```cpp
// process_text_ansi.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 <stdlib.h>
#include <string.h>
#include <iostream>
using namespace std;

#define MAXLINE 255

void my_getch();
void GetNamesOfIOFiles(const char[], char[]);
//---------------------------------------------------------------------------
int _tmain(int argc, _TCHAR* argv[])
{
    unsigned i, j, pocet_slov;
    char c, Slova[50][20], temp[20];
    FILE* in;
    char nameoffile[MAXLINE];

    cout << "\n  Zo suboru Veta.txt nacitajte vetu ukoncenu bodkou, v ktorej su"
        << " slova\noddelene jednou medzerou bez interpunkcie. Slova vety vypiste"
        << " pod seba\nusporiadane podla dlzky slov tak, ze slova rovnakej dlzky"
        << " su usporiadane\npodla abecedy.";

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

    do                          // Tento blok tu nemusel byt
        c = (char)fgetc(in);
    while (isspace(c));
    ungetc(c, in);              // Vraciame do vstupneho prudu 1-ve pismeno

    cout << "\n\nNacitana veta je: ";
    i = 0;
    j = 0;
    while ((c = (char)fgetc(in)) != EOF) {
        cout << c;
        if (!isspace(c) && !ispunct(c)) {
            Slova[i][j] = c;
            j++;
        }
        else {
            Slova[i][j] = '\0';
            i++;
            j = 0;
        }
    }
    pocet_slov = i;
    fclose(in);

    for (i = 0; i < pocet_slov - 1; i++)
        for (j = i + 1; j < pocet_slov; j++)
            if (strlen(Slova[i]) > strlen(Slova[j]) ||
```

```
                (strlen(Slova[i]) == strlen(Slova[j]) && Slova[i] < Slova[j])) {
                strcpy(temp, Slova[i]);
                strcpy(Slova[i], Slova[j]);
                strcpy(Slova[j], temp);
            }

    printf("\n\nSlova vety podla usporiadania su:");
    for (i = 0; i < pocet_slov; i++)
        printf("\n%s", Slova[i]);
    printf("\n");

    my_getch();
    return 0;
}
//---------------------------------------------------------------------------
void my_getch()
{
#ifdef _WIN32
    _getch();
#else
    cout << endl;
#endif
}
//---------------------------------------------------------------------------
void GetNamesOfIOFiles(const char name_of_input_file[], char path_to_input_file[])
{
    char current_path[MAXLINE];
    current_path[0] = '\0';

#ifdef _WIN32
    TCHAR exePath[MAXLINE];

    HMODULE hModule = GetModuleHandle(NULL);
    if (hModule != NULL) {
        if (!GetModuleFileName(hModule, exePath, MAXLINE)) {
            cout << "Nepodarila sa zistit cesta k exe-suboru.\n";
            my_getch();
            exit(1);
        }
    }
    else {
        cout << "Module handle is NULL.\n" << endl;
        my_getch();
        exit(1);
    }

    int iii;
    bool flag = false;
    for (iii = (int)wcslen(exePath); iii >= 0; iii--) {
        if (!flag && exePath[iii] == '\\') {
            current_path[iii + 1] = '\0';
            flag = true;
        }
        if (flag)
            current_path[iii] = (char)exePath[iii];
    }
#elif (defined __linux__) || (defined __APPLE__)
    unsigned iii;
    char line[MAXLINE];
    FILE* fp;
    if ((fp = popen("/bin/pwd", "r")) == NULL) {
        perror("popen error");
        exit(1);
    }
    if (fgets(line, MAXLINE, fp) == NULL) {
        perror("fgets error");
        exit(1);
    }
    pclose(fp);

    iii = 0;
    while (line[iii] != '\r' && line[iii] != '\n') {
        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) || (defined __APPLE__)
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 
```