Zasady i zestaw zadań

Gliwice, 10-11 kwietnia 2012

Spis treści

1 Wstęp

Już po raz czwarty spotykamy się na finale konkursu Deadline24 organizowanego przez firmę Future Processing, tym razem także pod zaszczytnym patronatem rektorów, dziekanów uniwersytetów i uczelni technicznych. W tym roku przenieśliśmy miejsce finału do Zabytkowej Kopalni Węgla Kamiennego Guido, która jest także współorganizatorem tego wydarzenia. Mamy nadzieję, że ta unikalna sceneria konkursu będzie niezapomnianym przeżyciem w czasie zaciętej walki o każdy punkt do ostatniej sekundy konkursu.

W tegorocznej edycji przygotowaliśmy trzy zadania, tradycyjnie osadzone w świecie cywilizacji żukoskoczków, ambitnych stworzeń ogarniętych chęcią podboju kosmosu. Będziemy mieli okazję wraz z nimi przenieść się do kopalni także wirtualnie (zadanie **Fedrunek**), opracować system wspierający produkcję zbrojeniową żukoskoczków (zadanie **Wycinanka**), a wreszcie powrócić na znaną z poprzedniej edycji planetę CFK-1, zamieszkaną przez krwiożercze coory (zadanie **Osada**).

Mamy nadzieję, że rozgrywka będzie emocjonująca, a system konkursowy i wszyscy uczestnicy wytrzymają napięcie przez całe 24 godziny. Niech wygra najlepszy!

Organizatorzy

2 Komunikacja z serwerem

Uzyskiwanie aktualnych informacji o wirtualnym świecie oraz wydawanie rozkazów jest możliwe za pomocą protokołu TCP/IP. Drużyna łączy się jako klient do odpowiedniego serwera konkursowego. Adres IP oraz port, z którym należy się połączyć są podane w sekcjach "Serwery" specyfikacji poszczególnych zadań. Można nawiązać wiele połączeń jednocześnie, jednak sumaryczny transfer przypadający na każdy komputer jest ograniczony. Maksymalna liczba połączeń i maksymalny transfer podane są w "Ustaleniach Technicznych". Komunikacja odbywa się w trybie tekstowym. Bezpośrednio po połączeniu należy się zalogować, następnie sesja przechodzi w tryb poleceń.

2.1 Logowanie

Bezpośrednio po nawiązaniu połączenia serwer wysyła prośbę o login zakończoną znakiem końca linii: `LOGIN`. Należy wysłać swój login a następnie znak końca linii. Następnie serwer zapyta o hasło (`PASS`), na co należy analogicznie odpowiedzieć hasłem. Jeśli autoryzacja przebiegła pomyślnie serwer odpowie ciągiem znaków: `OK` i przejdzie w stan oczekiwania na komendy. W przeciwnym wypadku dostaniemy odpowiedź `FAILED 1 bad login or password` po czym nastąpi zamknięcie połączenia.

Poniżej znajduje się przykładowy zapis komunikacji w czasie logowania.

klient → serwer	serwer → klient
	`LOGIN`
`login1`	
	`PASS`
`secret`	
	`OK`

2.2 Komendy

Każde polecenie składa się z nazwy komendy, argumentów (liczba zależna od polecenia) oraz znaku końca linii. Parametry powinny być oddzielone co najmniej jednym białym znakiem.

Na każdą komendę serwer odpowiada jednym z poniższych ciągów:

- '`OK`' — w przypadku zaakceptowania komendy
- '`FAILED` *e msg*' — w przypadku błędu; gdzie *e* to kod błędu, a *msg* — komunikat błędu.

Następnie zależnie od komendy serwer może opcjonalnie wysłać lub odebrać dodatkowe dane. Jeśli dodatkowe dane są wysyłane od klienta do serwera, to po odebraniu tych danych serwer ponownie odpowie w sposób opisany powyżej. Przykładowe zapisy komunikacji z serwerem oraz zestawienia możliwych błędów dla poszczególnych zadań znajdują się w opisie każdego z nich.

Ograniczenie liczby komend Na każdym serwerze każdego z zadań obowiązuje limit na maksymalną liczbę komend wydawanych w czasie tury. Zbliżenie się do limitu i jego osiągnięcie zostanie zasygnalizowane odpowiednimi błędami.

kod błędu	komunikat błędu
6	commands limit reached, next call will force waiting
7	commands limit reached, forced waiting activated

Po wystąpieniu drugiego z błędów serwer prześle dodatkowy komunikat

FORCED_WAITING x — gdzie x oznacza ilość sekund do zakończenia oczekiwania, tj. do końca bieżącej tury

2.3 Konwencje

Jeśli nie jest zaznaczone inaczej, to przyjmujemy że:

- Każda linia zakończona jest pojedynczym znakiem o kodzie ASCII 10 ('\n'). Znak powrotu karetki ('\r'; kod 13) towarzyszący mu na niektórych systemach operacyjnych będzie traktowany jako biały znak.
- W przypadku danych przesyłanych przez serwer, liczby oraz wyrazy oddzielone są pojedynczą spacją.
- W przypadku danych przesyłanych od klienta do serwera (np. parametry komendy) dozwolona jest dowolna (niezerowa) ilość białych znaków pomiędzy danymi, a także na końcu oraz na początku linii.
- Za białe znaki uznajemy: spację, powrót karetki ('\r') oraz tabulator ('\t').

3 Punktacja ogólna

Punkty Na każdym serwerze drużyny zdobywają punkty w sposób zdefiniowany w specyfikacji zadania rozgrywanego na danym serwerze. Punkty zdobyte na poszczególnych serwerach są przeliczane na **punkty rankingowe**, które definiujemy jako iloczyn liczby 100 oraz stosunku wyniku drużyny (na danym serwerze) do średniej arytmetycznej z trzech najlepszych wyników (na danym serwerze). Punkty przyznawane drużynie za konkretne zadanie to średnia punktow rankingowych z wszystkich serwerów danego zadania.

Ranking Ranking zawodów jest rankingiem sumy punktów rankingowych wszystkich zadań.

4 Sytuacje awaryjne

W razie wystąpienia sytuacji, w której nie wszystkie drużyny będą w stanie brać udział w konkursie na ustalonych zasadach (np. brak prądu, awaria sieci lokalnej lub jej części, problemy z systemem konkursowym, etc.) oraz wina nie leży po stronie tych drużyn ani ich sprzętu, organizatorzy wstrzymają działanie systemu konkursowego, a o jego ponownym uruchomieniu uczestnicy zostaną poinformowani w lokalnym serwisie WWW konkursu. W tym czasie punkty nie będą naliczane. W takiej sytuacji mogą zostać zerwane wszystkie połączenia z serwerem konkursowym.

5 Zadanie Fedrunek

5.1 Wprowadzenie

Żukoskoczki nie potrzebują pragmatycznych motywacji do podjęcia decyzji o kolejnych inwazjach - ambicja i ekspansywność leży w ich naturze. Nie znaczy to oczywiście, że bogate złoża surowców na rozważanej planecie nie działają na wyobraźnię wojskowych planistów!

W ostatnich latach żukoskoczki opanowały wyjątkowo bogaty w cenne metale układ planet, co dla Ministerstwa Nieograniczonej Eksploatacji okazało się klęską urodzaju - żukoskoczkom brakuje mocy przerobowych do skutecznego wydobywania złóż, które zdobyli. Ministerstwo postanowiło zaprosić do współpracy najlepsze uczelnie techniczne w galaktyce i przygotować nową generację robotów poszukiwawczych i wydobywczych.

Waszym (specjalnie zatrudnionych inżynierów wyspecjalizowanych w pracy pod ziemią) zadaniem jest sterowanie pracą floty robotów powierzoną Wam przez Ministerstwo tak, aby zyski z wydobycia były jak największe.

5.2 Model rozgrywki

Rozgrywka odbywa się w turach równej długości. Pomiędzy turami drużyny komunikują się z serwerem i wydają polecenia swoim jednostkom. Każda operacja podejmowana przez jednostkę trwa określoną, całkowitą liczbę tur, po których upływie można wydać kolejne polecenie danej jednostce.

Zakończenie rozgrywki W miarę postępów w wydobyciu wyszukiwanie surowców na pustoszejącej asteroidzie staje się coraz trudniejsze, i co za tym idzie - coraz mniej opłacalne. Po upływie 50 tur od momentu wydobycia 90% surowców danej asteroidy żukoskoczki decydują o zakończeniu eksploatacji i opuszczeniu asteroidy.

5.3 Świat

Rozgrywka toczy się na asteroidzie reprezentowanej przez regularną kostkę o boku długości N, złożoną z N^3 jednostkowych sześcianów, które nazywamy **polami**. Każde pole jest identyfikowane przez trójkę współrzędnych w układzie kartezjańskim: P_X, P_Y, P_Z, gdzie każda współrzędna przyjmuje całkowite wartości z przedziału od 1 do N. Pola dzielimy na dwie grupy:

- Pola na powierzchni - pola położone na jednej ze ścian (być może na krawędzi lub w rogu asteroidy, a więc na dwóch lub trzech ścianach jednocześnie).
- Pola podziemne - pozostałe pola.

5.3.1 Pola na powierzchni

W niektórych polach na powierzchni znajdują się elementy infrastruktury dostępnej na asteroidzie:

- **Punkty skupu** metali - miejsca, w którym metale wymieniane są na kredyty - walutę żukoskoczków.
- **Warsztaty** - miejsca, w których za kredyty można ulepszać istniejące roboty i kupować nowe.

