Politechnika Śląska
Wydział Matematyk Stosowanej
Kierunek Informatyka

Gliwice, 30.05.2025

# Programowanie II

# **projekt zaliczeniowy**

## "***Space Invaders project***"

**Tomasz Przebindowski gr. lab. 3**

**1. Opis projektu.**

*Projekt ma imitować mechaniki kultowej gry, wydanej w 1978 roku pod tytułem „Space Invaders". Jednakże w projekcie użyłem jedynie konsoli, nie tworząc innych interfejsów graficznych.*

**2. Funkcjonalności.**

*- Animowane menu główne,*

*- Płynne wyświetlanie przebiegu gry,*

*- Muzyka oraz dźwięki*

*- Lokalne zapisywanie statystyk gracza,*

*- Mechanika gry, w tym:*

    *- animowane obiekty (śmierć gracza, śmierć przeciwnika, uderzenie pocisku),*

    *- kolizje pocisków z graczem, przeciwnikami oraz granicami areny,*

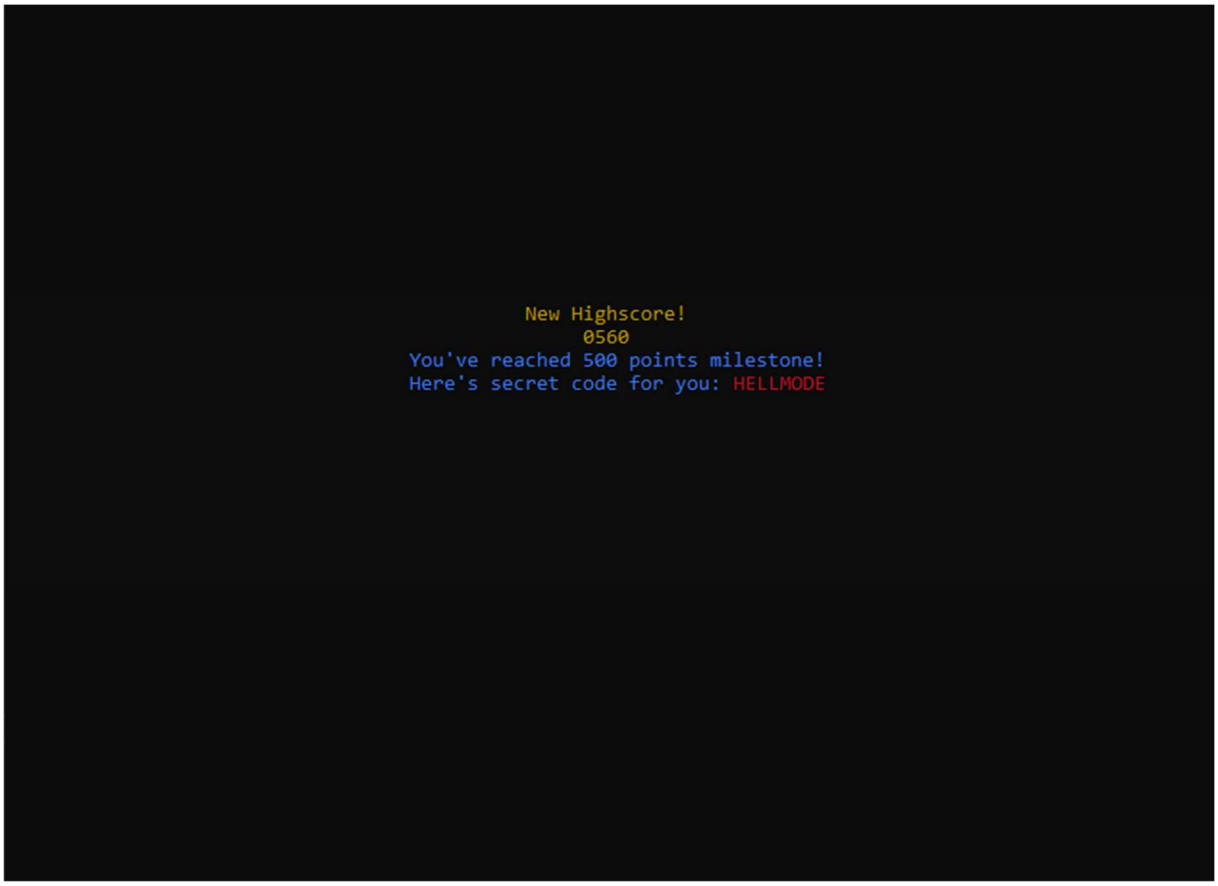
    *- poruszanie się przeciwników w stronę gracza (szybkość poruszania jest zależna od ilości przeciwników),*

