Instrukcja obsługi

Uruchamianie programu:

1) Aby zapisać świat do pliku (domyślna nazwa pliku to test.txt, można zmienić w makefile albo wywołać program ręcznie wpisując ./bot_nieinteligenty do_pliku nazwa.txt), wpisać w konsoli: make do_pliku
2) Aby wyświetlić świat w konsoli, wpisać: make wypisz
3) Aby zresetować świat, wpisać: make reset
4) Aby usunąć plik wykonywalny oraz txt, wpisać: make wyczysc

Oznaczenia użyte w programie

1 = trawa

2 = piasek

8 = ściana

Podział programu na pliki:

Program został podzielony na pięć plików .c i cztery pliki .h

1) lacznosc.c składa się z dwóch funkcji odpowiedzialnych za komunikację z serwerem
2) lacznosc.h zawiera deklaracje z lacznosc.c oraz strukturę Memory przechowującą odpowiedzi z serwera
3) macierz.c zajmuje się alokowaniem miejsca na macierze, zwalnianiem pamięci, zerowaniem macierzy i przepisywaniem macierzy
4) macierz.h zawiera kluczową strukturę Macierz, która przewija się w całym programie
5) mozg.c plik składa się z funkcji kontrolującej rozmiar macierzy, oraz dwóch rozbudowanych funkcji opartych na strukturze Macierz:
    a) pierwszy_bot odpowiada za eksplorowanie świata i wpisywanie rodzaju podłoża do macierzy
    b) drugi_bot jest sercem programu i to on zawiera główną pętle wykonującą. Wywołuje pierwszego bota, zwiększa macierz o ile jest to potrzebne, oraz zarządza ruchem czołgu.
6) mozg.h zawiera prostą strukturę AkPol (Aktualna pozycja) która jest używana do kontrolowania potrzebnego miejsca w macierzy
7) zapis.c zawiera dwie proste, niemal bliźniacze funkcje, jedna odpowiedzialna jest za wypisywanie świata w konsoli, druga za jego zapis do pliku
8) zapis.h
9) main.c główny plik został skrócony do możliwie najmniejszej formy, składa się z jednej funkcji main w środku której znajdziemy trzy polecenia if, które na podstawie linii komend rozpoznają którą z trzech operacji (reset, do_pliku, wypisz) wykonać.

Działanie sztucznej inteligencji:

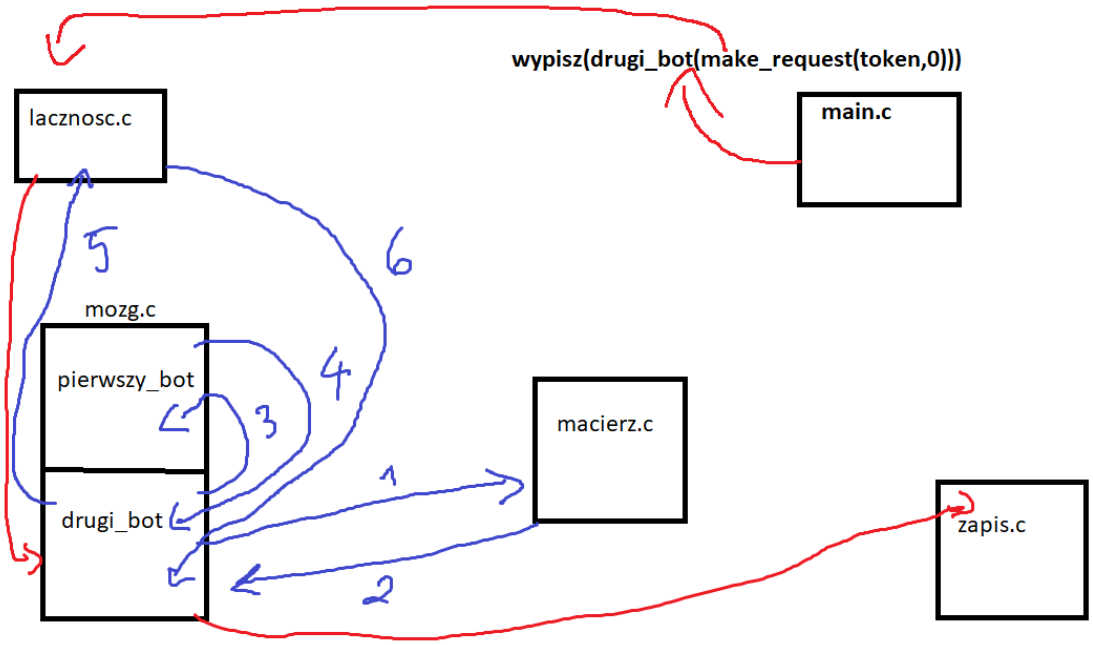
Tak jak sugeruje nazwa programu, wykorzystany w programie algorytm jest stosunkowo prymitywny i opiera się na jednej zależności jaką jest pole bezpośrednio przed czołgiem. Jeżeli przed czołgiem znajduje się ściana (8) czołg skręca w prawo, w przeciwnym wypadku idzie prosto, skręca w lewo oraz bez sprawdzania co jest przed nim próbuje iść prosto (program nie wie czy mu się udało czy nie). Następnie schemat ten jest powtarzany 50*(liczba powiększeń macierzy) razy. Mechanizm jest prosty lecz mimo tego jest w stanie odkryć wszystkie (z wyjątkiem jednego, o tym dalej) pola, niezależnie od położenia początkowego czy też zwrotu.

Zarządzanie pamięcią

Drugi_bot tworzy domyślnie macierz początkową M 5x5, chyba że położenie początkowe ma którąś ze współrzędnych większą niż pięć, wtedy przy pomocy pętli while, macierz ta ma wielkość odpowiednich wielokrotności pięciu. Dla uproszczenia w programie używane są macierze kwadratowe, lecz struktura pozwala na zamianę ich na macierz prostokątną. Następnie tworzona jest macierz B w której będzie zapisywana mapa, ma ona początkowo taki sam wymiar jak macierz M. W głównej części programu gdy aktualne położenie czołgu wykroczy po

za obszar macierzy, macierz B jest realokowana. Program posiada funkcję do zwalniana zaalokowanych macierzy, jednak jest ona mało skuteczna.

Prosty schemat kolejności funkcji na przykładzie polecenia wypisz(na niebiesko główna pętla wykonawcza)



Wady i błędy programu

-Problemy z pamięcią, bardzo duże wycieki

- Czasami program się zatnie, trzeba nacisnąć Ctrl + c i uruchomić go ponownie

- Czołg czasami kręci się w kółko, ale mimo tego mapa zazwyczaj powstaje

- Program nie odkrywa pola na mapie w miejscu (zaznaczone obok)

- Program wolno działa i generuje dużą liczbę kroków