W każdym polu znajduje się co najwyżej jeden z powyższych budynków. Roboty żukoskoczków są wyjątkowo chwytne i mogą poruszać się po powierzchni asteroidy na każdej z jej ścian (a także przechodzić z jednej ściany na drugą).

5.3.2 Pola podziemne

Każde z podziemnych pól zbudowane jest z jednego z dwóch materiałów:

- **gruntu** lub
- **kamienia**.

Poszczególne materiały różnią się czasem potrzebnym do wydrążenia w nich tunelu.

Każde pole może (ale nie musi) zawierać jedną lub więcej jednostek rud metali:

- **żelaza**,
- **srebra** lub
- **złota**.

Liczba jednostek każdego metalu występująca w pojedynczym polu wyraża się liczbą naturalną z zakresu $[0, 10]$, przy czym liczba jednostek każdego z metali jest niezależna od pozostałych.

5.4 Roboty

Każda drużyna dysponuje flotą robotów, wśród których wyróżniamy dwa rodzaje:

- **Poszukiwacz** - porusza się wyłącznie w polach na powierzchni i wykonuje **sondy** oraz **odwierty**.
- **Górnik** - porusza się zarówno w polach na powierzchni jak i w polach podziemnych; drąży tunele, wydobywa metale i dostarcza je do punktów skupu.

Każdy robot opisywany jest przez zestaw cech, różny dla poszukiwacza i górnika. Każdą z cech można ulepszać (osobno, niezależnie od pozostałych cech) w przygotowanych na asteroidzie warsztatach. W każdym polu asteroidy, łącznie z warsztatami i punktami skupu, w dowolnym momencie może przebywać dowolnie wiele robotów (być może różnych drużyn).

5.4.1 Poszukiwacz

Poszukiwacz porusza się wyłącznie po powierzchni i wykonuje operacje badawcze dostarczające informacji o terenie w obszarze pól znajdujących się w danym momencie bezpośrednio pod robotem.

Operacje

- **Sonda** - dostarcza informacji o sumarycznej liczbie jednostek poszczególnych metali w $N-2$ polach znajdujących się pod poszukiwaczem oraz o sumarycznej liczbie pól zbudowanych z poszczególnych materiałów (grunt lub kamień). Wszystkie informacje uzyskane z sondy odzwierciedlają stan pól w ostatniej turze jej trwania.
- **Odwiert** - dostarcza szczegółowych informacji o materiałach i obecności metali w K polach pod poszukiwaczem. Czas trwania odwiertu jest monotonicznie zależny od wartości K. Wszystkie informacje uzyskane w odwiercie odzwierciedlają stan pól w ostatniej turze trwania odwiertu.

Cechy Podane w nawiasie wartości odzwierciedlają dany parametr na kolejnych poziomach ulepszeń wykonywanych w warsztatach.

- Czas wykonywania sondy w turach (3 / 2 / 1).
- Czas wykonywania odwiertu głębokiego na trzy pola w turach (3 / 2 / 1).

5.4.2 Górnik

Górnik porusza się zarówno po powierzchni bryły jak i pod ziemią, gdzie drąży tunele i wydobywa metale. Te z kolei przechowuje w swojej ładowni, a następnie dostarcza do punktów skupu na powierzchni.

Operacje

- **Drążenie** - konstruuje tunel pomiędzy sąsiadującymi polami (w dowolnym z sześciu kierunków), z których co najmniej jedno jest polem podziemnym.
- **Wydobycie** - przenosi jedną jednostkę metalu ze złoża do ładowni górnika.

Cechy Podane w nawiasie wartości odzwierciedlają dany parametr na kolejnych poziomach ulepszeń wykonywanych w warsztatach.

- **Ładowność** (5 / 10 / 15) - pojemność ładowni robota mierzona liczbą wydobytych jednostek metali, które może przenosić robot.
- **Szybkość** - czas drążenia tunelu (zależny od materiału w polu **do** którego prowadzi drążony tunel) (grunt: 4, kamień: 10 / grunt: 3, kamień: 8 / grunt: 2, kamień: 5).
- **Wydajność** - czas wydobywania jednej jednostki dowolnego metalu w turach (3 / 2 / 1).

5.4.3 Początkowy stan floty robotów

W momencie rozpoczęcia rozgrywki wszystkie roboty należące do drużyny znajdują się w tym samym polu na powierzchni asteroidy, różnym od pól, w których znajdują się roboty pozostałych drużyn. Początkowa flota robotów będzie składać się z od jednego do pięciu poszukiwaczy i od jednego do pięciu górników; początkowa liczba robotów każdego rodzaju w pojedynczej rozgrywce będzie ta sama dla wszystkich drużyn.

5.5 Eksploracja i eksploatacja

W momencie rozpoczęcia rozgrywki asteroida jest w naturalnym stanie - pomiędzy jej podziemnymi polami nie ma żadnych tuneli. Zadaniem drużyny jest eksploracja podziemnych złóż asteroidy i sprzedaż wydobytych metali w punktach skupu.

5.5.1 Drążenie tuneli

Jeśli robot ma przejść z pola a do sąsiedniego, podziemnego pola b (lub z podziemnego pola na powierzchnię), pomiędzy tymi polami musi istnieć tunel. Operację drążenia tunelu wykonuje robot-górnik, a czas trwania jego pracy (mierzony całkowitą liczbą tur) zależy od materiału, z którego zbudowane jest pole docelowe (grunt lub kamień, w przypadku drążenia tunelu do pola znajdującego się na powierzchni, za jego materiał przyjmujemy grunt) oraz poziomu możliwości drążenia danego robota.

Drążenie tunelu wprawia materię w wibracje, które uniemożliwiają innym robotom drążenie tunelu pomiędzy tą samą parą pól. Oznacza to, że dwa roboty (tej samej drużyny albo różnych drużyn) nie mogą jednocześnie wiercić tunelu pomiędzy tą samą parą pól (nawet, jeśli wiercą w przeciwnych kierunkach). Raz wydrążony tunel pozostaje na planszy do końca rozgrywki i jest dostępny dla wszystkich drużyn. Tunele są dwukierunkowe.

Tunel między polami należy rozumieć jako wąski chodnik o średnicy istotnie mniejszej od wymiarów pojedynczego pola. Tunele nie naruszają stabilności terenu, w szczególności wydrążenie tunelu **nie** powoduje zapadania się terenu z pól znajdujących się powyżej.

5.5.2 Wydobycie

Operację wydobycia definiujemy jako przeniesienie jednej jednostki metalu ze złoża w polu, w którym znajduje się górnik do jego ładowni. Czas trwania pracy (mierzony całkowitą liczbą tur) jest zależny od poziomu możliwości wydobycia danego robota. Wydobycie w jednym momencie w tym samym polu może prowadzić wiele robotów, także różnych drużyn. W momencie wydania polecenia rozpoczęcia wydobycia, wydobywana jednostka metalu jest natychmiast rezerwowana dla danego robota i od tego momentu przestaje być dostępna (i widoczna) dla pozostałych robotów.

5.6 Przychody i rozchody

Walutą stosowaną przez żukoskoczki w górniczych rozliczeniach są *kredyty*. Każda drużyna zaczyna grę dysponując sumą 20 kredytów.

5.6.1 Rozbudowa floty

Drużyna może wydawać kredyty na rozbudowę i ulepszenia swojej floty robotów, wg cennika poniżej:

operacja	koszt
Ulepszenie cechy poszukiwacza	5
Ulepszenie cechy górnika	10
Zakup nowego poszukiwacza	50
Zakup nowego górnika	100

Aby ulepszyć cechę robota, dany robot musi znajdować się w warsztacie. Operacja ulepszenia trwa jedną turę, w tym samym momencie w warsztacie może być ulepszanych wiele robotów. Aby kupić nowego robota należy wskazać warsztat, w którym nowy robot ma się pojawić. Liczba robotów we flocie drużyny jest ograniczona i może wynosić maksymalnie 25.

5.6.2 Sprzedaż metali

Kiedy górnik wchodzi na pole skupu metali, zawartość jego ładowni jest (automatycznie i od razu) wyprzedawana i zamieniana na kredyty, wg cennika poniżej:

metal	cena
żelazo	1
srebro	3
złoto	5

5.7 Punktacja

Końcowy wynik drużyny jest wyliczany wg wzoru: $(A + \frac{S}{2}) \times K$, gdzie A oznacza liczbę kredytów, które drużyna posiada w momencie zakończenia rozgrywki, S oznacza liczbę kredytów wydanych przez drużynę na rozwój floty robotów, K to współczynnik skalujący wynik, charakterystyczny dla poszczególnych rozgrywek.

5.8 Komendy

Ogólne założenia protokołu komunikacji (łączenie się, logowanie, wysyłanie komend) opisane są w rozdziale "Komunikacja z serwerem". Poniżej znajduje się lista komend dostępnych dla zadania "Fedrunek".

DESCRIBE_WORLD Zwraca parametry świata i wartość współczynnika skalującego wynik.

Parametry: brak

Dane (od serwera):

W pojedynczej linii serwer zwróci trzy wartości:

- N ($N \in \mathbb{N}$) — długość boku kostki asteroidy
- T ($T \in \mathbb{N}$, $1 \leqslant T \leqslant 10$) — czas trwania pojedynczej tury w sekundach
- K ($K \in \mathbb{R}$, $1 \leqslant K \leqslant 5$) — wartość współczynnika skalującego wynik

TIME_TO_MINE Zwraca liczbę tur pozostałych do zakończenia rozgrywki.

