

Collegium Witelona Uczelnia Państwowa w Legnicy

Wydział Nauk Technicznych i Ekonomicznych

Kierunek: Informatyka



Sebastian Waga nr. Indeksu 43894

Rok studiów: 1

Grupa: 2(2)

Rok akademicki: 2023-2024

Projekt z przedmiotu „Podstawy programowania”

Prosta gra w zgadywanie liczb w języku C++

Prowadzący przedmiot

mgr inż. Marcin Tracz

Legnica 2023

Spis treści

Krótki opis programu.....	3
Zrzuty ekrany pokazujące działanie programu.....	4
Zrzuty ekrany kodu i jego działanie	7

Język programu : C++

Użyty program: Visual Studio Code

Działanie programu: Program wyświetla menu, w którym dostępne są 4 opcje:

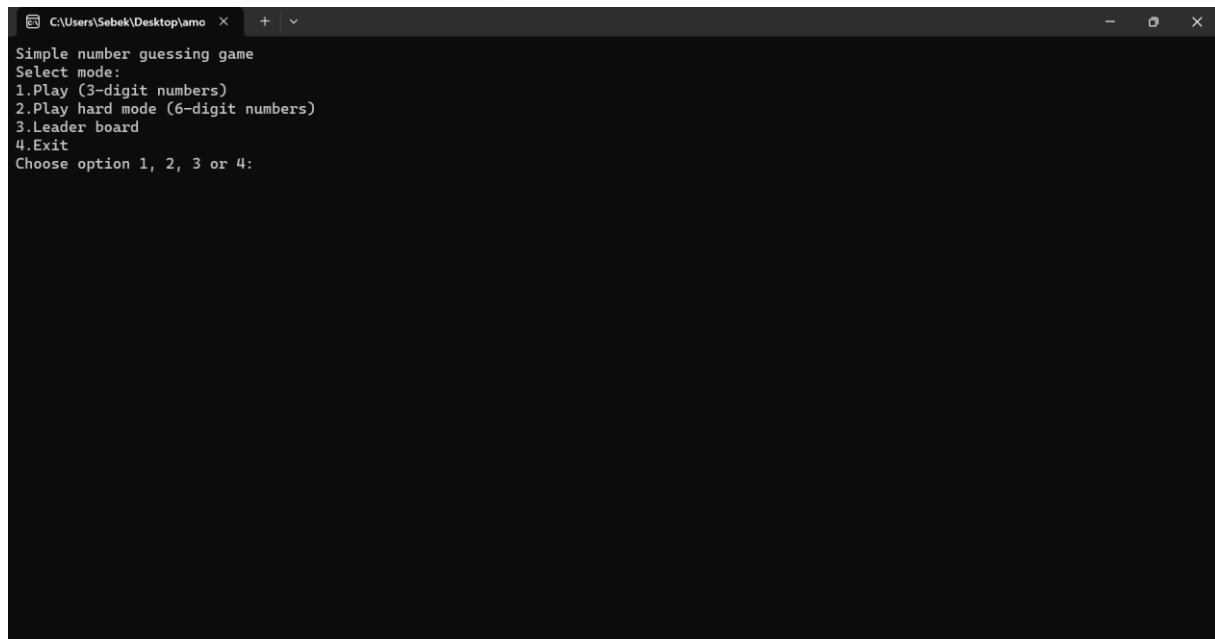
-Play (3-digit numbers) – program generuje liczbę trzycyfrową którą użytkownik musi odgadnąć wpisując trzy cyfry w konsole. Odgadnięte liczby wyświetlane są w kolorze zielonym, a nie poprawne w kolorze czerwonym. Po poprawnym odgadnięciu liczby program wyświetla gratulacje oraz prosi użytkownika o podanie nazwy użytkownika (składającej się z 5 cyfr lub liter). Po wpisaniu nazwy zapisywana jest ona razem z ilością prób oraz trybem gry w pliku „Leader_board”.

