"
39          << " su usporiadane\npodla abecedy.";
40
41     GetNamesOfIOFiles("Veta.txt", nameoffile);
42     if ((in = fopen(nameoffile, "r")) == NULL) {
43         cout << "\n\n Subor Veta.txt sa nepodarilo otvorit!\n";
44         my_getch();
45         return 1;
46     }
47
48     do // Tento blok tu nemusel byt
49         c = (char)fgetc(in);
50     while (isspace(c));
51     ungetc(c, in); // Vraciame do vstupneho prudu 1-ve pismeno
52
53     cout << "\n\nNacistana veta je: ";
```

```
54     i = 0;
55     j = 0;
56     while ((c = (char)fgetc(in)) != EOF) {
57         cout << c;
58         if (!isspace(c) && !ispunct(c)) {
59             Slova[i][j] = c;
60             j++;
61         }
62         else {
63             Slova[i][j] = '\\0';
64             i++;
65             j = 0;
66         }
67     }
68     pocet_slov = i;
69     fclose(in);
70
71     for (i = 0; i < pocet_slov - 1; i++)
72         for (j = i + 1; j < pocet_slov; j++)
73             if (strlen(Slova[i]) > strlen(Slova[j]) ||
74                 (strlen(Slova[i]) == strlen(Slova[j]) && Slova[i] < Slova[j])) {
75                 strcpy(temp, Slova[i]);
76                 strcpy(Slova[i], Slova[j]);
77                 strcpy(Slova[j], temp);
78             }
79
80     printf("\\n\\nSlova vety podla usporiadania su:");
81     for (i = 0; i < pocet_slov; i++)
82         printf("\\n%s", Slova[i]);
83     printf("\\n");
84
85     my_getch();
86     return 0;
87 }
88 //-----
89 void my_getch()
90 {
91 #ifdef _WIN32
92     _getch();
93 #else
94     cout << endl;
95 #endif
96 }
97 //-----
98 void GetNamesOfIOFiles(const char name_of_input_file[], char path_to_input_file[])
99 {
100     char current_path[MAXLINE];
101     current_path[0] = '\\0';
102
103 #ifdef _WIN32
104     TCHAR exePath[MAXLINE];
105
106     HMODULE hModule = GetModuleHandle(NULL);
```

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

```
160     exit(1);
161 }
162 pclose(fp);
163
164     iii = 0;
165     while (line[iii] != '\\r' && line[iii] != '\\n') {
166         current_path[iii] = line[iii];
167         iii++;
168     }
169     current_path[iii] = '\\0';
170 #endif
171
172     path_to_input_file[0] = '\\0';
173     strcat(path_to_input_file, current_path);
174 #if (defined __linux__) || (defined _AIX)
175     strcat(path_to_input_file, "/inputs/");
176 #elif (defined _WIN32)
177     strcat(path_to_input_file, "inputs\\");
178 #endif
179     strcat(path_to_input_file, name_of_input_file);
180 }
181 //-----
182
```