Parametry: brak

Dane (od serwera):

Jeśli w momencie zakończenia poprzedniej tury liczba wydobytych jednostek metali wyniosła co najmniej 90% początkowej liczby jednostek asteroidy, funkcja w pojedynczej linii zwróci liczbę tur pozostałych do zakończenia wydobycia:

- L ($L \in \mathbb{N}$, $1 \leqslant L \leqslant 50$)

Jeśli wydobycie nie osiągnęło jeszcze 90%, funkcja zwróci jedną linię postaci:

- UNKNOWN

LIST_STATIONS Zwraca listę punktów skupu metali.

Parametry: brak

Dane (od serwera):

Pierwsza linia:

- D ($D \in \mathbb{N}$) — liczba punktów skupu

Każda kolejna linia opisuje jeden punkt skupu za pomocą oddzielonych spacjami liczb określających poszczególne współrzędne pola z punktem skupu:

- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$)
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$)
- P_Z ($P_Z \in \mathbb{N}$, $1 \leqslant P_Z \leqslant N$)

LIST_WORKSHOPS Zwraca listę warsztatów.

Parametry: brak

Dane (od serwera):

Pierwsza linia:

- W ($W \in \mathbb{N}$) — liczba warsztatów

Każda kolejna linia opisuje jeden warsztat za pomocą oddzielonych spacjami liczb określających poszczególne współrzędne pola z punktem skupu:

- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$)
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$)
- P_Z ($P_Z \in \mathbb{N}$, $1 \leqslant P_Z \leqslant N$)

LIST_SCOUTS Zwraca listę robotów-poszukiwaczy należących do drużyny.

Parametry: brak

Dane (od serwera):

Pierwsza linia:

- R ($R \in \mathbb{N}$) — liczba robotów-poszukiwaczy

Każda kolejna linia opisuje jednego robota za pomocą oddzielonych spacjami liczb:

- ID ($ID \in \mathbb{N}$) — unikalny numer porządkowy robota (poszukiwacze i górnicy numerowani są łącznie, w jednej przestrzeni identyfikatorów)
- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola, w którym znajduje się robot
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola, w którym znajduje się robot
- P_Z ($P_Z \in \mathbb{N}$, $1 \leqslant P_Z \leqslant N$) — współrzędna Z pola, w którym znajduje się robot
- Lev_S ($Lev_S \in \{1, 2, 3\}$) — poziom umiejętności wykonywania sondy, wyznaczający czas trwania tej operacji (początkowo 1)
- Lev_D ($Lev_D \in \{1, 2, 3\}$) — poziom umiejętności wykonywania odwiertu, wyznaczający czas trwania tej operacji (początkowo 1)
- T ($T \in \mathbb{N}$) — liczba tur pozostała do czasu zakończenia obecnie wykonywanej operacji. Wynosi 0 dla robota wolnego (któremu w tej chwili można wydać polecenie) i np. 3 dla robota, któremu właśnie zlecono wykonanie operacji trwającej trzy tury.

LIST_MINERS Zwraca listę robotów-górników należących do drużyny.

Parametry: brak

Dane (od serwera):

Pierwsza linia:

- R ($R \in \mathbb{N}$) — liczba robotów-górników

Każda kolejna linia opisuje jednego robota za pomocą oddzielonych spacjami liczb:

- ID ($ID \in \mathbb{N}$) — unikalny numer porządkowy robota (poszukiwacze i górnicy numerowani są łącznie, w jednej przestrzeni identyfikatorów)
- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola, w którym znajduje się robot
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola, w którym znajduje się robot
- P_Z ($P_Z \in \mathbb{N}$, $1 \leqslant P_Z \leqslant N$) — współrzędna Z pola, w którym znajduje się robot
- Lev_S ($Lev_S \in \{1, 2, 3\}$) — poziom ładowności, wyznaczający pojemność ładowni górnika (początkowo 1)
- Lev_V ($Lev_V \in \{1, 2, 3\}$) — poziom szybkości, wyznaczający czas trwania drążenia tuneli przez danego górnika (początkowo 1)
- Lev_E ($Lev_E \in \{1, 2, 3\}$) — poziom wydajności, wyznaczający czas trwania wydobycia jednej jednostki metalu przez danego górnika (początkowo 1)
- T ($T \in \mathbb{N}$) — liczba tur pozostała do czasu zakończenia obecnie wykonywanej operacji. Wynosi 0 dla robota wolnego (któremu w tej chwili można wydać polecenie) i np. 3 dla robota, któremu właśnie zlecono wykonanie operacji trwającej trzy tury.

BUY Kupuje nowego robota. Nowy robot będzie dostępny w kolejnej turze we wskazanym warsztacie. W momencie wydania polecenia drużyna musi posiadać co najwyżej 24 roboty.

Parametry:

- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola zawierającego warsztat
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola zawierającego warsztat
- P_Z ($P_Z \in \mathbb{N}$, $1 \leqslant P_Z \leqslant N$) — współrzędna Z pola zawierającego warsztat

- $Kind$ — rodzaj robota, SCOUT (poszukiwacz) lub MINER (górnik)

Dane (od serwera):
W pojedynczej linii serwer zwróci wartośc:

- ID ($ID \in \mathbb{N}$) — unikalny numer porządkowy robota

UPGRADE Podnosi wskazaną umiejętność wskazanego robota na wyższy (maksymalnie trzeci) poziom. Wkazany robot musi znajdować się w polu-warsztacie aż do zakończenia operacji podnoszenia umiejętności. Operacja ta trwa jedną turę.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota
- $Skill$ — podnoszona umiejętność: SCAN (szybkość sondy - tylko poszukiwacz), PUSH (szybkość odwiertu - tylko poszukiwacz), DRILL (szybkość drążenia tunelu - tylko górnik), STORAGE (pojemność ładowni - tylko górnik) lub MINE (szybkość wydobycia - tylko górnik)

MOVE Wydaje robotowi polecenie przejścia do sąsiedniego pola, wskazywanego przy pomocy względnych współrzędnych. Jeśli co najmniej jedno spośród pól: źródłowego i docelowego jest podziemne, pomiędzy polami musi istnieć tunel. Dokładnie jedna ze względnych współrzędnych D_X, D_Y, D_Z musi być niezerowa. Operacja trwa jedną turę.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota
- D_X ($D_X \in -1, 0, 1$) — przesunięcie wzdłuż osi X
- D_Y ($D_Y \in -1, 0, 1$) — przesunięcie wzdłuż osi Y
- D_Z ($D_Z \in -1, 0, 1$) — przesunięcie wzdłuż osi Z

INFO Zwraca informacje o polu, w którym znajduje się robot i sześciu sąsiednich polach.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota

Dane (od serwera):
Operacja zwróci opis siedmiu pól - pola, w którym znajduje się wskazany robot i jego sześciu sąsiadów. Każde pole zostanie opisane w osobnej linii o jednej z postaci:

- NIL — jeśli dane pole nie istnieje (w przypadku pól na powierzchni część ich sąsiadów nie będzie istniała)
- *Route* D_X D_Y D_Z SURFACE — jeśli dane pole jest polem na powierzchni i nie zawiera żadnego budynku
- *Route* D_X D_Y D_Z STATION — jeśli dane pole jest polem na powierzchni i zawiera punkt skupu metali
- *Route* D_X D_Y D_Z WORKSHOP — jeśli dane pole jest polem na powierzchni i zawiera warsztat
- *Route* D_X D_Y D_Z UNDERGROUND $Kind$ M_I M_S M_G — jeśli dane pole jest polem podziemym

W powyższej specyfikacji zmienna *Route* przyjmuje jedną z wartości: FREE (aby przejść do tego pola nie jest potrzebny tunel, albo jest potrzebny i jest już wydrążony), TUNNEL (aby przejść do tego pola należy wydrążyć tunel), zmienne D_X D_Y D_Z oznaczają względne (względem aktualnego pola) współrzędne opisywanego pola, $Kind$ to jeden z ciągów znaków: GROUND, STONE; oznaczających meteriał, z którego zbudowane jest dane podziemne pole (kolejno grunt lub kamień), M_I M_S M_G oznaczają liczbę jednostek poszczególnych metali w danym polu (kolejno żelaza, srebra i złota).