    *- system żyć gracza,*

    *- nieskończona rozgrywka,*

*- Specjalny kod „HELLMODE", o którym gracz dowiaduje się po uzyskaniu co najmniej 500 punktów. Ten kod można wpisać w zakładce „CODES", znajdującej się w menu głównym, aby aktywować specjalny poziom trudności, w którym gracz ma tylko jedno życie, a przeciwnicy strzelają ze znacznie większą częstotliwością.*

*Otrzymanie kodu HELLMODE (wyświetla się przez 5 sekund, za każdym razem gdy gracz uzyska co najmniej 500 punktów w grze):*




*Napis „New Highscore!”, oraz nowy highscore (poniżej), wyświetla się tylko wtedy, kiedy gracz otrzyma więcej punktów, niż jego aktualny najwyższy wynik.*

**3. Przebieg realizacji.**

***NAJWAŻNIEJSZE WYKONANE ZADANIA W SKRÓCIE:***

- *Zablokowanie możliwości zmiany wymiarów okna konsoli, oraz zablokowanie „scrollowania” podczas gry*
- *Odtwarzanie dźwięków*
- *System kolizji między obiektami*
- *Wyświetlanie obrazu bez migotania*
- *Ruch przeciwników w stronę gracza (przy uwzględnieniu zderzeń ze ścianami areny)*
- *„wymyślenie na nowo” systemu wpisywania znaków do konsoli, co było potrzebne w sekcji CODES, aby użytkownik nie wpisał kodu dłuższego niż 8 znaków.*

- *Animacje obiektów poprzez autorski system z wykorzystaniem instrukcji switch(animation_frame)*
- *Nieskończona, zapętlona rozgrywka*
- *Zapisywanie statystyk do pliku binarnego*

*I tym podobne.*

*PLIKI:*

***header.h*** *– Zawiera wszystkie deklaracje klas, funkcji, oraz zmiennych globalnych, jak i również implementację bibliotek, użytych w projekcie. W tym pliku są również zdefiniowane makra, dla łatwiejszego odczytu programu.*

***MAIN.cpp*** *– Zawiera główną funkcję main(), która steruje przebiegiem gry czyli:*

*- Wyświetla aktualną klatkę .*

*- Sprawdza wszelkie kolizje (gracz – pocisk, przeciwnik – pocisk, przeciwnik – bariera, pocisk – granica areny, pocisk - pocisk).*

*- Wykonuje ruch gracza, przeciwników i pocisków.*

*- Oblicza czas wykonania pojedynczej klatki.*

*- Obsługuje strzelanie pociskami przez gracza jak i przez przeciwników, używając funkcji.*

*- Wykonuje wszelkie animacje.*

*- Wyświetla menu główne.*

*- Zapisuje statystyki do pliku binarnego.*

*- Obsługuje nieskończoną rozgrywkę (m.in. funkcja NewRound()).*

*- Sprawdza, czy gracz powinien otrzymać specjalny kod „HELLMODE”.*

***Projectile.cpp*** *– Plik zawiera definicje funkcji, silnie związanych z obsługiwaniem pocisków.*

***Player.cpp*** *– Plik zawiera definicje funkcji, silnie związanych z obsługiwaniem gracza, oraz konstruktor klasy player.*

***Enemy.cpp*** *- Plik zawiera definicje funkcji, silnie związanych z obsługiwaniem przeciwników, oraz konstruktor klasy enemy.*

***Core.cpp*** *– Plik najbardziej obszerny, który zawiera kod związany z obsługą samego systemu gry. Czyli: definicje zmiennych globalnych, wszelkie definicje funkcji, które są używane do wyświetlenia przeciwników, interfejsu itd. Itp.*