-Play (6-digit numbers) – program generuje liczbę sześciocyfrową którą użytkownik musi odgadnąć wpisując sześć cyfr w konsole. Odgadnięte liczby wyświetlane są w kolorze zielonym, a nie poprawne w kolorze czerwonym. Po poprawnym odgadnięciu liczby program wyświetla gratulacje oraz prosi użytkownika o podanie nazwy użytkownika (składającej się z 5 cyfr lub liter). Po wpisaniu nazwy zapisywana jest ona razem z ilością prób oraz trybem gry w pliku „Leader_board”.

-Leader board - wyświetla zawartość pliku „Leader_board”, jeśli istnieje i nie jest pusty.

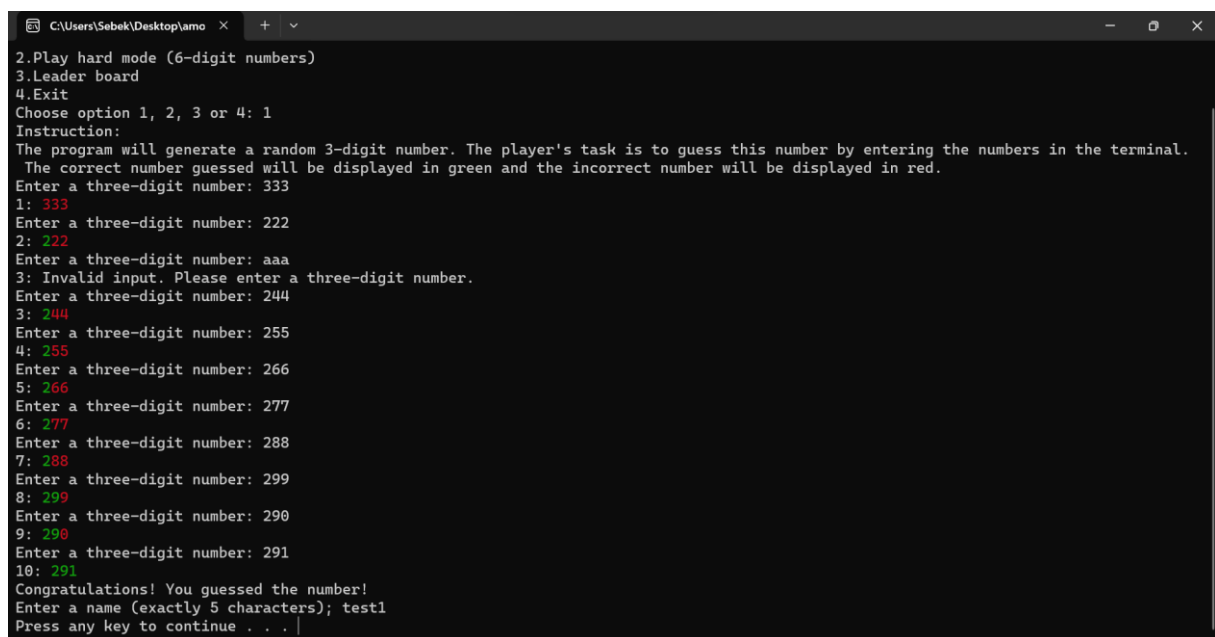
-Exit – kończy działanie programu.

Menu główne:



```
C:\Users\Sebek\Desktop\lamo x + v
Simple number guessing game
Select mode:
1.Play (3-digit numbers)
2.Play hard mode (6-digit numbers)
3.Leader board
4.Exit
Choose option 1, 2, 3 or 4:
```

Rozgrywka w trybie 3-digit numbers. Po zakończeniu usuwany jest zawartość konsoli i ponownie wyświetla się menu.



```
C:\Users\Sebek\Desktop\lamo x + v
2.Play hard mode (6-digit numbers)
3.Leader board
4.Exit
Choose option 1, 2, 3 or 4: 1
Instruction:
The program will generate a random 3-digit number. The player's task is to guess this number by entering the numbers in the terminal.
The correct number guessed will be displayed in green and the incorrect number will be displayed in red.
Enter a three-digit number: 333
1: 333
Enter a three-digit number: 222
2: 222
Enter a three-digit number: aaa
3: Invalid input. Please enter a three-digit number.
Enter a three-digit number: 244
3: 244
Enter a three-digit number: 255
4: 255
Enter a three-digit number: 266
5: 266
Enter a three-digit number: 277
6: 277
Enter a three-digit number: 288
7: 288
Enter a three-digit number: 299
8: 299
Enter a three-digit number: 290
9: 290
Enter a three-digit number: 291
10: 291
Congratulations! You guessed the number!
Enter a name (exactly 5 characters); test1
Press any key to continue . . . |
```