STORAGE Zwraca informację o zawartości ładowni wskazanego górnika.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota

Dane (od serwera):
W pojedynczej linii serwer zwróci wartości:

- M_I ($M_I \in \mathbb{N}$) — liczba jednostek żelaza w ładowni górnika
- M_S ($M_S \in \mathbb{N}$) — liczba jednostek srebra w ładowni górnika
- M_G ($M_G \in \mathbb{N}$) — liczba jednostek złota w ładowni górnika
- F ($F \in \mathbb{N}$) — wolna przestrzeń, tj. liczba jednostek metali, które można jeszcze umieścić w ładowni górnika

DRILL Wydaje górnikowi polecenie wydrążenia tunelu do zadanego sąsiedniego pola, wskazywanego przy pomocy względnych współrzędnych. Wskazany robot musi być górnikiem. Co najmniej jedno spośród pól: źródłowego i docelowego musi być podziemne. Pomiędzy aktualnym a wskazanym polem nie może istnieć tunel. Jeśli tunel pomiędzy daną parą pól nie istnieje, ale jest już wiercony, żądanie zostanie odrzucone. Jeśli przed tą samą turą wiele robotów zgłosi polecenie drążenia tego samego tunelu, zaakceptowany zostanie wniosek który zostanie odebrany przez serwer jako pierwszy. Dokładnie jedna ze względnych współrzędnych D_X, D_Y, D_Z musi być niezerowa.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota
- D_X ($D_X \in -1, 0, 1$) — przesunięcie wzdłuż osi X
- D_Y ($D_Y \in -1, 0, 1$) — przesunięcie wzdłuż osi Y
- D_Z ($D_Z \in -1, 0, 1$) — przesunięcie wzdłuż osi Z

Dane (od serwera):
Serwer odpowie pojedynczą linią o jednej z dwóch postaci:

- DRILLING T — jeśli polecenie drążenia zostało zaakceptowane. Liczba T oznacza liczbę tur, jaką będzie trwało drążenie tunelu
- FORBIDDEN — jeśli polecenie zostało odrzucone, gdyż trwa już drążenie tunelu pomiędzy daną parą pól (być może w wykonaniu robota innej drużyny)

MINE Wydaje górnikowi polecenie wydobycia jednej jednostki metalu z pola, w którym się znajduje. W momencie wydania polecenia górnik musi znajdować się w podziemnym polu z niewydobytą jednostką metalu i mieć wolne miejsce w ładowni. W momencie zaakceptowania polecenia przez serwer jednostka metalu jest natychmiast rezerwowana dla zadanego robota i przestaje być dostępna (a także widoczna) dla pozostałych robotów. Jeśli w polu jest kilka rodzajów złóż, wydobywane jest najcenniejsze dostępne, począwszy od złota a skończywszy na żelazie.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota

Dane (od serwera):
Serwer odpowie pojedynczą linią o jednej z dwóch postaci:

- MINING M T — jeśli polecenie wydobycia zostało zaakceptowane. M przyjmuje postać ciągu znaków IRON, SILVER lub GOLD i oznacza metal, którego jednostka będzie wydobywana. Liczba T oznacza liczbę tur, jaką będzie trwało wydobycie
- VOID — jeśli polecenie zostało odrzucone, gdyż w polu nie ma już wolnych jednostek metali

PUSH Wydaje poszukiwaczowi polecenie wykonania odwiertu. Poszukiwacz musi znajdować się na polu na powierzchni, które znajduje się na dokładnie jednej ścianie asteroidy. Jako liczbę pól do przekopania można podać dowolną wartość od 1 do N-2, ale odwiert zawsze będzie wykonywany w cyklach po trzy pola, o czasie trwania zależnym od poziomu możliwości danego robota. Oznacza to, że odwiert 4 pól będzie trwał tyle samo, co odwiert 6 pól.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota
- K ($K \in \mathbb{N}$, $1 \leqslant K \leqslant N$) — głębokość odwiertu wyrażona liczbą pól, które mają być zbadane

Dane (od serwera):
W pojedynczej linii serwer zwróci wartości:

- PUSHING T — Liczba T oznacza liczbę tur, jaką będzie trwało drążenie odwiertu

YIELD_LAST_PUSH Zwraca wynik ostatniego odwiertu wykonanego przez danego robota-poszukiwacza (niekoniecznie w poprzedniej rundzie). Wskazany robot musi mieć niezerową liczbę wykonanych odwiertów.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota-poszukiwacza

Dane (od serwera):
Pierwsza linia:

- P_X P_Y P_Z K — Współrzędne pola, w którym został wykonany odwiert i jego głębokość mierzona liczbą pól

W K kolejnych liniach opisywane są wyniki wiercenia dla kolejnych pól. Każda z linii ma postać:

- P_X P_Y P_Z $Kind$ M_I M_S M_G

Liczby P_X P_Y P_Z wskazują bezwględne współrzędne opisywanego pola, $Kind$ to jeden z ciągów znaków: GROUND, STONE; oznaczających materiał, z którego zbudowane jest dane podziemne pole (kolejno: grunt lub kamień), M_I M_S M_G oznaczają liczbę jednostek poszczególnych metali w danym polu (kolejno: żelaza, srebra i złota). Wszystkie wyniki opisują stan pola na początku ostatniej tury drążenia odwiertu.

SCAN Wydaje poszukiwaczowi polecenie wykonania sondy. Poszukiwacz musi znajdować się na polu na powierzchni, które znajduje się na dokładnie jednej ścianie asteroidy.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota

Dane (od serwera):
W pojedynczej linii serwer zwróci wartości:

- SCANNING T — Liczba T oznacza liczbę tur, jaką będzie trwało wykonanie sondy

YIELD_LAST_SCAN Zwraca wynik ostatniej sondy wykonanej przez danego robota-poszukiwacza (niekoniecznie w poprzedniej rundzie). Wskazany robot musi mieć niezerową liczbę wykonanych sond.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID robota-poszukiwacza

Dane (od serwera):
Odpowiedź serwera zawiera następujące, oddzielone spacjami wartości:

- P_X — współrzędna X pola, w którym została wykonana sonda
- P_Y — współrzędna Y pola, w którym została wykonana sonda
- P_Z — współrzędna Z pola, w którym została wykonana sonda
- K_G — sumaryczna liczba podziemnych pól zbudowanych z gruntu w przebadanym obszarze
- K_S — sumaryczna liczba podziemnych pól zbudowanych z kamienia w przebadanym obszarze
- M_I — sumaryczna liczba jednostek żelaza w przebadanym obszarze
- M_S — sumaryczna liczba jednostek srebra w przebadanym obszarze
- M_G — sumaryczna liczba jednostek złota w przebadanym obszarze

Wszystkie wyniki odzwierciedlają stan pól na początku ostatniej tury wykonywania sondy.

FINANCIAL_REPORT Zwraca aktualną liczbę kredytów posiadanych przez drużynę i liczbę kredytów wydanych na rozwój floty robotów.

Parametry: brak

Dane (od serwera): W pojedynczej linii serwer zwróci wartości:

- C ($C \in \mathbb{N}$) — liczba posiadanych kredytów
- S ($S \in \mathbb{N}$) — liczba kredytów wydanych na rozwój floty robotów

WAIT Czeka do momentu rozpoczęcia kolejnej tury.

Parametry: brak

Dane (od serwera):

Serwer odpowiada jedną linią postaci:

- WAITING S

gdzie S ($S \in \mathbb{R}, 0 \leqslant S$) oznacza ilość sekund do zakończenia oczekiwania

5.9 Błędy

Zgodnie z opisem w rozdziale "Komunikacja z serwerem", w wypadku podania nieprawidłowego polecenia, serwer odpowiada komunikatem

- '`FAILED` *e msg*'

gdzie *e* to kod błędu, a *msg* — komunikat błędu. W tabeli znajduje się zestawienie błędów, które mogą wystąpić przy wydawaniu poleceń w zadaniu Fedrunek.

kod błędu	komunikat błędu
1	bad login or password
2	unknown command
3	bad format
4	too many arguments
5	internal error, sorry...
6	commands limit reached, next call will force waiting
7	commands limit reached, forced waiting activated
101	incorrect robot identifier
102	inappropriate robot type
103	destination is not neighbour
104	destination is outside the world
105	not enough money
106	robot count limit reached
107	cannot drill there
108	cannot raise this skill anymore
109	no workshop in this field
110	this robot is busy
111	cannot move there
112	out of resources
113	invalid skill type
114	invalid location
115	no push results available
116	no scan results available

5.10 Serwery

Rozgrywki będą odbywać się na trzech serwerach, różniących się między sobą relacją pomiędzy głębokością pól (odległością od powierzchni asteroidy) a dominującym rodzajem materiału i występowaniem złóż poszczególnych rodzajów metali. Przykładowo, na jednym z serwerów *mogłyby* być wykorzystywane plansze, w których blisko powierzchni znajdują się wielkie złoża złota w polach zbudowanych z gruntu, a blisko centrum asteroidy występuje wyłącznie kamień pozbawiony jakichkolwiek złóż.

nazwa	adres:port
A1	universum.dl24:20000
A2	universum.dl24:20001
A3	universum.dl24:20002

5.11 Przykład

Poniżej znajduje się przykładowy zapis komunikacji z serwerem.

klient → serwer	serwer → klient
	`LOGIN`
`login1`	
	`PASS`
`secret`	
	`OK`
`DESCRIBE_WORLD`	
	`OK`
	`128 5 1.000000`
`LIST_MINERS`	
	`OK`
	`2`
	`1 128 64 64 1 1 1 0`
	`2 128 64 64 1 1 1 0`
`INFO 1`	
	`OK`
	`FREE 0 0 0 SURFACE`
	`FREE 0 -1 0 STATION`
	`FREE 0 1 0 SURFACE`
	`FREE 0 0 -1 SURFACE`
	`FREE 0 0 1 SURFACE`
	`NIL`
	`TUNNEL -1 0 0 UNDERGROUND GROUND 0 0 1`
`DRILL 1 -1 0 0`	
	`OK`
	`DRILLING 4`
`DRILL 2 -1 0 0`	
	`FAILED 107 cannot drill there`
`MOVE 2 0 1 0`	
	`OK`

6 Zadanie Wycinanka

6.1 Wprowadzenie

Ciągła ekspansja militarna cywilizacji żukoskoczków pochłania ogromne ilości zasobów i stanowi wielkie wyzwanie dla przemysłu zbrojeniowego. Żukoskoczki doceniają wagę tego problemu i stale pracują nad optymalizacją swojej produkcji zbrojeniowej. Ostatnio dział badań firmy zbrojeniowej Żukar bada zagadnienie optymalnego wycinania gumowych stelaży amortyzujących z większych płatów materiału.