*BIBLIOTEKI:*

***Iostream*** *– Głównie instrukcje cout, służące do wyświetlenia danych w konsoli.*

***Windows.h*** *– nagłówek, który w tym projekcie, służy głównie do obsługi samej konsoli. Użyłem funkcji zmieniających atrybuty konsoli, oraz funkcji, która pozwala mi zmieniać pozycję kursora w konsoli (na tej funkcji oparłem system wyświetlania następnych klatek, aby ekran nie migał, jak przy użyciu instrukcji system(„cls")). Ten nagłówek zawiera również bibliotekę <ctime>, którą wykorzystałem przy generowaniu losowych liczb. Z nagłówka windows.h użyłem również funkcję GetAsyncKeyState, która umożliwiła mi rejestrowanie czy gracz wcisnął jakiś przycisk.*

***Chrono*** *– Chrono wykorzystałem do liczenia czasu, jaki upływa przy wyświetlaniu każdej z klatek. Ten czas, gdyby był większy niż 16 milisekund (co da nam około 60 klatek na sekundę) jest potrzebny do wyliczania czasów odnowienia strzelania gracza, strzelania przeciwników i poruszania się przeciwników.*

***Vector*** *– Służy do tworzenia wektorów, czyli tak jakby dynamicznych tablic. Wykorzystałem to przy tworzeniu pocisków, ponieważ tablica obiektów pocisków musi być dynamiczna.*

***Iomanip*** *– Biblioteka, której użyłem przy wyświetlaniu wyników na interfejsie, aby były wyświetlane w sposób np.: 0030, zamiast 30 (instrukcje setw,setfill).*

***Fstream*** *– Biblioteka obsługująca pliki. Wykorzystałem ją do zapisywania statystyk gracza (highscore) do pliku binarnego, aby w pewien sposób zabezpieczyć plik przed edycją.*

***Conio.h*** *– nagłówek, służący do operowania nad wejściem i wyjściem w konsoli. Wykorzystałem go, aby w menu głównym, rejestrować wciśnięte przez użytkownika przyciski. Różni się to od funkcji GetAsyncKeyState, ponieważ GetAsyncKeyState sprawdza czy klawisz został ostatnio kliknięty, a w nagłówku Conio.h, instrukcja _getch() zwraca wciśnięty klawisz.*

**SFML/Audio.hpp** – Biblioteka obsługująca wielowątkowe odtwarzanie dźwięków, była niezbędna do jednoczesnego odtworzenia kilku efektów dźwiękowych.

**4. Instrukcja użytkownika.**

*Aby program działał poprawnie na systemie Windows 11, należy zmienić w ustawieniach, żeby system używał innej konsoli, niż domyślnej.*

Ścieżka do ustawienia:

