1. URUCHAMIANIE PROGRAMU

Program uruchamia się z makefile, używając odpowiednich komend:

make - kompiluje oraz uruchamia główny program do automatycznego odkrywania mapy

make clean - usuwa pliki wykonywalne powstałe po kompilacji

make move/lewo/prawo/info/explore - wykonuje pojedynczy ruch czołgu, odpowiedni do wpisanej komendy po "make"

make duzo - wykonuje przykładową serię ruchów czołgu

Po uruchomieniu trzeba wpisać token swojego świata

2. UŻYWANE STRUKTURY

```
      typedef struct _Odp

{
    char *status;
    char *name;
    int current_x;
    int current_y;
    char *current_session;
    char *direction;
    int step;
    char *field_type;
    char *field_bonus;
    int x1, y1, x2, y2, x3, y3;
    char *type1, *type2, *type3;
} odp;

typedef struct _Expl{
    char *status;
    int x;
    int y;
    char *type;
}expl;

typedef struct _Field
{
    char **mapa;    //nasza mapka
    int now_x;      //aktualne położenie "x"
    int now_y;      // --||--
    int d_x;        //delta "x", tak zwany offset
    int d_y;
    int rozmiar_x;  //rozmiar naszej planszy
    int rozmiar_y;  //rozmiar naszej planszy
    char *dir;      //zwrot czołgu
    char *komenda;
    expl *pole[2];
    odp *response;
```

```
}field;
```

3. SCHEMAT DZIAŁANIA PROGRAMU

Program na początku alokuje pamięć dla struktury odp

Program prosi o wpisanie tokenu świata

Program szuka obwiedni zadanego świata

Program szuka dolnej granicy zadanego świata

Program przeczesuje świat od „lewego dołu" do „prawej góry"

Program zapisuje odkryty świat do pliku save.txt

4. MODUŁY

Program został podzielony na 4 moduły i główny main

Struktury.h - w tym module znajdują się nasze struktury

Komunikacja.h/.c – odpowiada za komunikację naszego programu z API serwera http://edi.iem.pw.edu.pl:30000/ poprzez skonfigurowanie i wysłanie zapytania typu GET, a następnie odebranie odpowiedzi z serwera

Funkcje.h/.c – odpowiada za przetworzenie odpowiedzi z serwera i wpisanie jej elementów do struktury oraz za m.in. przypisanie elementów otoczenia do lokalnego świata, wypisanie świata w konsoli, czy zapisanie do pliku

Automat.c\.h – odpowiada za autonomiczność poruszania się czołgu po wirtualnym świecie oraz zawiera funkcje łączące komunikację z API z przetwarzaniem odpowiedzi

Utworzyliśmy taki podział, ze względu na zastosowanie poszczególnych funkcji

5. JAK DZIAŁA AUTONOMICZNOŚĆ CZOŁGU

Funkcja szukaj_obwiedni – opiera się na różnicy skrętów w prawo i skrętów w lewo podczas objeżdżania obiektów. Czołg rusza z położenia startowego i jedzie przed siebie do momentu napotkania ściany. Następnie sprawdza, czy dojechał do środka ściany czy do rogu (jest to niezbędne, żeby różnica skrętów była prawidłowa) i obraca się w prawo. Funkcja zapamiętuje aktualne położenie czołgu, żeby po objechaniu świata dookoła był spełniony warunek wyjścia z pętli. Następnie czołg sprawdza za każdym ruchem możliwość skrętu w lewo i jeżeli takowa występuje to wykonuje skręt. Jeżeli takiej nie ma to jedzie przed siebie, a

w momencie dojechania do ściany obraca się w prawo. Po dojechaniu do miejsca początkowego sprawdza różnicę pomiędzy liczbą skrętów w prawo i liczbą skrętów w lewo. Jeżeli jest ona równa 4 to funkcja się kończy, a jeżeli jest różna od 4 to funkcja szukaj_obwiedni wywołuje się rekurencyjnie.

Funkcja szukaj_granicy – ustawia czołg w lewym dolnym rogu świata. Na początku czołg zjeżdża w dół aż do napotkania ściany. Następnie skręca w prawo i stara się jadąc naprzód skręcić w lewo. Jeżeli taka możliwość nastąpi to funkcja wywołuje się rekurencyjnie. Jeżeli możliwości skrętu nie będzie, to czołg przejeżdża dolną granicę upewniając się, że nie ma żadnej możliwości zjechania niżej. Jeżeli takowa jest to funkcja wywołuje się rekurencyjnie, a jeżeli nie ma to czołg ustawia się w lewym dolnym rogu skierowany w kierunku E.

Funkcja czyść_mape – czołg przeczesuje świat poziomo. Za każdym razem dojeżdża do ściany i upewnia się że przed nim faktycznie jest ściana a nie tylko jeden bloczek przeszkody. Jeżeli będzie to tylko jeden bloczek, to funkcja wywoła się rekurencyjnie.