Jako podwykonawcy startujący w przetargu na dostarczenie rozwiązania, waszym zadaniem jest opracowanie prototypu algorytmu, który będzie wyszukiwał w płacie materiału fragmenty odpowiadające kolejnym zamówieniom spływającym do fabryki.

6.2 Model rozgrywki

Wszystkie polecenia wydawane przez drużyny rozpatrywane są w czasie rzeczywistym. Rozgrywka z technicznego punktu widzenia rozgrywa się jednak w turach, które służą wyłącznie ograniczeniu liczby poleceń, które drużyny mogą przekazywać do serwera (por. Komunikacja z serwerem - Ograniczenie liczby komend).

Zakończenie rozgrywki Każda rozgrywka będzie trwała z góry określoną liczbę tur. Po zakończeniu rozgrywki rozpoczyna się kolejna, z nowym płatem materiału.

6.3 Materiał

Płat materiału, z którego mają zostać wycięte stelaże ma postać spójnego, planarnego grafu nieskierowanego bez krawędzi wielokrotnych o N wierzchołkach i M krawędziach. Płat materiału jest stały i znany wszystkim drużynom przez cały czas czas trwania rozgrywki.

6.4 Zamówienia

Pojedyncze zamówienie składa się ze spójnego, planarnego grafu nieskierowanego opisującego strukturę, którą należy wskazać w płacie materiału, oraz wartości zamówienia, określającej priorytet, jaki dla fabryki ma uzyskanie danego elementu (równoważny liczbie punktów, jakie można uzyskać za realizację danego zamówienia).

Zamówienia będą składane w czasie rozgrywki, przed rozpoczęciem poszczególnych tur; przed każdą turą zostanie dodane co najmniej jedno zamówienie.

6.4.1 Realizacja

Aby zrealizować zamówienie, drużyna musi przedstawić jego *rozwiązanie*, czyli takie przyporządkowanie MAP wierzchołków grafu płatu materiału do wierzchołków grafu zamówienia, ża dla każdej krawędzi łączącej wierzchołki a i b w grafie zamówienia, wierzchołki $MAP(a)$ i $MAP(b)$ w grafie płatu materiału są połączone krawędzią.

Nie jest wymagane, aby zachodził warunek dualny, tj. aby dla każdej krawędzi pomiędzy wierzchołkami $MAP(a)$ i $MAP(b)$ w grafie płatu materiału istniała krawędź łącząca wierzchołki a i b w grafie zamówienia (zbędne krawędzie są wycinane w fabryce tą samą techniką, która umożliwia wycięcie dowolnego fragmentu płatu materiału).

Rozgrywka stanowi jedynie symulację procesu produkcyjnego prowadzoną w ramach procedury przetargowej. Fragmenty wskazane przez drużyny w ramach realizacji zamówień **nie są** wycinane z płatu materiału. Dowolny wierzchołek płatu materiału może zostać wielokrotnie wskazany, także przez tą samą drużynę. Każde zamówienie może zostać zrealizowane przez dowolną liczbę drużyn.

Nie ma gwarancji, że wszystkie konstrukcje występujące w zamówieniach będą występować w płacie materiału; innymi słowy - nie ma gwarancji, że wszystkie zamówienia będzie dało się zrealizować.

6.5 Punktacja

Każdemu zamówieniu przyporządkowana jest liczba P - jego wartość wyrażona liczbą punktów. Drużyna, która jako I-ta (licząc od zera) zrealizuje dane zamówienie (poprawnie wskaże żądany graf jako podgraf w grafie wejściowym) otrzymuje liczbę punktów wyrażoną wzorem $P \times (\frac{9}{10})^I$.

Końcowy wynik drużyny w pojedynczej rozgrywce stanowi sumę punktów uzyskanych za realizację poszczególnych zamówień przemnożoną przez współczynnik skalujący wynik K.

6.6 Komendy

Ogólne założenia protokołu komunikacji (łączenie się, logowanie, wysyłanie komend) opisane są w rozdziale "Komunikacja z serwerem". Poniżej znajduje się lista komend dostępnych dla zadania "Wycinanka".

DESCRIBE_WORLD Zwraca opis płatu materiału, parametry rozgrywki i wartość współczynnika skalującego wynik.

Parametry: brak

Dane (od serwera):

Pierwsza linia zawiera oddzielone pojedynczymi spacjami wartości:

- N ($N \in \mathbb{N}$) — liczba wierzchołków grafu materiału
- M ($M \in \mathbb{N}$) — liczba krawędzi grafu materiału
- T ($T \in \mathbb{N}$, $1 \leqslant T \leqslant 10$) — czas trwania pojedynczej tury w sekundach
- K ($K \in \mathbb{R}$, $1 \leqslant K \leqslant 5$) — wartość współczynnika skalującego wynik

W M kolejnych liniach opisywane są krawędzie grafu materiału. Opis każdej krawędzi ma postać pojedynczej linii zawierającej dwie, oddzielone spacjami wartości:

- A ($A \in \mathbb{N}$, $1 \leqslant A \leqslant N$) — pierwszy z wierzchołków połączonych daną krawędzią
- B ($B \in \mathbb{N}$, $1 \leqslant B \leqslant N$, $A \neq B$) — drugi z wierzchołków połączonych daną krawędzią

Wierzchołki grafu materiału numerowane są od 1.

TIME_TO_CUT Zwraca liczbę tur pozostałych do zakończenia rozgrywki.

Parametry: brak

Dane (od serwera):

Serwer zwraca pojedynczą wartość:

- L ($L \in \mathbb{N}$) — liczba tur pozostałych do zakończenia rozgrywki

GET_ORDER_COUNT Zwraca liczbę zamówień złożonych przez fabrykę. Kolejne zamówienia identyfikowane są numerami porządkowymi postaci kolejnych liczb naturalnych, zaczynając od 1.

Parametry: brak

Dane (od serwera):

Pierwsza linia zawiera pojedynczą wartość:

- $LastID$ ($LastID \in \mathbb{N}$) — liczba zamówień złożonych przez fabrykę (wartość różna od zera jest równoważna numerowi porządkowemu ostatniego złożonego zlecenia)

DESCRIBE_ORDER Opisuje zamówienie o wskazanym numerze porządkowym.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer porządkowy zlecenia, które ma zostać opisane

Dane (od serwera):
Pierwsza linia zawiera oddzielone pojedynczymi spacjami wartości:

- V ($N \in \mathbb{N}$) — liczba wierzchołków grafu zamówienia
- E ($M \in \mathbb{N}$) — liczba krawędzi grafu zamówienia
- P ($P \in \mathbb{R}$, $1.0 \leqslant P \leqslant 100.0$) — wartość (początkowa) zamówienia
- C ($C \in \mathbb{N}$) — liczba drużyn, które zrealizowały już dane zamówienie

W N kolejnych liniach opisywane są krawędzie grafu zamówienia. Opis każdej krawędzi ma postać linii zawierającej dwie, oddzielone spacjami wartości:

- A ($N \in \mathbb{N}$, $1 \leqslant A \leqslant V$) — pierwszy z wierzchołków połączonych daną krawędzią
- B ($M \in \mathbb{N}$, $1 \leqslant B \leqslant V$, $A \neq B$) — drugi z wierzchołków połączonych daną krawędzią

Wierzchołki grafu zamówienia numerowane są od 1.

COMMIT_SOLUTION Wysyła do weryfikacji rozwiązanie dla wskazanego zamówienia. Drużyna nie może wskazać zamówienia, które już zrealizowała (ale może wielokrotnie, do skutku, próbować zrealizować zamówienie).

Parametry:

W przypadku zamówienia na stelaż o L wierzchołkach, należy przekazać $L + 2$ parametrów: numer porządkowy realizowanego zamówienia, liczbę wierzchołków grafu realizowanego zamówienia i numery wierzchołków grafu materiału odpowiadające kolejnym wierzchołkom grafu zamówienia:

- ID ($ID \in \mathbb{N}$) — numer porządkowy zlecenia, którego rozwiązanie jest przedstawiane
- S ($S \in \mathbb{N}$) — liczba wierzchołków grafu realizowanego zamówienia
- V_1 ($V_1 \in \mathbb{N}$, $1 \leqslant V_1 \leqslant N$) — wierzchołek grafu materiału, który odpowiada pierwszemu wierzchołkowi grafu zamówienia
- (...)
- V_L ($V_L \in \mathbb{N}$, $1 \leqslant V_L \leqslant N$) — wierzchołek grafu materiału, który odpowiada ostatniemu wierzchołkowi grafu zamówienia

Numery wierzchołków V_1, V_2, ... , V_L muszą być parami różne.
Dane (od serwera):

Po przyjęciu zgłoszenia serwer zweryfikuje, czy wszystkim krawędziom grafu zamówienia odpowiadają krawędzie pomiędzy odpowiednimi wierzchołkami grafu materiału (por. Zamówienia -¿ Realizacja). Jeśli zgłoszone rozwiązanie zostanie zaakceptowane, serwer zwróci pojedynczą linią zawierającą następujące, oddzielone pojedynczymi spacjami wartości:

- ACCEPTED
- P ($P \in \mathbb{R}$) — liczba punktów uzyskana za realizację zamówienia (zależna od początkowej wartości zamówienia i liczby drużyn, które zrealizowały je wcześniej)

W przeciwnym wypadku serwer zwróci pojedynczą linię zawierającą wartość:

- INCORRECT

GET_SCORE Zwraca aktualną liczbę punktów zdobytych przez drużynę (nie przemnożoną przez K).