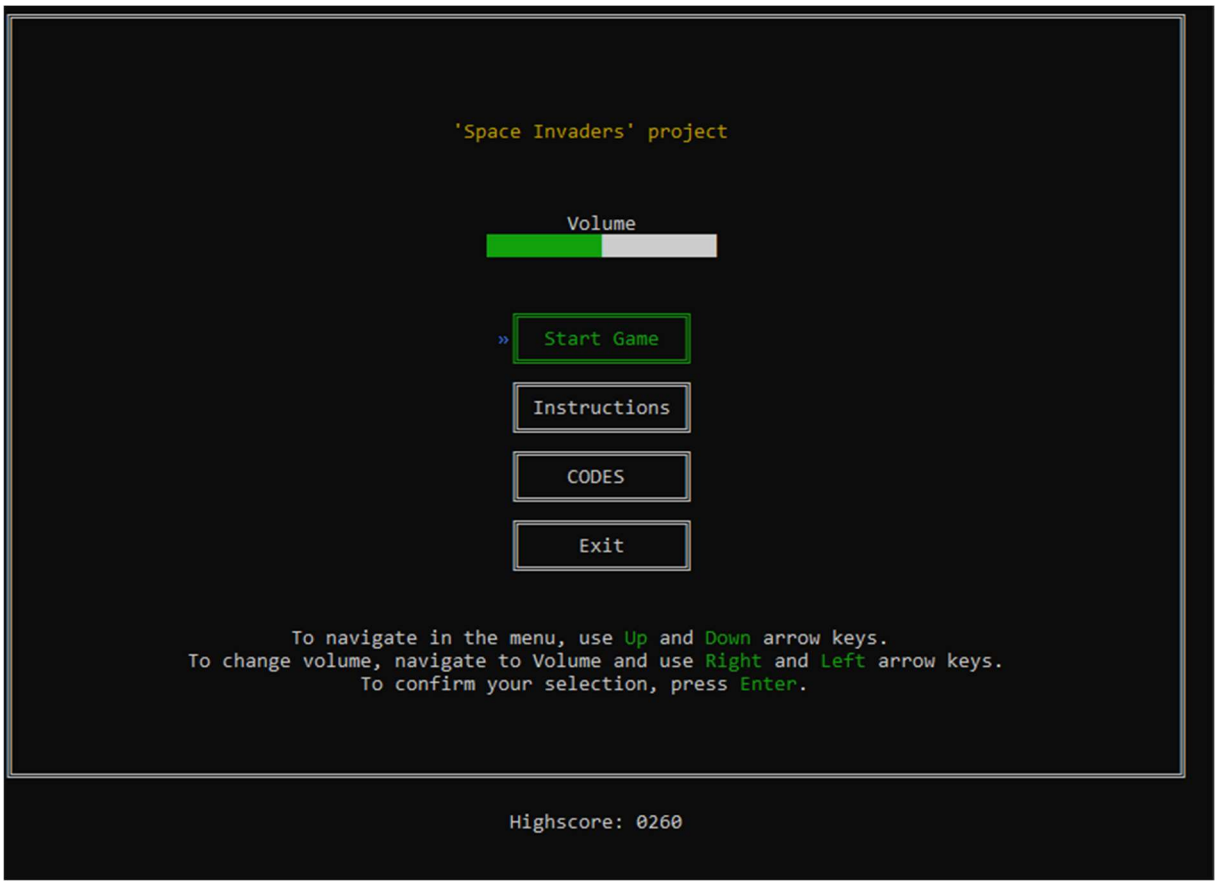
Settings -> System -> For developers -> Terminal -> Windows Console Host

Jak powinno wyglądać poprawne ustawienie:


*W innym wypadku, program może nie wyświetlać poprawnie gry, oraz nie rejestrować poprawnie współrzędnych obiektów (gracz, przeciwnik, pocisk).*

*Na laptopach może być konieczność, aby przełączyć tryb zasilania w systemie z „Najwyższa wydajność zasilania” na „Najwyższa wydajność”. W ten sposób upewnimy się, że program będzie działał płynnie i nie będzie ograniczany przez system.*