Rozgrywka w trybie 6-digit numbers. Po zakończeniu usuwany jest zawartość konsoli i ponownie wyświetla się menu.

```
C:\Users\Sebek\Desktop\lamo x + v
Simple number guessing game
Select mode:
1.Play (3-digit numbers)
2.Play hard mode (6-digit numbers)
3.Leader board
4.Exit
Choose option 1, 2, 3 or 4: 2
Instruction:
The program will generate a random 6-digit number. The player's task is to guess this number by entering the numbers in the terminal.
The correct number guessed will be displayed in green and the incorrect number will be displayed in red.
Enter a six-digit number: 333333
1: 333333
Enter a six-digit number: 222222
2: 222222
Enter a six-digit number: 111111
3: 111111
Enter a six-digit number: 144144
4: 144144
Enter a six-digit number: 144166
5: 144166
Congratulations! You guessed the number!
Enter a name (exactly 5 characters); test2
Press any key to continue . . . |
```

Wyświetlenie zawartości pliku „Leader_board”

```
C:\Users\Sebek\Desktop\lamo x + v
Simple number guessing game
Select mode:
1.Play (3-digit numbers)
2.Play hard mode (6-digit numbers)
3.Leader board
4.Exit
Choose option 1, 2, 3 or 4: 3
Name: test1 number of tries: 10 Mode: 3-digits
Name: test2 number of tries: 5 Mode: 6-digits
Press any key to continue . . . |
```

Powiadomienie w przypadku braku pliku lub braku zawartości pliku „Leader_board”.

```
C:\Users\Sebek\Desktop\amo x + v
Simple number guessing game
Select mode:
1.Play (3-digit numbers)
2.Play hard mode (6-digit numbers)
3.Leader board
4.Exit
Choose option 1, 2, 3 or 4: 3
File leader_board is empty or does not exist
Press any key to continue . . . |
```

Biblioteki użyte w projekcie. Rozszerzają one możliwości języka umożliwiając użycie gotowych funkcji.

```
1  #include <iostream>
2  #include <cstdlib>
3  #include <ctime>
4  #include <algorithm>
5  #include <random>
6  #include <fstream>
7  #include <limits>
8  #include <windows.h>
9  #include <string> //Libraries
```

Dyrektywa „Using namespace std” umożliwia pisanie komend pokroju „cout” bez potrzeby dopisywania „std::cout”, ponieważ pokazuje ona kompilatorowi żeby w przypadku nie znalezienia danej nazwy szukał jej w przestrzeni „std”

```
using namespace std; //Command for the compiler so that you don't have to add "std:" before each command
```

„Int main ()” rozpoczyna program. Całą grą znajduję się w pętli while (true) by umożliwić powrót do menu głównego po skończonej operacji oraz usuwanie wcześniej wpisanych komend. Poniżej dodana jest zmienna „Menunumber” której wartość odpowiada wybranej pozycji w menu. Od linijki 18 do 24 wyświetlane jest menu. Linijka 26 prosi użytkownika o wpisanie numeru menu a funkcja „switch” umożliwia wybór pozycji z menu poprzez przypisanie zmiennych „Menunumber” do różnych przypadków czyli do „case”.

```
11 using namespace std; //Command for the compiler so that you don't have to add "std:" before each command
12
13 int main () //Start of the program
14 {
15     while (true) //A loop that allows you to return to the menu after the operation is completed
16     {
17         int Menunumber;
18         cout << "Simple number guessing game"<<endl;
19         cout << "Select mode:" <<endl;
20         cout << "1.Play (3-digit numbers)" <<endl;
21         cout << "2.Play hard mode (6-digit numbers)" <<endl;
22         cout << "3.Leader board" <<endl;
23         cout << "4.Exit" <<endl;
24         cout << "Choose option 1, 2, 3 or 4: ";
25         cin >> Menunumber;
26         switch (Menunumber) //Main menu
27         {
```

