

Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Zaawansowane programowanie w C++

Dokumentacja wstępna

Kamil Gabryjelski, Antoni Róžański

Prowadzący: Konrad Grochowski

Warszawa, 2017

1. Temat projektu

Tematem projektu jest gra Core Wars. Dwóch graczy pisze programy (tak zwanych "wojowników") w języku podobnym do assemblera, które rywalizują między sobą o kontrolę zasobów wirtualnej maszyny. Wygrywa ten gracz, którego procesy zajmą całą pamięć lub wyeliminują wszystkie procesy przeciwnika. Pojedynek toczony jest w specjalnym symulatorze, który dba o kolejność wykonywania instrukcji i czuwa nad ich poprawnością.

2. Funkcjonalność

Nasz projekt będzie oferował użytkownikom następujące funkcjonalności:

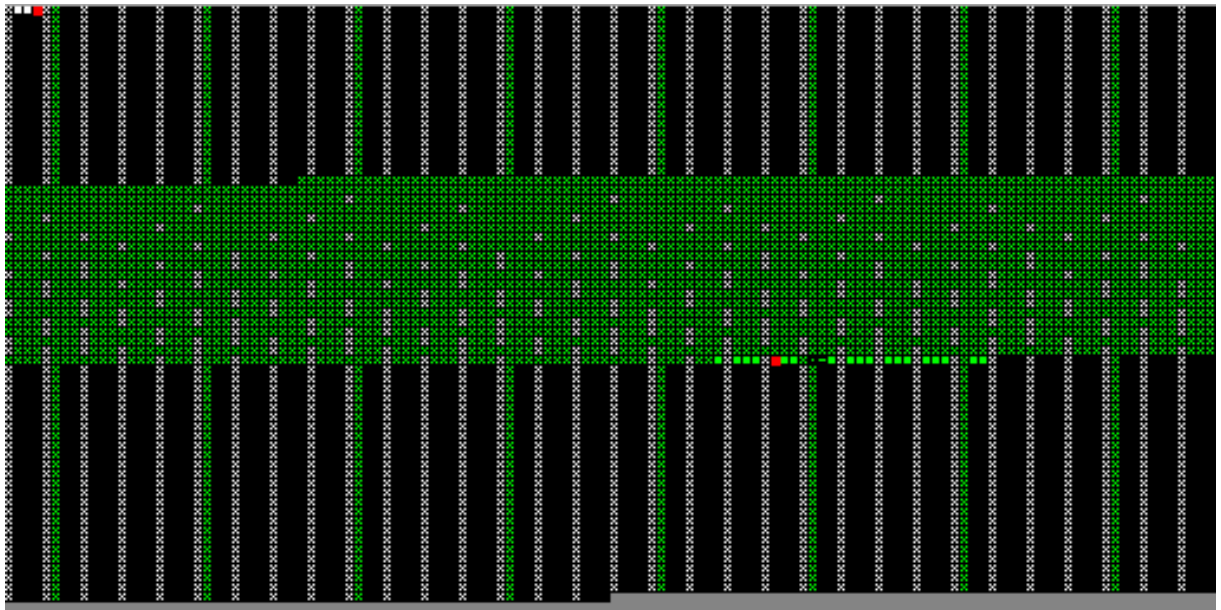
2.1. Interfejs graficzny

Użytkownikowi zostanie udostępniony interfejs graficzny z poziomu przeglądarki internetowej. Za jego pomocą będzie odbywać się wszelka interakcja użytkownika z grą, a będzie się ona składać z następujących etapów:

1. Wprowadzenie wojownika poprzez wpisanie w odpowiednim polu kodu lub wczytanie istniejącego już zestawu instrukcji z pliku/bazy danych;
2. Obserwacja walki wraz z możliwością ingerencji w ustawienia, m. in. szybkości rozgrywki;
3. Wyświetlenie interesujących danych dotyczących walki;
4. Zapis wojownika/statystyk do bazy danych.

Kolejność, w której wymieniliśmy funkcjonalności odpowiada naszym priorytetom. Integracja z bazą danych jest elementem, który zostanie zaimplementowany, jeśli pozostałe etapy zostaną ukończone przed terminem oddania projektu.

Zamierzamy zwizualizować stan rywalizacji jako dwuwymiarową tablicę, jak na rys. 2.1, gdzie każdej komórce odpowiadać będzie adresowi w pamięci. Kolory poszczególnych komórek odpowiadać będą procesom użytkowników, co umożliwi ocenę, jaką część pamięci kontroluje każdy z graczy.



Rys. 2.1: Przykładowa plansza obrazująca stan rozgrywki

2.2. Tworzenie wojowników (Wprowadzanie instrukcji)

Każdy wojownik (program w języku RedCode) będzie się składał z zestawu instrukcji, opisanych w punkcie 2.3. Gracze będą mieli możliwość wprowadzenia tych instrukcji do programu na kilka sposobów:

1. Wprowadzenie instrukcji z poziomu interfejsu graficznego lub z pliku tekstowego:
Gracz dostanie możliwość tworzenia programu poprzez wpisanie kodu w wyznaczonym miejscu interfejsu graficznego, lub załadowania kodu z pliku tekstowego. Zostanie zaimplementowana walidacja kodu pod względem poprawności składni.
2. Pobranie kodu z relacyjnej bazy danych, np. SQLite

2.3. Instrukcje RedCode

Językiem, w którym gracze będą pisali swoje programy, będzie Redcode. Zestaw instrukcji, które zamierzamy zaimplementować, jest zgodny ze standardem *ICWS '88* i zawiera polecenia: *MOV*, *ADD*, *SUB*, *JMP*, *JMZ*, *JMN*, *CMP*, *SLT*, *DJN*, *SPL*, *NOP*.

3. Narzędzia

Narzędzia, z których zamierzamy korzystać:

1. Środowisko programistyczne - JetBrains CLion
2. Kompilatory - Windows: MinGW oraz Linux: GCC
3. Testy jednostkowe - Google Test
4. Kontrola wersji - git/GitHub
5. Optymalizacja kodu - gprof
6. Generowanie dokumentacji - Doxygen
7. Komunikacja - Slack