***Menu główne:***




*W menu głównym mamy pasek głośności, oraz 4 opcje:*

*Start Game – rozpocznie rozgrywkę.*

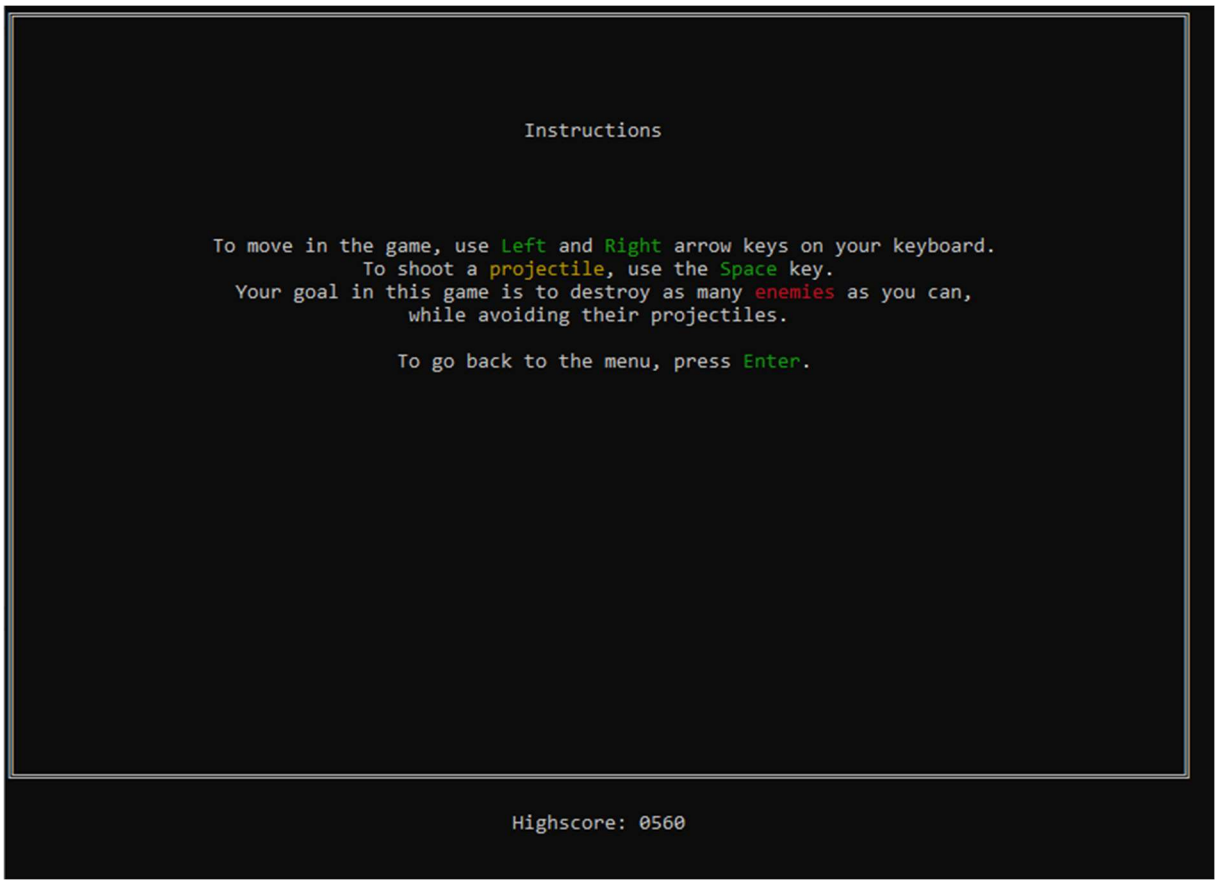
*Instructions – przejdziemy do sekcji z instrukcją.*

