# Project report "Tzaryat - text game" (c/c++)

Group composition: Kacper Suder, Marcin Rzęsista, Kamil Starobrat,

Aleksandra Słysz

## What is our game about?

The general idea was to write a text-based game that offers many different paths through a fantastic land, depending on the player's choices. The game takes place on the extraordinary continent of Tzaryatu, in the new Colony on the Homeland Islands, which has become a gathering place for the worst criminals. One of the Great Masters of the Order has proposed to make Cape Rakovitza a place where the evil will finally be tamed and defeated once and for all, and the former glory of the city restored.

You wake up from a deep sleep. As you rub your eyes, you see tall towers and guards bustling around you. You find yourself in a Penal Colony, like the worst criminal. But your fate can still be changed... Make thoughtful decisions and take action

## Division of work in our project:

Executive functions (and original creator in C language): Kacper Suder

The plot of the game:Marcin Rzęsista

C to C++ translation support: Kamil Starobrat

Creator of the documentation: Aleksandra Słysz

## How was the entire program written?

The program was originally written in the C language, without the use of programming constructs such as goto, which were added to the program in C++.

1. Used libraries.

**<sstream>**, which stands for string streams, is used in our program to perform string operations, such as formatting. Another important library is **cstdlib**, which is needed, among other things, for generating pseudo-random numbers using **rand()**. We also used **ctime** for the same purpose. The other included libraries are **windows.h** (which defines many functions specific to the Windows operating system) and **iostream** (useful for input and output). The last library is **conio.h**, which is necessary for the getchar() function.

## 2. The most important variable and function declarations.

HANDLE color is a variable that stores the console color. The text in the console is displayed in different colors depending on whether it is the narrator speaking (white), a warning message such as selecting the wrong option or the game being over (red), the player facing a choice of options (blue), or a message about saving progress (purple).

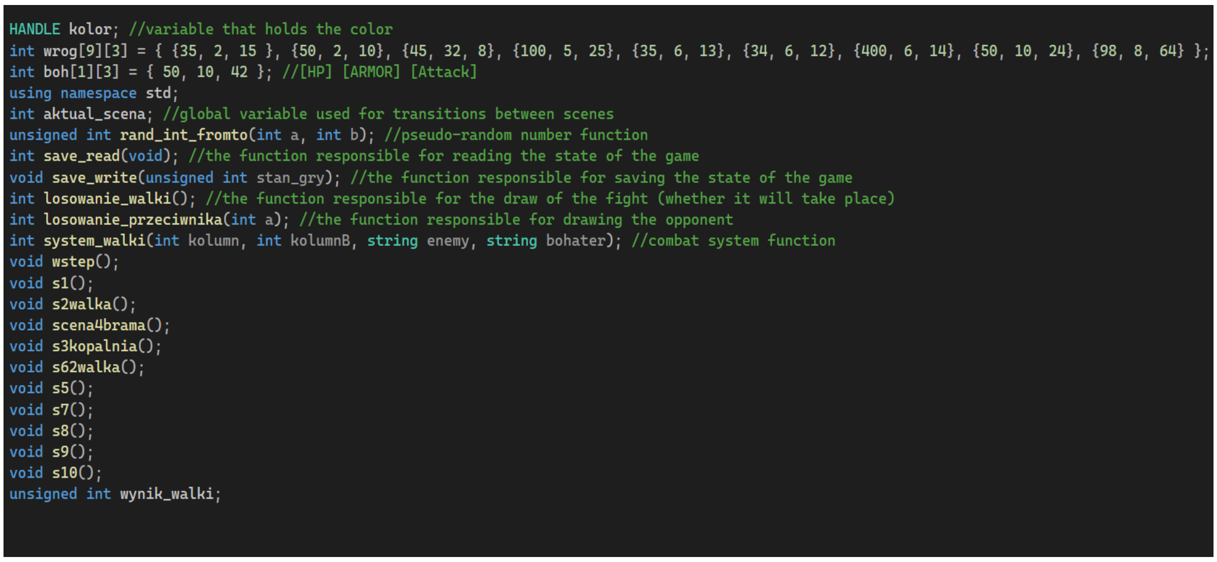
The hero's statistics and the nine possible enemies, whose parameters are HP, Armor, and Attack, are stored in a two-dimensional array.