Parametry: brak

Dane (od serwera):

- P ($P \in \mathbb{R}$) — liczba punktów

WAIT Czeka do momentu rozpoczęcia kolejnej tury.

Parametry: brak

Dane (od serwera):

Serwer odpowiada jedną linią postaci:

- WAITING S

gdzie S ($S \in \mathbb{R}, 0 \leqslant S$) oznacza ilość sekund do zakończenia oczekiwania

6.7 Błędy

Zgodnie z opisem w rozdziale "Komunikacja z serwerem", w wypadku podania nieprawidłowego polecenia, serwer odpowiada komunikatem

- '`FAILED` *e msg*'

gdzie *e* to kod błędu, a *msg* — komunikat błędu. W tabeli znajduje się zestawienie błędów, które mogą wystąpić przy wydawaniu poleceń w zadaniu Fedrunek.

kod błędu	komunikat błędu
1	bad login or password
2	unknown command
3	bad format
4	too many arguments
5	internal error, sorry...
6	commands limit reached, next call will force waiting
7	commands limit reached, forced waiting activated
101	incorrect order identifier
102	you have already answered this order
103	your solution contains incorrect vertex id
104	your solution contains duplicate vertex id
105	the size of your solution is incorrect

6.8 Serwery

Rozgrywki będą odbywać się na jednym serwerze.

nazwa	adres:port
B1	universum.dl24:20003

6.9 Przykład

Poniżej znajduje się przykładowy zapis komunikacji z serwerem.

klient → serwer	serwer → klient
	`LOGIN`
`login1`	
	`PASS`
`secret`	
	`OK`
`DESCRIBE_WORLD`	
	`OK`
	`6 6 10 1.000000`
	`1 2`
	`4 2`
	`2 3`
	`3 4`
	`3 6`
	`4 5`
`GET_ORDER_COUNT`	
	`OK`
	`2`
`DESCRIBE_ORDER 2`	
	`OK`
	`4 3 10.000000 0`
	`1 4`
	`2 4`
	`3 4`
`COMMIT_SOLUTION 2 4 1 1 4 2`	
	`FAILED 104 your solution contains duplicate vertex id`
`COMMIT_SOLUTION 2 4 1 3 4 2`	
	`OK`
	`ACCEPTED 7.500000`

7 Zadanie Osada

7.1 Wprowadzenie

Rok temu mieliście okazję pomóc żukoskoczkom w opanowaniu niestabilnej sytuacji na świeżo podbitej planecie CFK - pokrytej siecią podziemnych tuneli zamieszkanych przez krwiożercze potwory - coory. Niestety, pomimo szeroko zakrojonej akcji wyłapywania coor krążących po tunelach, żukoszkoczkom nie udało się wyeliminować zagrożenia. Udało się za to co innego - skuteczne rozwścieczenie ocalałych z łapanki coor.

Jako wojskowi dowódcy spędziliście ostatnie tygodnie koordynując akcję wyłapywania coor w jednej z niewielu osad, które żukoskoczkom udało się zbudować na planecie. Pewnego dnia względny spokój przerwał nagle porucznik wywiadu, który wpadł do pokoju oficerskiego krzycząc *Nadchodzą!*. Wszystko wskazuje na to, że coory opuściły tunele i wyszły na powierzchnię - złe i groźne. Generał Żukk zakończył przeprowadzoną naprędce odprawę słowami: *Obrońcie osadę - lub zgińcie próbując.*

W zadaniu będziecie dowodzić zespołem najlepszych komandosów, jakimi dysponuje armia żukoskoczków. Ramię w ramię z innymi drużynami zmierzycie się z hordą coor i wyzwaniem, jakie postawił przed Wami Generał - obronić osadę. Za wszelką cenę i walcząc do ostatniego żołnierza.

7.2 Model rozgrywki

Rozgrywka odbywa się w turach równej długości. W walce pomiędzy żukoskoczkami i coorami na zmianę przypadają tury, w których akcje wykonują żukoskoczki i tury, w których akcje wykonują coory. Po każdej turze, w której akcje wykonują coory, a bezpośrednio przed turą, w której akcje wykonują żukoskoczki, drużyny komunikują się z serwerem i wydają polecenia, które będą wykonywane w czasie kolejnej tury.

Zakończenie rozgrywki Każda rozgrywka będzie trwała z góry określoną liczbę tur. Ewentualna śmierć wszystkich komandosów jednej lub wielu drużyn nie wpływa na moment zakończenia rozgrywki.

Osady W pojedynczej osadzie (tj. na danym układzie pól) zostanie rozegranych od jednej do kilku rozgrywek (jedna po drugiej, po czym rozgrywki nie powrócą już do danej osady), co odpowiada odpieraniu kolejnych ataków coor na tę samą osadę. Pomiędzy rozgrywkami (także pomiędzy kolejnymi rozgrywkami w tej samej osadzie), wszystkie parametry potyczki (początkowy skład zespołu komandosów, przypisany maszt sztandarowy, natężenie ataków coor i ich strategie) mogą się zmieniać.

7.3 Świat

Polem bitwy będzie osada żukoskoczków reprezentowana przez kwadratową, regularną siatkę $N \times N$ pól rozmieszczonych w N rzędach po N pól każdy. Każe pole zbudowane jest z określonego rodzaju terenu.

Teren Na planszy wyróżniamy następujące rodzaje terenu:

- **Stabilny** - wolne pole, na którym można budować umocnienia. Pole dostępne zarówno dla żukoskoczków jak i dla coor (za wyjątkiem pola z masztem sztandarowym).
- **Grząski** - wolne pole, na którym nie można budować umocnień. Pole dostępne zarówno dla żukoskoczków jak i dla coor.
- **Zabudowany** - pole niedostępne ani dla żukoszkoczków ani dla coor.

Maszty sztandarowe W niektórych polach planszy wybudowane są maszty sztandarowe, służące do wywieszania bojowych proporców poszczególnych zespołów. Tylko coory mogą wejść do tych specjalnych pól (tylko coory potrafią wdrapać się na wysoki maszt).

Sąsiedztwo pól W tym zadaniu definiujemy *sąsiednie pola* jako zbiór (ośmiu) pól dzielących krawędź lub wierzchołek z rozważanym.

7.4 Zespół komandosów

7.4.1 Profesje

Zespół komandosów, którym będziecie dowodzić, będzie składał się z trzech rodzajów żołnierzy:

- **Szturmowiec** - walczy z coorami w starciu bezpośrednim
- **Snajper** - atakuje coory bronią dystansową
- **Inżynier** - buduje pułapki i umocnienia

Szturmowiec Szturmowiec w każdej turze może przesunąć się do jednego z sąsiednich pól. Jeśli na polu, do którego przesunie się szturmowiec będzie znajdowała się coora, dojdzie do walki, którą szturmowiec wygra (tym samym zabijając coorę) z prawdopodobieństwem 90%, i w takim wypadku na początku następnej tury będzie już w polu docelowym. W przeciwnym wypadku szturmowiec zginie, a ojczyzna będzie głęboko wdzięczna za jego bohaterską ofiarę.

Snajper Snajper w każdej turze może przesunąć się do jednego z sąsiednich pól. Jeśli na polu, do którego przesunie się snajper będzie znajdowała się coora, dojdzie do walki, w której snajper zginie z prawdopodobieństwem 100% (i także w tym wypadku ojczyzna będzie głęboko wdzięczna). Snajper może także (zamiast przejścia do nowego pola) zaatakować z dystansu dowolną coorę w zasięgu wzroku (por. sekcja **Zasięg wzroku**), w takim wypadku coora zostaje zabita z prawdopodobieństwem 70%.

Inżynier Inżynier w każdej turze może przesunąć się do jednego z sąsiednich pól. Jeśli na polu, do którego przesunie się inżynie będzie znajdowała się coora, dojdzie do walki, w której inżynier zginie z prawdopodobieństwem 100% (i także w tym wypadku ojczyzna będzie głęboko wdzięczna). Inżynier może także (zamiast przejścia do nowego pola) (wy)budować konstrukcje:

- **Pułapki** - pułapka w formie wilczego dołu wybudowana na wolnym polu zabije pierwszą coorę, która na nim stanie (żukoskoczki wszystkich drużyn mogą bezpiecznie poruszać się po polu z pułapką)
- **Wieżyczka** - wieżyczka strzelnicza wybudowana na wolnym polu zwiększa o 1 zasięg wzroku dowolnej jednostki, która się w niej znajduje

Czas wybudowania pułapki (liczony w turach) wynosi $2+P$, gdzie P oznacza liczbę pułapek wybudowanych przez danego inżyniera (inżynierowie szybko męczą się pracą w ciężkich kombinezonach bojowych). Czas wybudowania wieżyczki (liczony w turach) wynosi $(2 + W)^2$, gdzie W oznacza liczbę wieżyczek wybudowanych przez danego inżyniera.