*CODES – przejdziemy do sekcji wpisywania kodów.*

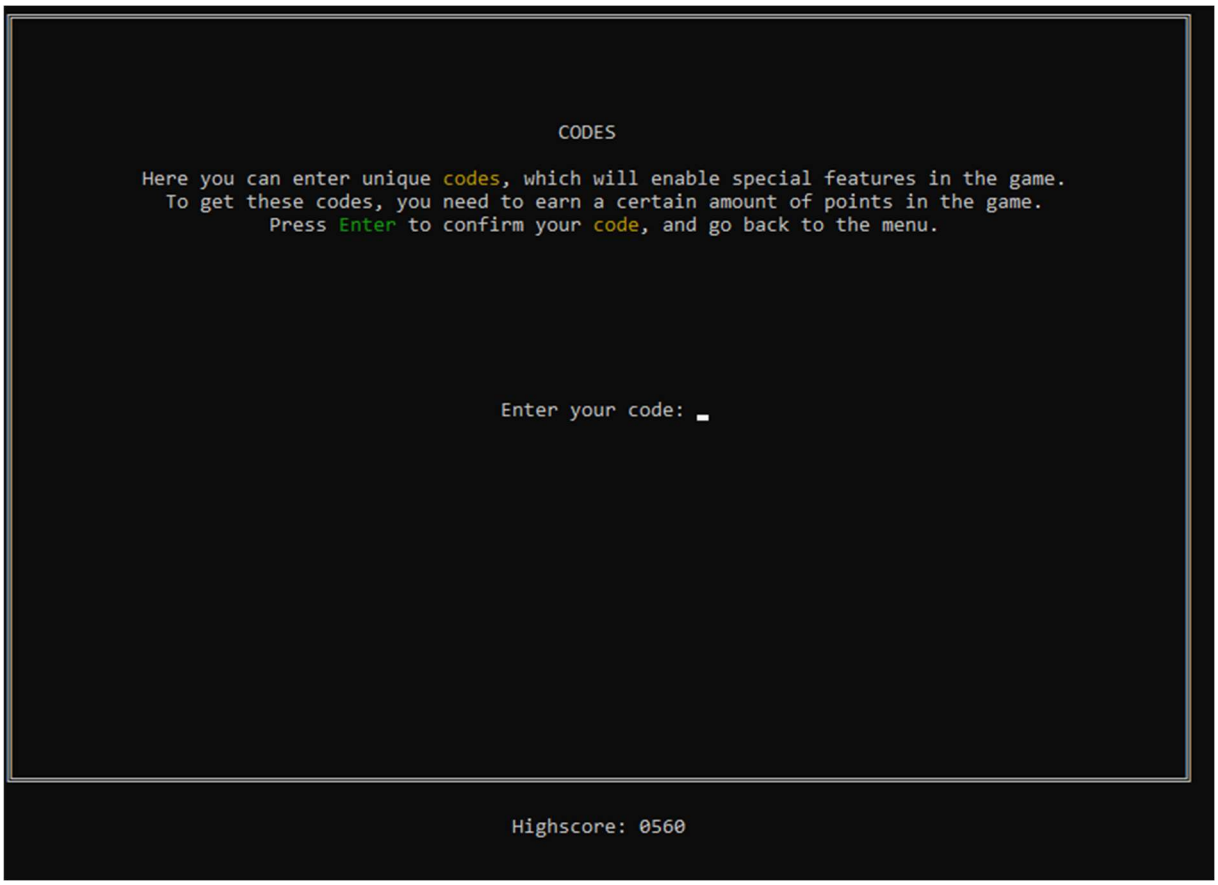
*Exit – wyjdzie z programu.*

*Nawigacja po menu głównym jest umożliwiona za pomocą strzałek góra-dół oraz klawisza Enter, który służy za potwierdzenie wyboru. Aktualny wybór jest podświetlony na zielono, a przy nim znajduje się niebieska strzałka. Aby zmienić głośność, należy użyć przejść strzałkami (góra,dół) na pasek głośności. A następnie użyć strzałek lewa oraz prawa, aby wyregulować głośność. Przy każdej zmianie zostanie odtworzony dźwięk, reprezentujący poziom głośności.*

## *Sekcja Instructions*



***Sekcja CODES***



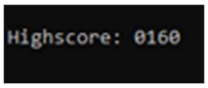

*W tej sekcji mamy możliwość wpisania unikalnych kodów, które mogą odblokować specjalną mechanikę w grze. Kody mają maksymalnie długość 8 znaków, aby zatwierdzić wpisany kod, należy wcisnąć Enter. Aby powrócić do menu głównego, również należy wcisnąć Enter (zatwierdzić dowolny kod, może również być pusty). Po wpisaniu poprawnego kodu (HELLMODE), w menu głównym zamiast pola „Start Game” powinno wyświetlić się pole „HELLMODE”. Gdy wybierzemy to pole, rozpoczniemy rozgrywkę na trybie HELLMODE. Aby zresetować tryb trudności, należy wyłączyć i włączyć program.*

***Przebieg gry***


*Celem gry, jest pokonywanie przybywających przeciwników, strzelając do nich pociskami. Poruszanie się, umożliwiają klawisze strzałek lewa oraz prawa, a pociski możemy wystrzeliwać spacją. Pod areną wyświetlone są statystyki. Patrząc od lewej do prawej mamy:*