The global variable used for overwriting, on which the current game scene that the player is moved to depends, is int current_scene. At the beginning, it takes the value 0, which represents the main menu. It then takes values from 1 to 7, representing respectively: prologue, battle, mine, gate, forest, battle2, and chapter I.

The following are the next important functions:

rand_int_fromto (int a, int b) used to draw a number between a and b

save_write (unsigned int stan_gry) to save game state



```
HANDLE kolor; //variable that holds the color
int wrog[9][3] = { {35, 2, 15 }, {50, 2, 10}, {45, 32, 8}, {100, 5, 25}, {35, 6, 13}, {34, 6, 12}, {400, 6, 14}, {50, 10, 24}, {98, 8, 64} };
int boh[1][3] = { 50, 10, 42 }; //[HP] [ARMOR] [Attack]
using namespace std;
int aktual_scena; //global variable used for transitions between scenes
unsigned int rand_int_fromto(int a, int b); //pseudo-random number function
int save_read(void); //the function responsible for reading the state of the game
void save_write(unsigned int stan_gry); //the function responsible for saving the state of the game
int losowanie_walki(); //the function responsible for the draw of the fight (whether it will take place)
int losowanie_przeciwnika(int a); //the function responsible for drawing the opponent
int system_walki(int kolumn, int kolumnB, string enemy, string bohater); //combat system function
void wstep();
void s1();
void s2walka();
void scena4brama();
void s3kopalnia();
void s62walka();
void s5();
void s7();
void s8();
void s9();
void s10();
unsigned int wynik_walki;
```

## 3. main() function.

At its beginning there is a seed for randomizing the number of srands (time(NULL)) depending on the Unix time, thanks to which the drawn number will be different each time. Then the wynik_walki() function is declared with the unsigned int type. The next step is to change the color of the text displayed in the console with:

**Kolor** = GetStdHandle (STD_OUTPUT_HANDLE)

The argument to the GetStdHandle function is a constant representing the handle. We used STD_OUTPUT_HANDLE as the console output handle; we store it in the kolor variable.

Then we call the SetConsoleTextAttribute function, which first argument is **kolor,** and the second is the number corresponding to the color.

In this way, we have simplified the way of writing this function, because we use it many times in the program.

The main function contains all the locations that the player goes through - written as a function: wstęp(), s1(), s2walka(), scena4brama(), s3kopalnia()…

4. **The function void wstęp()** – this is the main initial menu in the game ("scene zero"), which is used by the goto command.

   The main loop that is used to navigate through the game structure is the **while** loop. Its condition is aktual_scena =0. Here, the player has the following five options to choose from:

   **1.** Starting a new game.
   **2.** Continuation of the previously started game.
   **3.** Reading information about the creators.
   **4.** The end of the adventure (exit).
   **5.** Debug mode.

   We used here - as in many other places - the switch selection instruction, which in this type of game is the most optimal solution due to readability and the ability to choose from many options.
   Case 1: Contains an **if... else if... else** conditional statement, which, in the case of choosing to start the game, leads to a transition to scene number 1, while in the case of choosing to quit the game, displays the message "Since you don't want to..." and uses the goto function to take the player back to the menu. In the last case, i.e. choosing an option other than 1 or 2, the player is informed of the need to select the correct value.
   Case 2: Here, the previously started game is loaded from the last save point. It is done as follows:
   Aktual_scena =save_read()
   Where int save_read(void) is the function responsible for saving the state of the game.
   Case 3: The getchar() function retrieves a character from the player's keyboard, and then displays the text - information about the creators.
   Case 4: Once again, the if...else if statement is used. If the player decides not to give up, the game continues. If they choose to exit, they receive a message saying "You chose to run away, coward!" The last case is a message about an incorrect value and a return to the choice using the **goto** function.
   Case 5: This is the debug mode created for testing our program - it includes the function "**losowane_walki()"** (randomized battles).

