Politechnika Śląska

Wydział Matematyki Stosowanej

Kierunek Informatyka

Gliwice, 24.01.2022

Programowanie I

**projekt zaliczeniowy**

**"*Icy Tower*"**

**Jakub Głuszek gr. lab. 7**

**1. Opis projektu.**

*Icy Tower* to gra konsolowa swoim działaniem przypominająca platformową grę komputerową o tej samej nazwie. Celem gry jest przetrwanie możliwie jak najdłużej, a gra kończy się gdy postać spadnie na dół planszy. Aby tego uniknąć, gracz musi przeskakiwać postacią przez luki w opuszczających się, i z czasem przyśpieszających przeszkodach.

**2. Wymagania**

* Bohater przemieszcza się w lewo klikając klawisz „a” oraz w prawo klikając klawisz „d”,
* W momencie kliknięcia spacji bohater podskakuje,
* W grze występuje imitacja zjawiska grawitacji, więc po podskoczeniu bohater opada z większą prędkością niż przeszkoda, dopóki na nią nie natrafi,
* Przeszkody opadają w jednakowym tempie i generują się w takich samych odstępach,
* W regularnych odstępach czasu prędkość opadania przeszkód wzrasta,
* W przeszkodach znajdują się losowo generowane otwory przez które postać może dostać się wyżej,
* Postać nie przebija się przez przeszkody, tak samo jak przez ściany i sufit,
* Na początku gry postać stoi na podłodze, jednak gdy pierwsza przeszkoda osiągnie poziom podłogi ta znika, a znalezienie się postaci na poziomie podłogi będzie równoznaczne z końcem gry,
* Na ekranie oprócz planszy wyświetla się też wynik, który jest ilością przetrwanych sekund,
* W grze występują poziomy trudności,
* Po włączeniu programu otwiera się instrukcja działania programu, menu, w którym pojawia się możliwość rozpoczęcia i zakończenia gry, a po wyborze rozpoczęcia gry możliwość ustawienia poziomu trudności,
* Po przegranej grze pojawia się komunikat o przegranej, ilość punktów związanych z czasem gry oraz komunikat o możliwości naciśnięcia dowolnego klawisza, co automatycznie przekierowuje gracza z powrotem do menu gry, w którym znów można rozpocząć rozgrywkę bądź zakończyć działanie programu.

**3. Przebieg realizacji**

*Projekt zrealizowałem w jednym pliku o rozszerzeniu .cpp.*

*W całym kodzie użyłem łącznie 6 bibliotek – iostream, ctime, math.h, conio.h, random i windows.h.*

*Biblioteka iostream została przeze mnie wykorzystana do operacji na konsoli, takich jak na przykład instrukcje wejścia i wyjścia.*

*Biblioteka ctime była mi potrzebna do wyznaczania ilości sekund jakie upłunęły od rozpoczęcia rozgrywki (funkcja time(NULL)).*

*Biblioteka math.h służyła mi do wykonywania bardziej skomplikowanych operacji matematycznych.*

*Biblioteka conio.h była używana do otrzymywania informacji jaki klawisz gracz nacisnął bez konieczności zatrzymywania gry. W tym celu użyłem funkcji \_kbhit() i \_getch().*

*Biblioteka random była potrzebna do generowania losowych liczb, aby luki w przeszkodach zawsze były generowane losowo.*

*Biblioteka windows.h została wykorzystana, aby spowolnić generowania elementów na nowym polu (w tym celu użyłem funkcji Sleep()). Używałem jej również do bardziej efektywnej metody generowania planszy.*