7.4.2 Sztandar zespołu

Każdemu zespołowi przypisany jest tradycyjny bojowy sztandar - duma i symbol danego oddziału. Sztandar umieszczony jest na jednym (przypisanym na stałe do drużyny) z masztów sztandarowych w osadzie, wyznaczającym zgrubnie fragment osady, którego obronę powierzył danemu zespołowi generał Żukk. Dowolna coora, która dostanie się do pola ze sztandarem, zabiera sztandar i nosi go przy sobie do momentu śmierci (z rąk dowolnego zespołu - w takim wypadku sztandar natychmiast wraca na swoje miejsce) lub opuszczenia planszy przez coorę (w takim wypadku sztandar jest bezpowrotnie stracony).

Puste stanowisko sztandaru okrywa hańbą zespół i umniejsza wymiar jego bojowych zasług. Z tego powodu każda coora zabita przez zespół w czasie obecności sztandaru na miejscu liczona jest do punktacji podwójnie, a coora zabita w czasie nieobecności sztandaru - pojedynczo.

7.4.3 Początkowy stan zespołu

W momencie rozpoczęcia rozgrywki wszyscy komandosi należący do drużyny znajdują się w tym samym polu na planszy, sąsiadującym ze sztandarem drużyny. Zespół będzie składać się z od jednego do 25 komandosów poszczególnych profesji. Początkowa liczba komandosów każdej profesji w pojedynczej rozgrywce będzie taka sama dla wszystkich drużyn.

7.4.4 Zasięg wzroku

Domyślne pole widzenia każdej z jednostek ma postać kwadratu o boku 5 wyśrodkowanego na danej jednostce. Jednostka przebywająca w polu z wieżyczką wyjątkowo ma pole widzenia postaci kwadratu o boku 7.

7.5 Coory

W momencie rozpoczęcia rozgrywki plansza będzie wolna od coor. Coory będą wchodzić na planszę i ewentualnie opuszczać ją przez jej krawędzie.

Mechanika ruchu coor W każdej turze obecna na planszy coora może przesunąć się do jednego z sąsiednich pól.

- Jeśli nowe pole jest zajmowane przez nieuruchomioną jeszcze pułapkę inżynierską, coora natychmiast ginie.
- Jeśli nowe pole jest masztem sztandarowym i zawiera sztandar, coora zdejmuje sztandar z masztu i od tego momentu nosi go przy sobie.
- Jeśli nowe pole jest zajmowane przez szturmowca, odbywa się walka, którą szturmowiec wygrywa z prawdopodobieństwem 90%.
- Jeśli nowe pole jest zajmowane przez inżyniera, lub snajpera (w wieżyczce lub nie), inżynier lub snajper natychmiast giną.

Strategie coor Naukowcy żukoskoczków wyodrębnili następujące rodzaje coor, różniące się zachowaniem w czasie bitwy:

- **Siepacze** - coory, które pseudolosowo krążą po planszy do momentu, w którym w ich polu widzenia znajdzie się żukoskoczek lub sztandar. Od tego momentu ścigają go aż do śmierci (celu lub własnej). W przypadku śmierci ściganego żukoskoczka lub zdobycia sztandaru, siepacze powracają do wyszukiwania kolejnego celu.
- **Złodzieje** - coory, które znając lokalizację sztandarów wchodzą na planszę z intencją kradzieży konkretnego (losowo wybranego spośród aktualnie obecnych) sztandaru i próbują go zdobyć do skutku lub własnej śmierci. Po zdobyciu sztandaru starają się opuścić planszę najkrótszą drogą.
- **Maruderzy** - coory, które pseudolosowo poruszają się po polu bitwy starając się unikać jakiejkolwiek walki.

Żukoskoczki patrząc na coorę nie są w stanie określić jej rodzaju.

7.6 Szczegółowe zasady poruszania jednostek

W dowolnym polu może znajdować się wiele żukoskoczków, jak i wiele coor; przy czym aktywna jest wyłącznie jednostka, która zajęła dane pole jako pierwsza. W szczególności jeśli w jednej turze dwa żukoskoczki (tej samej lub różnych drużyn) otrzymają polecenie przesunięcia się do tego samego pola, aktywną jednostką zostanie ten żukoskoczek, którego ruch został zgłoszony (dotarł do serwera) jako pierwszy. W momencie śmierci lub opuszczenia pola przez jednostkę aktywną, aktywną jednostką zostaje natychmiast następny żukoskoczek w kolejce.

Jednostka nieaktywna nie może podejmować żadnych akcji poza przejściem do sąsiedniego pola.

7.6.1 Przykład 1

Sytuacja początkowa Załóżmy, że w pewnym polu przebywa snajper, a w trzech sąsiednich polach znajduje się, kolejno: szturmowiec (szturmowiec A), kolejny szturmowiec (szturmowiec B) i coora.

Przed turą żukoskoczków Poniższa lista zawiera polecenia wydane jednostkom w kolejności, w jakiej dotarły do serwera:

- snajper otrzymuje polecenie zaatakowania coory
- szturmowiec B otrzymuje polecenie przejścia do pola ze snajperem
- szturmowiec A otrzymuje polecenie przejścia do pola ze snajperem

Tura żukoskoczków Snajper chybia. Szturmowcy przesuwają się do pola ze snajperem. W polu znajdują się teraz jednostki (w kolejności pierwszeństwa do bycia jednostką aktywną): snajper, szturmowiec B, szturmowiec A

Tura coor Coora przesuwa się do pola z żukoskoczkami. Stacza walkę ze snajperem, którego zabija od razu. Automatycznie jednostką aktywną staje się szturmowiec B który natychmiast stacza walkę z coorą. Jeśli przegra walkę (do czego dojdzie z prawdopodobieństwem 10%) natychmiast jednostką aktywną zostanie szturmowiec A i odbędzie się jeszcze jedna walka.

7.6.2 Przykład 2

Sytuacja początkowa Załóżmy, że w pewnym polu znajdują się dwie coory, a w dwóch sąsiednych polach znajdują się dwaj szturmowcy (szturmowiec A i szturmowiec B).

Przed turą żukoskoczków Poniższa lista zawiera polecenia wydane jednostkom w kolejności, w jakiej dotarły do serwera:

- szturmowiec A otrzymuje polecenie przejścia do pola z coorami
- szturmowiec B otrzymuje polecenie przejścia do pola z coorami

Tura żukoskoczków Szturmowiec A przechodzi do pola z coorami. Rozpoczyna się walka szturmowca z aktywną coorą, szturmowiec wygrywa tę walkę. Kolejna coora staje się jednostką aktywną i natychmiast walczy ze szturmowcem; szturmowiec wygrywa również tę walkę. Następnie do pola przechodzi szturmowiec B i nie toczy żadnej walki, gdyż wszystkie coory zostały wcześniej zabite przez szturmowca A.

7.7 Punktacja

Końcowy wynik drużyny równy jest liczbie coor zabitych przez zespół przemnożonych przez współczynnik skalujący K (wliczając coory, które wpadły w pułapki wybudowane przez inżynierów danego zespołu), przy czym coory które zginęły w turze, w której przez cały okres jej trwania sztandar zespołu był na swoim miejscu, liczone są podwójnie.

7.8 Komendy

Ogólne założenia protokołu komunikacji (łączenie się, logowanie, wysyłanie komend) opisane są w rozdziale "Komunikacja z serwerem". Poniżej znajduje się lista komend dostępnych dla zadania "Osada".

DESCRIBE_WORLD Zwraca wymiary świata, czas trwania pojedynczej tury, czas trwania rozgrywki w turach, współrzędne pola zawierającego sztandar drużyny i wartość współczynnika skalującego wynik.

Parametry: brak

Dane (od serwera):

Pierwsza linia:

- N ($N \in \mathbb{N}$, $64 \leqslant N \leqslant 512$) — długość boku planszy
- T ($T \in \mathbb{N}$, $1 \leqslant T \leqslant 10$) — czas trwania pojedynczej tury w sekundach
- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola, w którym znajduje się maszt sztandarowy Waszej drużyny
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola, w którym znajduje się maszt sztandarowy Waszej drużyny
- K ($K \in \mathbb{R}$, $1 \leqslant K \leqslant 5$) — wartość współczynnika skalującego wynik

TIME_TO_FIGHT Zwraca liczbę tur pozostałych do zakończenia rozgrywki i informację o tym, czy kolejna rozgrywka odbędzie się w tej samej osadzie.

Parametry: brak

Dane (od serwera):

Serwer zwraca oddzielone spacjami wartości:

- L ($L \in \mathbb{N}$) — liczba tur pozostałych do zakończenia rozgrywki
- $Next$ — jeden z dwóch ciągów znaków: STAY, jeśli kolejna rozgrywka będzie toczyć się w tej samej osadzie lub GO, jeśli kolejna rozgrywka będzie toczyć się w innej osadzie.

MAP Zwraca mapę kwadratowego fragmentu terenu zawierającego 11x11 pól. Zwrócona mapa pochodzi z archiwum żukoskoczków - nie zawiera więc umocnień, masztów sztandarowych, ani - tym bardziej - jednostek obecnych w poszczególnych polach. Wyszukiwanie mapy w archiwum żukoskoczków jest dla nich procesem czasochłonnym, stąd też polecenie można wykonywać tylko raz w ciągu tury. Wskazane pole musi leżeć na planszy.