„case 1:” jest to prosta gra w zgadnięcie liczby trzycyfrowej. Program najpierw wyświetla instrukcje jak grać. Następnie za pomocą „Handle hConsole = GetStdHandle(STD_OUTPUT_HANDLE)” tworzymy „uchwyt” do urządzenia wyjściowego czyli konsoli i przechowujemy go zmiennej o nazwie hConsole. Będzie to potrzebne do zmiany koloru tekstu. W linijkach od 33 do 35 program najpierw generuje losowe „ziarno” i generuje liczbę z zakresu od 100 do 999. Następnie program za pomocą "int drawn_number = distrib(gen)" generuje liczbę za pomocą generatora liczb losowych „gen” i dystrybucji „distrib”. Wynik jest przechowywany w zmiennej „drawn_number”. "string drawn_number_digit = to_string(drawn_number)" konwertuje liczbę "drawn_number" na ciąg znaków (string) za pomocą funkcji "to_string". Wynik jest przechowywany w zmiennej "drawn_number_digit". Komenda "string attempt" deklaruje pusty ciąg znaków o nazwie "attempt". A „int answer = 0” deklaruje zmienną „answer” typu „int” i inicjalizuje ją wartością 0.

```

26         switch (Menunumber)           //Main menu
27         {
28             case 1:                     //3 digit guessing game
29             {
30                 cout << "Instruction:" << endl;
31                 cout << "The program will generate a random 3-digit number. The player's task is to guess this number by entering the numbers i
32                 HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
33                 random_device rd;        // Used to obtain the seed for the random number generator
34                 mt19937 gen(rd());        // Standard random number generator mersenne_twister_engine
35                 uniform_int_distribution<> distrib(100, 999);    // Generates random numbers in the range [100, 999]
36
37                 int drawn_number = distrib(gen);
38                 string drawn_number_digit = to_string(drawn_number);
39                 string attempt;
40                 int answer = 0;

```

Całość następującego kodu zamknięta jest w pętli „do while” by gra zapętlala się do momentu poprawnie odgadniętej liczby. Następnie prosi użytkownika o podanie trzycyfrowej liczby oraz odlicza przy podaniu liczby ilość odpowiedzi których udzielił użytkownik. Następnie program sprawdza czy dopowiedz ma dokładnie trzy znaki i czy wszystkie znaki są cyframi, jeśli nie program pokazuje wiadomość o niepoprawnym zapisie. Program od liniiki 54 do 79 sprawdza każdą cyfrę, czy jest identyczna z wygenerowaną liczbą losową. Jeśli tak liczba wyświetlona jest w kolorze zielonym, jeśli nie to cyfra wyświetlona jest w kolorze czerwonym.

```

41
42         do                               //A loop that waits for the player to guess the name
43         {
44
45             cout << "Enter a three-digit number: ";
46             cin >> attempt;
47             cout << answer + 1 << ": ";
48
49             if (attempt.length() != 3 || !all_of(attempt.begin(), attempt.end(), ::isdigit)) {
50                 cout << "Invalid input. Please enter a three-digit number." << endl;
51                 continue;
52             }
53
54             if (drawn_number_digit[0] == attempt[0]){
55                 SetConsoleTextAttribute(hConsole, 2);
56                 cout << attempt[0];
57                 SetConsoleTextAttribute(hConsole, 7);}
58             else{
59                 SetConsoleTextAttribute(hConsole, 4);
60                 cout << attempt[0];
61                 SetConsoleTextAttribute(hConsole, 7);}