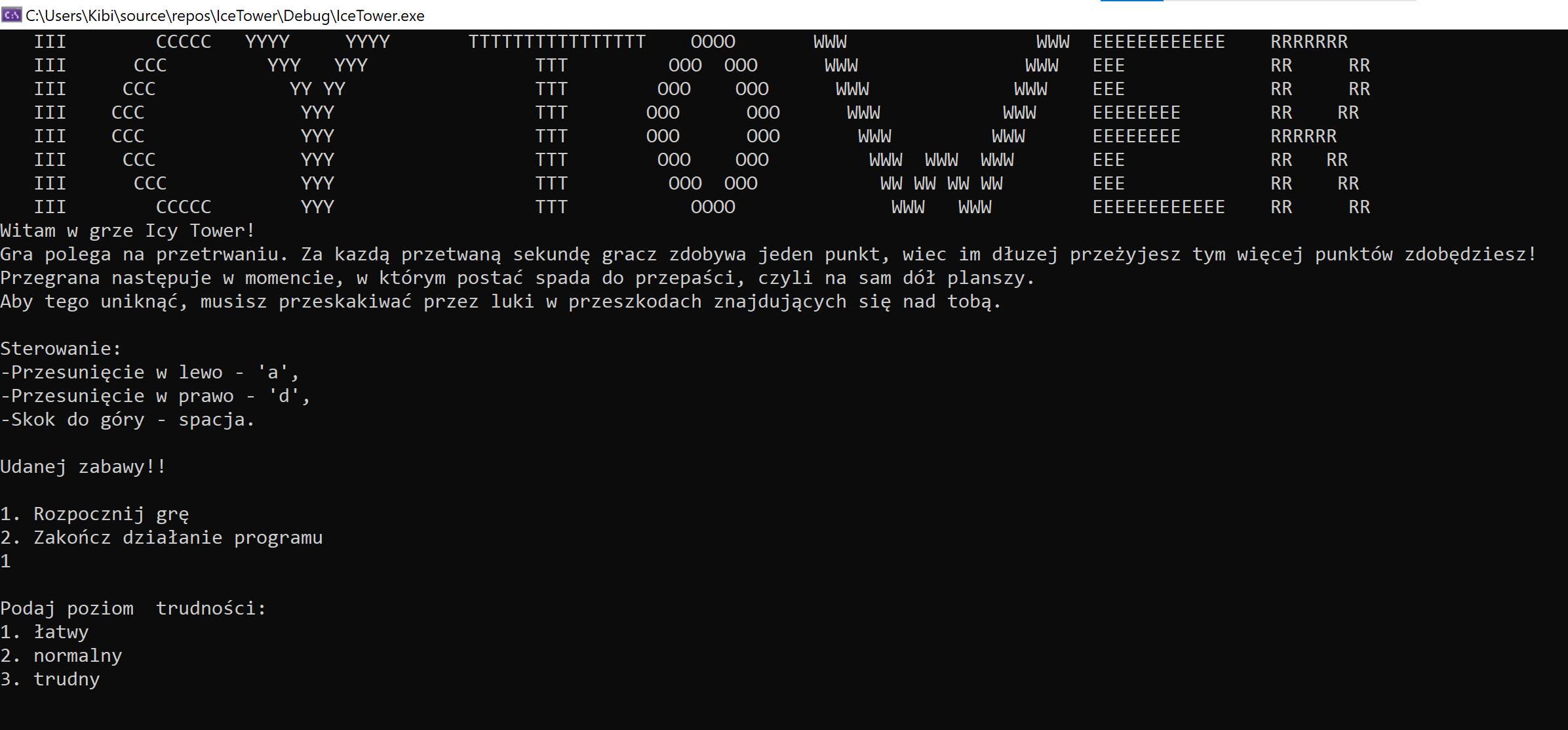
**4. Instrukcja użytkownika**

*Po włączeniu gry wyświetlony zostaje krótki opis gry i menu, z którego możemy wybrać rozpoczęcie gry lub jej zakończenie.*

*Obraz zawierający tekst, sprzęt elektroniczny

Opis wygenerowany automatycznie*

*Po wpisaniu 1 wyświetlone zostają możliwe poziomy trudności, od których zależy prędkość poruszania się przeszkód, a po wpisaniu 2 program skończy swoje działanie.*

**

*Poziomy trudności (łatwy, normalny bądź trudny) wybiera się wpisując klawisz 1, 2 albo 3. Gdy poziom trudności zostaje wybrany, wygenerowana zostaje plansza wraz z graczem i przeszkodami oraz ilość punktów w prawym górnym rogu planszy. Aby zdobyć jak największą liczbę punktów, gracz musi przeskakiwać przez luki w przeszkodach znajdujących się nad nim. Aby się przemieszczać, należy używać ‘a’ do ruchu w lewo, ‘d’ do ruchu w prawo i spacji do skoku. Spadek na sam dół planszy jest równoznaczny z przegraną.*

