Zasady i zestaw zadań

Gliwice, 26-27 kwietnia 2011

Spis treści

1 Wstęp

Już trzeci raz spotykamy się na Wydziale Automatyki, Elektroniki i Informatyki Politechniki Śląskiej w Gliwicach na finale konkursu Deadline24.

W tegorocznej edycji przygotowaliśmy trzy zadania, tradycyjnie osadzone w świecie cywilizacji żukoskoczków, ambitnych stworzeń ogarniętych chęcią podboju kosmosu. Jak się przekonamy, wśród żukoskoczków zdarzają się jednostki zainteresowane także innymi rzeczami, co wywołuje oburzenie Ministra Dyscypliny Wewnętrznej (zadanie Rekonstrukcja). Poznamy też planetę CFK-1, zamieszkaną wyłącznie przez coory i spróbujemy je wyłapać (zadanie Łapanka). Wreszcie na prośbę (znanego z eliminacji) Admirała Żukka rozegramy kilka partii gry planszowej w ramach przygotowań do interwencji militarnej w mieście Gachico (zadanie Złodzieje i złodzieje).

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 20 ('\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 z trzech najlepszych wyników (na danym serwerze).

Ranking Ranking zawodów jest rankingiem sumy punktów rankingowych poszczególnych drużyn ze wszystkich serwerów wszystkich zadań.

4 Sytuacje awaryjne

W razie wystąpienia sytuacji, w której nie wszystkie drużyny są 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 Łapanka

5.1 Wprowadzenie

Cywilizacja żukoskoczków przystąpiła do kolonizacji planety CFK-1. Podbój egzotycznych zakątków Wszechświata nie jest jednak zadaniem prostym - wśród wielu problemów, z jakimi trzeba sobie radzić w procesie kolonizacji, jest także radzenie sobie z rdzennymi mieszkańcami podbijanych planet. W tym wypadku sprawa jest stosunkowo prosta - na CFK-1 nie mieszkają żadne stworzenia odpowiadające definicji istot inteligentnych. Cała planeta zamieszkana jest wyłącznie przez coory.

Coory są stworzeniami podobnymi do ziemskich kur - to dwunożne ptaki nieloty. Niestety, w przeciwieństwie do ziemskich kur, są wyjątkowo agresywne, śmiertelnie niebezpieczne i trudne do złapania. Na domiar złego, CFK-1 pokryte jest siecią naturalnych tuneli, w których coory czują się jak u siebie, w przeciwieństwie do żukoskoczków. W związku z tym żukoskoczki postanowiły skierować na CFK-1 szwadrony swoich najlepszych komandosów z trudną misją wyłapania wszystkich coor żyjących w tunelach CFK-1 w ramach operacji "Łapanka".

Jako dowódcy jednego z oddziałów komandosów musicie kierować działaniami zespołu tak, aby powrócić na macierzystą planetę z jak największą liczbą coor.

5.2 Opis problemu

5.2.1 Coory

Tunele Coory zamieszkują sieć tuneli złożoną z N pól, nazywanych grotami. Przechodzenie pomiędzy grotami umożliwia coorom M korytarzy, z których każdy łączy dwie spośród grot planety.

Tury i podróże Życie coor dzieli się na tury - każdą z nich coory spędzają w grotach odpoczywając, drzemiąc czy jedząc. Kiedy tylko tura się kończy, coory w zbiorowym przypływie chęci do podróżowania wybierają jeden z korytarzy dostępnych w grocie, w której się znajdują, i przechodzą do nowej groty. Coory przemieszczają się na tyle szybko, że czas potrzebny na przejście korytarzem jest pomijalny - na początku następnej tury każda coora jest już w nowej grocie.

Wybór korytarza Coory z równym prawdopodobieństwem będą wybierały każdy spośród dostępnych w danej grocie korytarzy. Każda coora podejmuje decyzję niezależnie od pozostałych coor w tej samej grocie. Jeśli z jakiegoś powodu w danej grocie nie będzie dostępnych korytarzy (może się tak zdarzyć, patrz następna sekcja), coory znajdujące się w tej grocie pozostają w niej do następnej tury (lub dłużej).