```

Gra kończy się w momencie odgadnięcia liczby przez użytkownika. Program wyświetla wtedy gratulacje oraz prosi o podanie nazwy której długość nie może przekroczyć limitu 5 znaków. Następnie program zapisuje nazwę, ilość prób zgadywania oraz tryb gry w pliku „Leader_board.txt”. W przypadku braku takiego pliku zostaje on utworzony.


```

79 | SetConsoleTextAttribute(hConsole, 7);}
80 |
81 |     cout << endl;
82 |
83 |     answer++;           //The program checks whether the user enters the correct numbers and counts the number of attempts
84 |
85 | }
86 | while (drawn_number_digit != attempt);
87 |
88 |     cout << "Congratulations! You guessed the number!" <<endl;           //Congratulations
89 |
90 |     string name;
91 |     do
92 |     {
93 |         cout << "Enter a name (exactly 5 characters); ";
94 |         cin >> name;
95 |     }
96 |     while (name.length() != 5);
97 |
98 |     ofstream Leader_board("Leader_board.txt", ios::app);
99 |     Leader_board << "Name: " << name << " number of tries: " << answer << " Mode: 3-digits"<<endl;
100 |     Leader_board.close();           //The program asks you to enter a username that has exactly 5 letters and numbers, and t
101 |
102 |     system("pause");
103 |     break;
104 | }

```

„case 2” zrobiony został w identyczny sposób, zostały tylko zmienione niektóre parametry programu.

```

105 |     case 2:
106 |     {
107 |         cout <<"Instruction:"<<endl;
108 |         cout << "The program will generate a random 6-digit number. The player's task is to guess this number by entering the numbers i
109 |         HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
110 |         random_device rd;
111 |         mt19937 gen(rd());
112 |         uniform_int_distribution<> distrib(100000, 999999);
113 |         int drawn_number = distrib(gen);
114 |         string drawn_number_digit = to_string(drawn_number);
115 |         string attempt;
116 |         int answer = 0;
117 |
118 |         //A loop that waits for the player to guess the name
119 |         {
120 |
121 |             cout << "Enter a six-digit number: ";
122 |             cin >> attempt;
123 |             cout << answer + 1 << " ": ";
124 |
125 |             if (attempt.length() != 6 || !all_of(attempt.begin(), attempt.end(), ::isdigit)) {
126 |                 cout << "Invalid input. Please enter a three-digit number." << endl;
127 |                 continue;
128 |             }
129 |             if (drawn_number_digit[0] == attempt[0]){
130 |                 SetConsoleTextAttribute(hConsole, 2);
131 |                 cout << attempt[0];
132 |                 SetConsoleTextAttribute(hConsole, 7);}

```

```

129 |             if (drawn_number_digit[0] == attempt[0]){
130 |                 SetConsoleTextAttribute(hConsole, 2);
131 |                 cout << attempt[0];
132 |                 SetConsoleTextAttribute(hConsole, 7);}
133 |             else{
134 |                 SetConsoleTextAttribute(hConsole, 4);
135 |                 cout << attempt[0];
136 |                 SetConsoleTextAttribute(hConsole, 7);}
137 |
138 |             if (drawn_number_digit[1] == attempt[1]){
139 |                 SetConsoleTextAttribute(hConsole, 2);
140 |                 cout << attempt[1];
141 |                 SetConsoleTextAttribute(hConsole, 7);}
142 |             else{
143 |                 SetConsoleTextAttribute(hConsole, 4);
144 |                 cout << attempt[1];
145 |                 SetConsoleTextAttribute(hConsole, 7);}
146 |
147 |             if (drawn_number_digit[2] == attempt[2]){
148 |                 SetConsoleTextAttribute(hConsole, 2);
149 |                 cout << attempt[2];
150 |                 SetConsoleTextAttribute(hConsole, 7);}
151 |             else{
152 |                 SetConsoleTextAttribute(hConsole, 4);
153 |                 cout << attempt[2];
154 |                 SetConsoleTextAttribute(hConsole, 7);}
155 |
156 |             if (drawn_number_digit[3] == attempt[3]){
157 |                 SetConsoleTextAttribute(hConsole, 2);