*Obraz zawierający tekst

Opis wygenerowany automatycznie*

*Gdy rozgrywka dobiegnie końca, pojawia się komunikat o przegranej, liczbie uzyskanych punktów oraz o możliwości naciśnięcia dowolnego klawisza.*

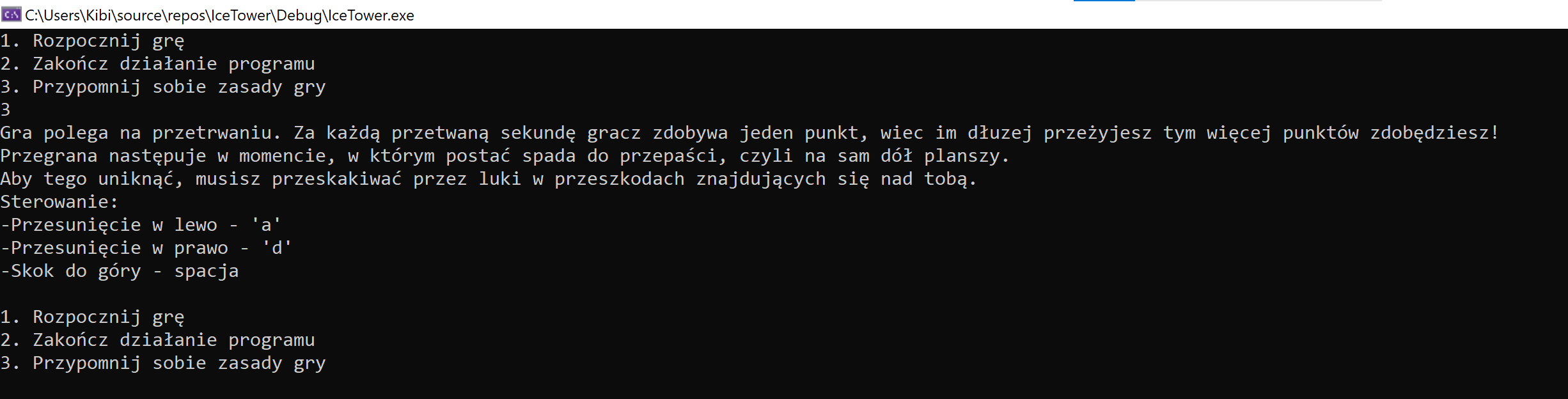
*Obraz zawierający tekst

Opis wygenerowany automatycznie*

*Po naciśnięciu dowolnego klawisza gra znowu przekierowuje użytkownika do menu, w którym teraz dodatkowo można wybrać opcję 3 – przypomnienie zasad gry.*

*Obraz zawierający tekst

Opis wygenerowany automatycznie*

**

*Wybranie opcji 1 ponownie pozwoli na rozgrywkę na wybranym poziomie trudności, a opcja 2 zakończy działanie programu.*

**5. Podsumowanie i wnioski.**

*Pomimo wielu początkowych niepowodzeń program projekt w całości spełnia swoje założenia. Wiele problemów sprawiło mi „przebijanie się” postaci przez przeszkody, problem z samym generowaniem przeszkód, a w szczególności ich znikaniem na samym dole planszy i późniejszym pojawianiem się ich na górze. Moim ostatnim dużym problemem był czas generowania całej planszy, która przez swoją wielkość i co za tym idzie – dużą ilość elementów, generowała się bardzo długo, co utrudniało dynamiczną grę. Problem rozwiązałem dzięki zamianie jedynie niektórych elementów na planszy znając ich współrzędne, co pozwoliło mi nie generować całej tablicy za każdym razem od nowa. Najdłużej zajęło mi zrozumienie problemu z „przebijaniem się” przez krawędź przeszkody w konkretnym przypadku spadania, jednak po dogłębnej analizie kodu zrozumiałem, że problem wiąże się z niepoprawną kolejnością wykonywania funkcji w kodzie. Przez pewien czas borykałem się też z problemem dotyczącym wyskakiwania błędu w przypadku wpisania złego typu znaku w menu, jednak w tym pomogła mi funkcja cin.fail() poznana na zajęciach z programowania. Uwzględniając moje początkowe założenia oraz ilość nowych rzeczy, których udało mi się nauczyć, myślę, że efekt końcowy mnie satysfakcjonuje.*