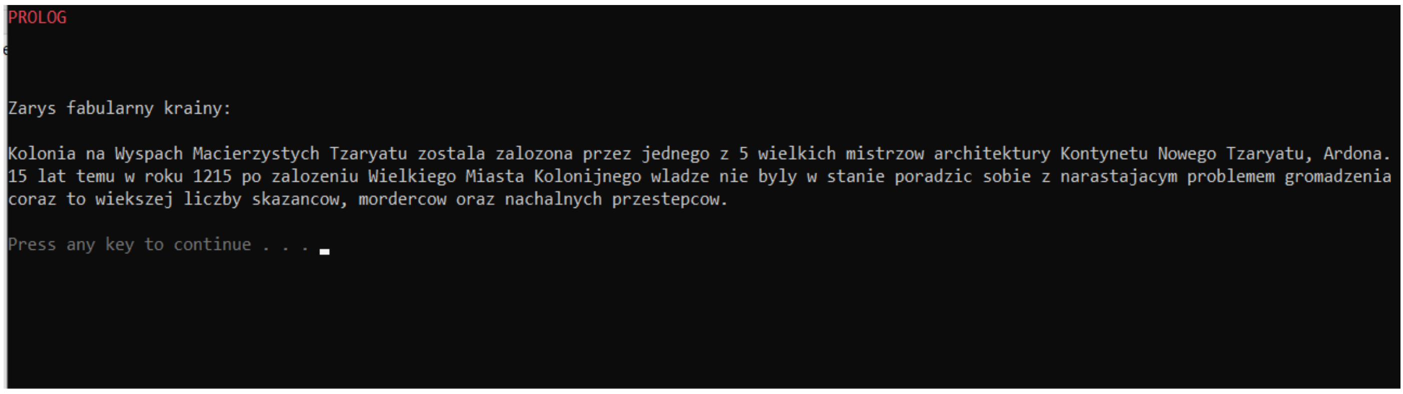
5. **void s1() – SCENE ONE**

The player reads the outline of the game world's plot, which is written in stages. After each stage, the program is momentarily paused until the player confirms with any key. We used for this:

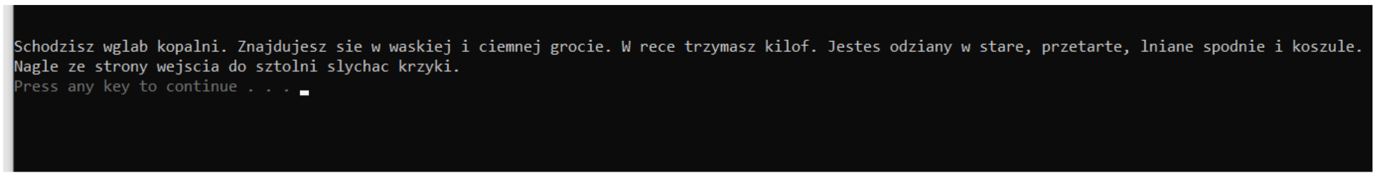
System(‘’pause”)

System(‘’cls’’)

These commands to suspend and clear the console.

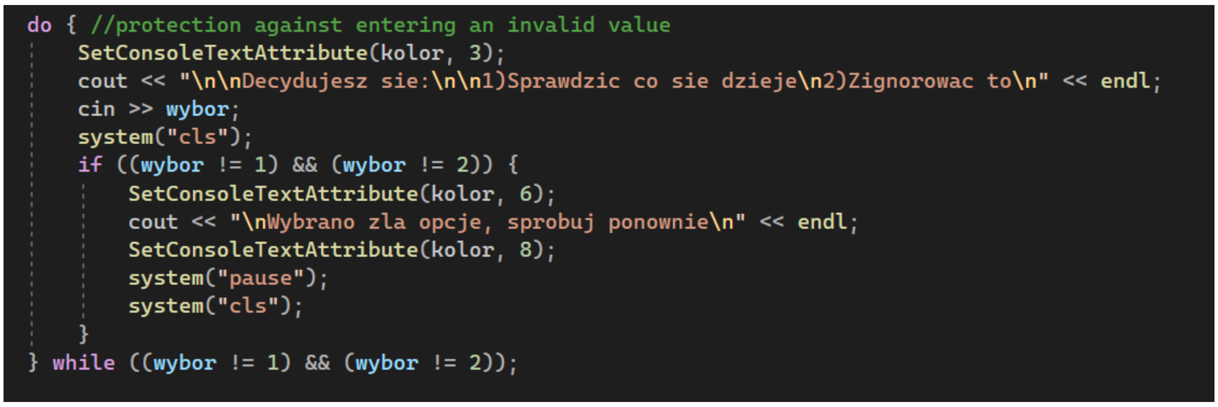


```
PROLOG


Zarys fabularny krainy:

Kolonia na Wyspach Macierzystych Tzaryatu zostala zalozona przez jednego z 5 wielkich mistrzow architektury Kontynetu Nowego Tzaryatu, Ardona.
15 lat temu w roku 1215 po zalozeniu Wielkiego Miasta Kolonijnego wladze nie byly w stanie poradzic sobie z narastajacym problemem gromadzenia
coraz to wiekszej liczby skazancow, mordercow oraz nachalnych przestepcow.

Press any key to continue . . .
```