```

```

C:\Users> Sebek > Desktop > amonra > G Projekt1.cpp > main()
156     if (drawn_number_digit[3] == attempt[3]){
157         SetConsoleTextAttribute(hConsole, 2);
158         cout << attempt[3];
159         SetConsoleTextAttribute(hConsole, 7);}
160     else{
161         SetConsoleTextAttribute(hConsole, 4);
162         cout << attempt[3];
163         SetConsoleTextAttribute(hConsole, 7);}
164
165     if (drawn_number_digit[4] == attempt[4]){
166         SetConsoleTextAttribute(hConsole, 2);
167         cout << attempt[4];
168         SetConsoleTextAttribute(hConsole, 7);}
169     else{
170         SetConsoleTextAttribute(hConsole, 4);
171         cout << attempt[4];
172         SetConsoleTextAttribute(hConsole, 7);}
173
174     if (drawn_number_digit[5] == attempt[5]){
175         SetConsoleTextAttribute(hConsole, 2);
176         cout << attempt[5];
177         SetConsoleTextAttribute(hConsole, 7);}
178     else{
179         SetConsoleTextAttribute(hConsole, 4);
180         cout << attempt[5];
181         SetConsoleTextAttribute(hConsole, 7);}
182
183     cout << endl;

```

```

        answer++;
        //The program checks whether the user enters the correct numbers and counts the number of attempts
    }
    while (drawn_number_digit != attempt);

        cout << "Congratulations! You guessed the number!" <<endl;
        //Congratulations

        string name;
        do
        {
            cout << "Enter a name (exactly 5 characters); ";
            cin >> name;
        }
        while (name.length() != 5);

        ofstream Leader_board("Leader_board.txt", ios::app);
        Leader_board << "Name: " << name << " number of tries: " << answer << " Mode: 6-digits"<<endl;
        Leader_board.close();
        //The program asks you to enter a username that has exactly 5 letters and numbers, and t

        system("pause");
        break;
    }

```

„case 3” rozpoczyna się komendą "ifstream Leader_board("Leader_board.txt")" która tworzy obiekt strumienia wejściowego o nazwie "Leader_board", który otwiera plik o nazwie "Leader_board.txt". Następnie program komendą "if (!Leader_board)" sprawdza, czy plik "Leader_board.txt" został poprawnie otwarty. Jeśli nie, wyświetla komunikat, że plik jest pusty lub nie istnieje. Jeśli plik został poprawnie otwarty wykonywana jest komenda „else”. Następnie "string line; bool isEmpty = true" deklaruje pusty ciąg znaków "line" i zmienną logiczną "isEmpty", która jest ustawiona na "true". W 220 linii komenda "while (getline(Leader_board, line))" czyta plik linia po linii. Jeśli linia nie jest pusta, wyświetla ją na ekranie i ustawia "isEmpty" na "false". Jeśli po przeczytaniu całego pliku "isEmpty" nadal jest "true", oznacza to, że plik był pusty. Wtedy wyświetla komunikat, że plik jest pusty lub nie istnieje.

```

209     case 3:
210     {
211         ifstream Leader_board("Leader_board.txt");
212         if (!Leader_board)
213         {
214             cout << "File leader_board is empty or does not exist" << endl;
215         }
216         else
217         {
218             string line;
219             bool isEmpty = true;
220             while (getline(Leader_board, line))
221             {
222                 if (!line.empty())
223                 {
224                     cout << line << endl;
225                     isEmpty = false;
226                 }
227             }
228             if (isEmpty)
229             {
230                 cout << "File leader_board is empty or does not exist" << endl; //The program displays the contents of the file
231             }
232         }
233         system("pause");
234         break;
235     }

```

„case4” kończy działanie programu

```

237     case 4:
238     {
239         exit(0); //Turning off the program
240         break;
241     }

```

„default” wykonuje się gdy zostanie użyty w menu inny znak niż od 1 do 4. W takim przypadku program najpierw sprawdza za pomocą "if (cin.fail() || Menunumber < 1 || Menunumber > 4)" czy wystąpił problem podczas próby wczytania wartości do zmiennej "Menunumber" lub czy wartość "Menunumber" jest mniejsza od 1 lub większa od 4. Następnie czyści błąd dla polecenia „cin” i wyświetla napis „Wrong input”. Następnie czyści dane w terminalu.

```

243     default:
244     {
245         if (cin.fail() || Menunumber < 1 || Menunumber > 4) {
246             cin.clear(); //Clear error status
247             cin.ignore(numeric_limits<streamsize>::max(), '\n'); //Skip invalid characters
248             cout << "Wrong input" << endl;
249             system("pause");
250             system("cls");
251             continue;
252         }
253     }
254 }
255 system("cls");
256 }
257 }
258 return 0;
259 }

```