Coorza imigracja Należy liczyć się z tym, że w miarę wyłapywania coor zamieszkujących system tuneli, zaczną się w nim pojawiać nowe coory przybywające z gęściej zamieszkanych (niż tunele) obszarów planety. Przybywające coory będą wybierać z równym prawdopodobieństwem każdą spośród grot systemu tuneli.

5.2.2 Łapanka

Ponieważ zejście do tuneli pełnych coor byłoby śmiertelnie niebezpieczne, Twojemu zespołowi pozostaje prowadzić łapankę z powierzchni. Macie do dyspozycji jedną klatkę na coory, echosondę podającą lokalizacje coor, oraz kombajn wiertniczy pozwalający na różne sposoby wpływać na podziemny świat coor - w szczególności umieszczać klatkę w grotach i wydobywać ją na powierzchnię wraz ze złapanymi coorami.

Energia Zasadniczym ograniczeniem dla działań Waszego zespołu jest ilość dostępnej energii elektrycznej. Każdej tury ogniwa słoneczne będą dostarczały Wam nowych jednostek energii, które możecie wykorzystywać do prowadzenia łapanki. Niewykorzystane jednostki energii mogą być akumulowane przez kolejne tury bez ograniczeń, natomiast liczba jednostek uzyskiwanych w każdej turze (która zależy przecież od pogody!) nie jest (to znaczy nie musi być) stała, ani określona z góry, ale w każdej turze będzie identyczna dla wszystkich drużyn.

Operacje Kluczowym elementem łapanki jest klatka na coory. Wasz zespół ma dokładnie jedną klatkę, którą może umieścić w dowolnej (nie zajętej w danym momencie przez inny zespół) grocie, a później, na tych samych zasadach, przestawiać ją do nowych grot. Macie także do dyspozycji wachlarz technik służących do wpływania na ruch kur w tunelach.

- **Klatka** - Ustawienie klatki w grocie (po raz pierwszy, albo przestawienie klatki do nowej groty).
- **Dwukierunkowa blokada korytarza** - Umieszczenie w wybranym korytarzu środka odstraszającego coory powoduje, że po zakończeniu tury korytarz nie będzie dostępny dla coor w grotach po obu stronach korytarza (działa jednorazowo).
- **Jednokierunkowa blokada korytarza** - Umieszczenie w wybranym korytarzu stracha na coory widocznego tylko z jednej strony powoduje, że po zakończeniu tury korytarz będzie dostępny tylko dla coor w grocie na wybranym końcu korytarza.
- **Granat hukowy** - Wrzucenie granatu hukowego do groty powoduje, że po zakończeniu tury wszystkie korytarze prowadzące do danej groty będą niedostępne (ale tylko w jedną stronę, to znaczy po zakończeniu tury żadna coora nie wejdzie do wybranej groty, ale mogą z niej wyjść te, które już się w niej znajdują).
- **Ziarno** - Umieszczenie w grocie karmy dla coor powoduje, że po zakończeniu tury wszystkie korytarze prowadzące z danej groty będą niedostępne (ale tylko w jedną stronę, to znaczy po zakończeniu tury żadna coora nie wyjdzie z wybranej groty, ale mogą do niej trafić nowe coory).

5.2.3 Punktacja

Wynik Waszej drużyny będzie równy łącznej liczbie złapanych przez Was coor.

5.3 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 "Łapanka".

TUNNELS Opisuje strukturę tuneli na planecie CFK-1.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- N ($N \in \mathbb{N}$) — liczba grot
- M ($M \in \mathbb{N}$) — liczba korytarzy

Druga linia:

- Ciąg M oddzielonych spacjami opisów poszczególnych korytarzy, z których każdy ma postać A-B, ($A \in \mathbb{N}$, $B \in \mathbb{N}$, $1 \leqslant A \leqslant N$, $1 \leqslant B \leqslant N$, $A \neq B$) gdzie A i B oznaczają numery grot, które łączy dany korytarz

COORS Opisuje lokalizacje poszczególnych coor.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- S ($S \in \mathbb{N}$) — liczba nie złapanych coor obecnych w systemie tuneli