Parametry:

- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola, które znajdzie się w środku zwróconej mapy
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola, które znajdzie się w środku zwróconej mapy

Dane (od serwera):
Serwer zwraca opis kwadratu zawierającego 11×11 pól w 11 liniach, odpowiadających kolejnym rzędom pól w kolejności rosnących współrzędnych P_Y. Opis pojedynczego rzędu będzie miał postać 11 znaków (jeden za drugim, bez spacji pomiędzy znakami), opisujących kolejne pola rzędu w kolejności rosnących współrzędnych P_X. Pojedyncze pole będzie opisane znakiem `'.'`, `'#'`, `'*'` lub `'/'`, oznaczającym kolejno: pole wolne; pole zabudowane, pole grząskie i pole leżące poza obszarem rozgrywki.

LIST_TROOPS Zwraca listę komandosów należących do drużyny.

Parametry: brak

Dane (od serwera):

Pierwsza linia:

- R ($R \in \mathbb{N}$) — liczba komandosów

Każda kolejna linia opisuje jednego szturmowca za pomocą oddzielonych spacjami wartości

- ID ($ID \in \mathbb{N}$) — unikalny numer porządkowy komandosa
- $Kind$ — jeden z ciągów znaków: STORMTROOPER, SNIPER, BUILDER, oznaczających rodzaj żołnierza
- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola, w którym znajduje się komandos
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola, w którym znajduje się komandos
- C ($C \in \mathbb{N}$) — liczba innych żukoskoczków zajmujących dane pole, które mają pierwszeństwo do bycia jednostką aktywną (0 oznacza, że to właśnie dany komandos jest jednostką aktywną)
- D ($D \in \mathbb{N}$, $0 \leqslant D \leqslant N$) — liczba tur, po których dana jednostka będzie mogła podjąć nową akcję

LOCATE_BANNER Zwraca współrzędne pola, w którym aktualnie znajduje się sztandar.

Parametry: brak

Dane (od serwera):

Jeśli sztandar znajduje się na swoim miejscu pierwsza linia zawiera oddzielone spacjami wartości:

- SAFE
- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola, w którym znajduje się sztandar
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola, w którym znajduje się sztandar

Jeśli sztandar jest w posiadaniu coory pierwsza linia zawiera oddzielone spacjami wartości:

- MISSING
- P_X ($P_X \in \mathbb{N}$, $1 \leqslant P_X \leqslant N$) — współrzędna X pola, w którym znajduje się sztandar
- P_Y ($P_Y \in \mathbb{N}$, $1 \leqslant P_Y \leqslant N$) — współrzędna Y pola, w którym znajduje się sztandar

Jeśli sztandar został wyniesiony poza planszę pierwsza linia zawiera jedną wartość:

- LOST

MOVE Wydaje komandosowi polecenie przejścia do sąsiedniego pola, wskazywanego przy pomocy względnych współrzędnych. Pole docelowe nie może być zabudowane, nie może także zawierać sztandaru. Co najmniej jedna ze względnych współrzędnych D_X, D_Y musi być niezerowa.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID komandosa
- D_X ($D_X \in \{-1, 0, 1\}$) — przesunięcie wzdłuż osi X
- D_Y ($D_Y \in \{-1, 0, 1\}$) — przesunięcie wzdłuż osi Y

SHOOT Wydaje snajperowi polecenie zaatakowania z dystansu coory na sąsiedniem polu, wskazywanego przy pomocy względnych współrzędnych. Jeśli w momencie ataku na wskazanym polu znajduje się kilka coor, celem ataku będzie dokładnie jedna z nich.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID komandosa
- D_X ($D_X \in \{-3, -2, -1, 0, 1, 2, 3\}$) — przesunięcie wzdłuż osi X
- D_Y ($D_Y \in \{-3, -2, -1, 0, 1, 2, 3\}$) — przesunięcie wzdłuż osi Y

BUILD Wydaje inżynierowi polecenie rozpoczęcia budowy wieżyczki lub pułapki na polu, w którym się aktualnie znajduje. Teren danego pola musi być stabilny, pole nie może zawierać żadnej konstrukcji, nie może również w nim trwać budowa. Jeśli w tej samej turze kilku inżynierów otrzyma polecenie rozpoczęcia budowy w tym samym polu, zaakceptowane zostanie to żądanie, które dotrze do serwera jako pierwsze.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID komandosa
- $Kind$ ($Kind \in \{TRAP, TOWER\}$) — rodzaj konstrukcji

Dane (od serwera):
Dokładnie jeden z poniższych napisów:

- BUILDING T — Jeśli żądanie zostało zaakceptowane. Liczba T oznacza liczbę tur, jaką będzie trwała konstrukcja wieżyczki
- REFUSED — jeśli polecenie zostało odrzucone, gdyż trwa już budowa w danym polu (być może w wykonaniu komandosa innej drużyny)

INFO Zwraca informacje o polach w zasięgu wzroku danej jednostki.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer ID komandosa

Dane (od serwera):
W pierwszej linii serwer zwróci pojedynczą wartość:

- C ($C \in \mathbb{N}$) — liczba pól w polu widzenia danej jednostki

Następnie każde pole zostanie opisane w osobnej linii. Opis pojedynczej linii będzie zawierał następujące, oddzielone spacjami wartości:

- D_X ($D_X \in \{-3, -2, -1, 0, 1, 2, 3\}$) — względna współrzędna X opisywanego pola
- D_Y ($D_Y \in \{-3, -2, -1, 0, 1, 2, 3\}$) — względna współrzędna Y opisywanego pola
- $Terrain$ ($Terrain \in \{SOLID, SOGGY, BLOCKED\}$) — rodzaj terenu, odpowiednio: stabilny, grząski lub zabudowany
- $Settlement$ ($Settlement \in \{MAST, TOWER, ACTIVE\_TRAP, USED\_TRAP, CONSTRUCTION, NONE$ — budynek obecny w danej planszy, odpowiednio: maszt sztandarowy; wieżyczka; pułapka która jeszcze nie została urochomiona; pułapka która została uruchomiona; brak umocnień.
- $\#COORS$ ($\#COORS \in \mathbb{N}$) — liczba coor w danym polu
- $\#TROOPS$ ($\#TROOPS \in \mathbb{N}$) — liczba żukoskoczków w danym polu

MEASURE_GLORY Zwraca liczbę coor zabitych przez drużynę (z uwzględnieniem podwójnego liczenia coor zabitych w czasie, kiedy sztandar drużyny był na maszcie sztandarowym).

Parametry: brak

Dane (od serwera):

- P ($P \in \mathbb{N}$) — liczba zabitych coor

WAIT Czeka do momentu rozpoczęcia kolejnej tury.

Parametry: brak

Dane (od serwera):

Serwer odpowiada jedną linią postaci:

- WAITING S

gdzie S ($S \in \mathbb{R}, 0 \leqslant S$) oznacza ilość sekund do zakończenia oczekiwania

7.9 Błędy

Zgodnie z opisem w rozdziale "Komunikacja z serwerem", w wypadku podania nieprawidłowego polecenia, serwer odpowiada komunikatem

- '`FAILED` *e msg*'

gdzie *e* to kod błędu, a *msg* — komunikat błędu. W tabeli znajduje się zestawienie błędów, które mogą wystąpić przy wydawaniu poleceń w zadaniu Osada.

kod błędu	komunikat błędu
1	bad login or password
2	unknown command
3	bad format
4	too many arguments
5	internal error, sorry...
6	commands limit reached, next call will force waiting
7	commands limit reached, forced waiting activated
101	incorrect unit identifier
102	incorrect field coordinates
103	destination is not neighbour
104	target is outside range
105	cannot move there
106	nobody to shoot there
107	cannot build here
108	not an active troop
109	incorrect building type
110	incorrect unit type
111	no more maps during this turn
112	unit already used in this turn

7.10 Serwery

Rozgrywki będą odbywać się na trzech serwerach, różniących się między sobą rodzajem coor (por. Strategie coor) dominujących wśród atakujących jednostek. W kolumnie *strategia dominująca* wskazany jest rodzaj coor, który reprezentować będzie co najmniej 50% jednostek atakujących w każdej rozgrywce na danym serwerze.

nazwa	adres:port	strategia dominująca
C1	universum.dl24:20004	siepacze
C2	universum.dl24:20005	złodzieje
C3	universum.dl24:20006	maruderzy

7.11 Przykład

Poniżej znajduje się przykładowy zapis komunikacji z serwerem.

klient → serwer	serwer → klient
	`LOGIN`
`login1`	
	`PASS`
`secret`	
	`OK`
`DESCRIBE_WORLD`	
	`OK`
	`100 10 5 5 1.000000`
`LIST_TROOPS`	
	`OK`
	`3`
	`1 STORMTROOPER 5 6 0 0`
	`2 BUILDER 5 6 1 0`
	`3 SNIPER 5 6 2 0`
`MAP 5 6`	
	`OK`
	`/.........`
	`/######....`
	`/.....#....`
	`/.........`
	`/.........`
	`/.........`
	`/.........`
	`/.....#....`
	`/######....`
	`/.........`
	`/.........`
`BUILD 2 TOWER`	
	`FAILED 108 not an active troop`
`MOVE 2 5 5`	
	`FAILED 105 cannot move there`
`MOVE 2 6 7`	
	`OK`
`WAIT`	
	`OK`
	`WAITING 8.670000`
`BUILD 2 TOWER`	
	`OK`
	`BUILDING 4`