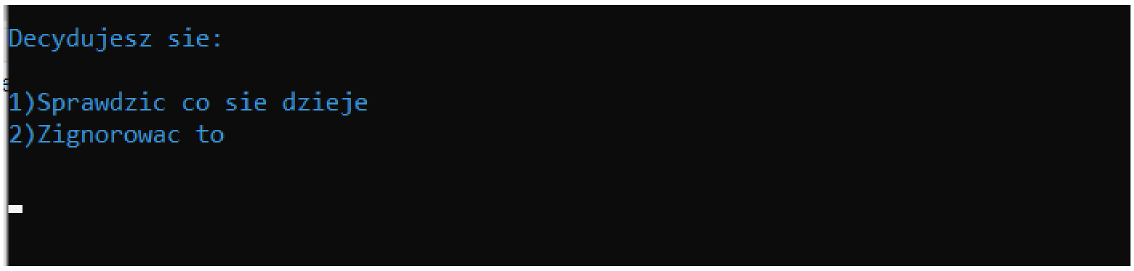


```
Schodzisz wglab kopalni. Znajdujesz sie w waskiej i ciemnej grocie. W rece trzymasz kilof. Jestes odziany w stare, przetarte, lniane spodnie i koszule.
Nagle ze strony wejscia do sztolni slychac krzyki.
Press any key to continue . . .
```

The declaration of an integer variable named "wybor" occurs. We will use it in the **do... while...** loop in the first choice the player faces. They are asked to choose option 1 or 2. As long as they do not choose one of these options, i.e. as long as the condition is met, the program loops and constantly displays a message, takes a digit from the keyboard, and checks the condition contained in the **if** statement. This solved the problem of the player entering an incorrect value, and this method was used in all moments when the player chooses an option. When they enter 1 or 2, the loop is exited and the program moves to the switch statement.



```cpp
do { //protection against entering an invalid value
    SetConsoleTextAttribute(kolor, 3);
    cout << "\n\nDecydujesz sie:\n\n1)Sprawdzic co sie dzieje\n2)Zignorowac to\n" << endl;
    cin >> wybor;
    system("cls");
    if ((wybor != 1) && (wybor != 2)) {
        SetConsoleTextAttribute(kolor, 6);
        cout << "\nWybrano zla opcje, sprobuj ponownie\n" << endl;
        SetConsoleTextAttribute(kolor, 8);
        system("pause");
        system("cls");
    }
} while ((wybor != 1) && (wybor != 2));
```



```
Decydujesz sie:

1)Sprawdzic co sie dzieje
2)Zignorowac to

▬
```

The switch statement used in this scene is nested, meaning that within case 1 there is another switch statement (after the safeguard do...while instruction). This is because the player has two paths to choose from, and each one leads to further options. In this scene, both after choosing to investigate the situation and ignoring it, the player will have to decide whether to engage in combat with an enemy or escape to the mine.