*Życia gracza (poparte grafiką, ile jeszcze statków mamy w zanadrzu),*

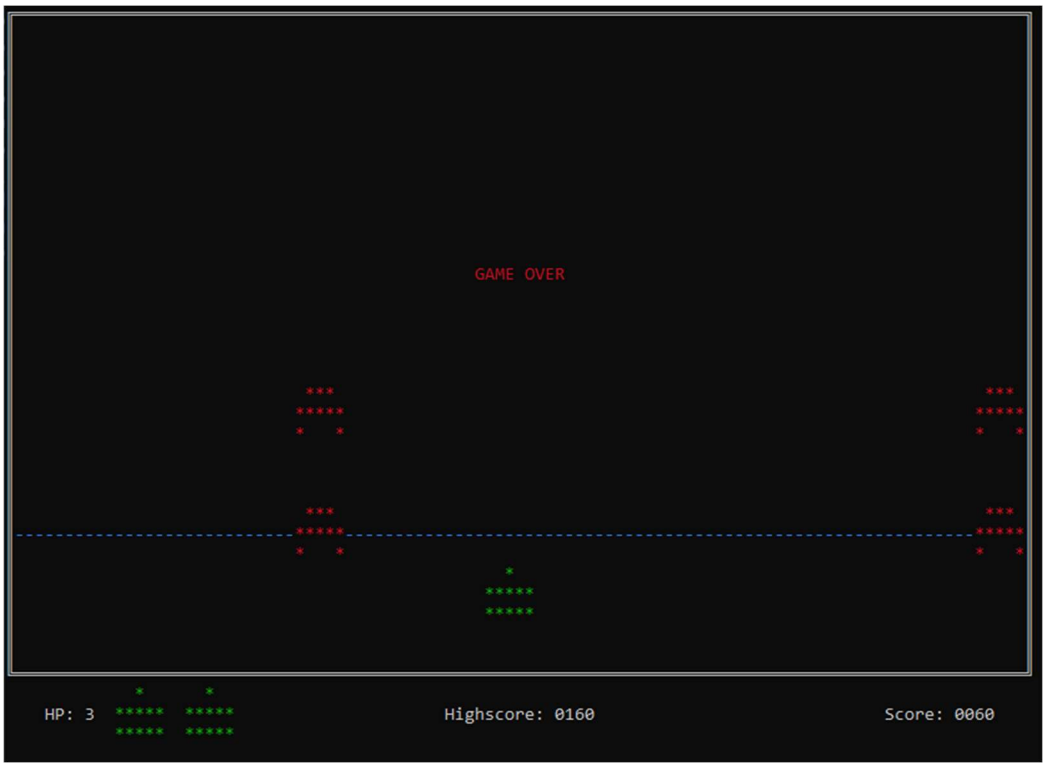

*Highscore, czyli najwyższy wynik, jaki kiedykolwiek uzyskał gracz,*




*Oraz Score, czyli wynik, jaki udało się dotychczas uzyskać w aktualnej rozgrywce.*

*Rozgrywka kończy się, kiedy graczowi skończą się życia, lub gdy przeciwnicy dotrą do niebieskiej linii. Gracz traci życia, kiedy trafiają go pociski wystrzelone przez przeciwników. Za każdego trafionego przeciwnika, gracz otrzymuje 10 punktów. Nie ma możliwości „zapauzowania” rozgrywki.*

*Przykłady przegranej:*




**5. Podsumowanie i wnioski.**

*Udało mi się zrealizować postawione przez siebie wyzwanie, prawie w 100 procentach. Problemy pojawiały się w wielu miejscach takich jak:*

- *Niepoprawne wymiary konsoli, co przełożyło się na niepoprawne interpretowanie pozycji obiektów.*
- *Kolizje pocisków z innymi pociskami.*
- *Zablokowanie wszelkiego przesuwania (scrollowania) w konsoli, zmiany rozmiaru konsoli itd. Itp.*
- *Wyświetlanie obrazu w konsoli bez migotania*
- *Losowanie przeciwnika, który ma wystrzelić pocisk*
- *Ruch przeciwników*
- *Animacje obiektów*
- *Zablokowanie możliwości wpisania kodu, dłuższego niż 8 znaków*

*Każdy z tych problemów wymagał zastanowienia się nad każdą możliwą kombinacją zdarzeń, aby wszystko zawsze ze sobą współgrało. Jednak pomimo tych trudności, udało mi się je rozwiązać i końcowo zaimplementować w moim projekcie, przy okazji zwracając uwagę na optymalizację.*