**6.** Scena walki.

It starts with a function call: void s2walka(). Here, it is checked using an if statement whether aktual_scena==2 and the game state is automatically saved to a file with Save_write(aktual_scena). The player encounters an opponent on their way and chooses which part of the body to strike. The outcome of the fight depends on a pseudorandom number generated by the function Rand_int_fromto(1,100), which selects a number from a specified range. For example, if we choose to attack the head, our strike will result in:

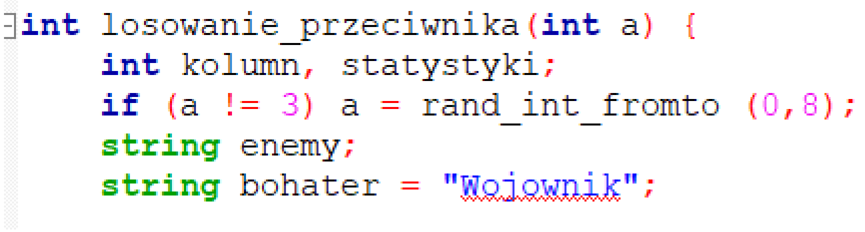
If: wynik_walki >=90 – critical hit, end of the fight

35 < Wynik_walki < 90 – successful hit, but the opponent is not finished off

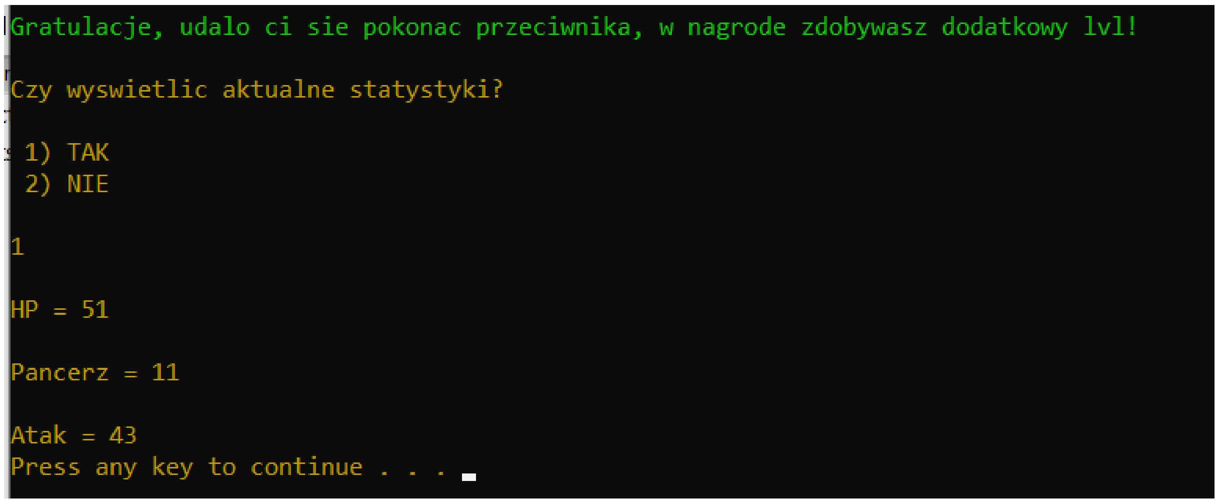
Wynik_walki < 35 - you are defeated.

These values change depending on which part of the body we choose. This is done using a switch statement.

We have also equipped our program with a battle generator. At any point, the player may encounter one of nine enemies whose stats (hp, armor, attack) are stored in an array. We have created the function " **losowanie_walki()**". The randomization process works as follows: we create an integer variable and assign it the value of 0. Then, we assign the result of "rand_int_fromto(1,100)" to the variable " **wynik** ", which is a random number generator from the specified range. If the generated number is less than 54, the " **losowanie_przeciwnika(a)”** function is called. However, if the generated number is greater than 85, there is no randomization, and the value of 3 indicates the selection of the strongest enemy, which is immediately passed to the "switch" statement. During the battle, it is also possible to check the stats of our enemy.



```
int losowanie_przeciwnika(int a) {
    int kolumn, statystyki;
    if (a != 3) a = rand_int_fromto (0,8);
    string enemy;
    string bohater = "Wojownik";
```



```
Gratulacje, udalo ci sie pokonac przeciwnika, w nagrode zdobywasz dodatkowy lvl!

Czy wyswietlic aktualne statystyki?

 1) TAK
 2) NIE

1

HP = 51

Pancerz = 11

Atak = 43
Press any key to continue . . .
```

The knowledge gained during this project, as well as the acquisition of skills in working in a 3-person group, allowed for a better understanding of the program and more conscious time planning. Moreover, it enabled the creation of more advanced programs in C++ and will bear fruit in future work.