Druga linia:

- Ciąg S oddzielonych spacjami liczb X_i ($X_i \in \mathbb{N}$, $1 \leqslant X_i \leqslant N$) — X_i oznacza grotę, w której znajduje się i-ta coora

ENERGY Zwraca liczbę dostępnych jednostek energii elektrycznej.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- N ($N \in \mathbb{N}$) — liczba dostępnych jednostek energii elektrycznej

PRICES Opisuje koszt poszczególnych operacji w jednostkach energii elektrycznej.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- $CagePrice$ ($CagePrice \in \mathbb{N}$) — koszt ustawienia / przestawienia klatki (komenda CAGE)
- $BBlockPrice$ ($BBlockPrice \in \mathbb{N}$) — koszt ustawienia dwukierunkowej blokady korytarza (komenda BBLOCK)
- $MBlockPrice$ ($MBlockPrice \in \mathbb{N}$) — koszt ustawienia jednokierunkowej blokady korytarza (komenda MBLOCK)
- $GrenadePrice$ ($GrenadePrice \in \mathbb{N}$) — koszt użycia granatu hukowego (komenda GRENADE)
- $GrainPrice$ ($GrainPrice \in \mathbb{N}$) — koszt użycia ziarna (komenda GRAIN)

CAGE Ustawia klatkę w wybranej grocie. Jeśli operacja się powiedzie, klatka złapie wszystkie coory, które będą znajdować się w danej grocie od początku następnej tury do momentu przestawienia klatki w kolejne miejsce. Aby wykonać operację drużyna musi posiadać odpowiednią liczbę jednostek energii elektrycznej, a grota musi być wolna, tj. nie zajęta przez żadną inną drużynę. Tego ostatniego warunku nie da się sprawdzić inaczej niż próbując ustawić w danym miejscu klatkę; jeśli grota okaże się zajęta serwer odpowie odpowiednim komunikatem a jednostki energii elektrycznej nie zostaną zużyte.

Parametry:

- V ($V \in \mathbb{N}$, $1 \leqslant V \leqslant N$) — numer groty

BBLOCK Ustawia dwukierunkową blokadę w korytarzu łączącym wybrane groty. Aby wykonać operację drużyna musi posiadać odpowiednią liczbę jednostek energii elektrycznej. Podane groty muszą być połączone korytarzem.

Parametry:

- V ($V \in \mathbb{N}$, $1 \leqslant V \leqslant N$) — numer groty na jednym z końców blokowanego korytarza
- W ($W \in \mathbb{N}$, $1 \leqslant W \leqslant N$) — numer groty na drugim z końców blokowanego korytarza

MBLOCK Ustawia jednokierunkową blokadę w korytarzu łączącym wybrane groty. Aby wykonać operację drużyna musi posiadać odpowiednią liczbę jednostek energii elektrycznej. Podane groty muszą być połączone korytarzem.

Parametry:

- V ($V \in \mathbb{N}$, $1 \leqslant V \leqslant N$) — numer groty na jednym z końców blokowanego korytarza
- W ($W \in \mathbb{N}$, $1 \leqslant W \leqslant N$) — numer groty na drugim z końców blokowanego korytarza
- D ($D \in \{0, 1\}$) — kierunek blokady, 0 oznacza że po zakończeniu tury coory będą mogły przejść danym korytarzem tylko z groty o mniejszym numerze do groty o większym numerze. 1 przeciwnie, oznacza że po zakończeniu tury coory będą mogły przejść danym korytarzem tylko z groty o większym numerze do groty o mniejszym numerze.

GRENADE Używa granatu hukowego w wybranej grocie. Aby wykonać operację drużyna musi posiadać odpowiednią liczbę jednostek energii elektrycznej.

Parametry:

- V ($V \in \mathbb{N}$, $1 \leqslant V \leqslant N$) — numer groty

GRAIN Używa ziarna w wybranej grocie. Aby wykonać operację drużyna musi posiadać odpowiednią liczbę jednostek energii elektrycznej.

Parametry:

- V ($V \in \mathbb{N}$, $1 \leqslant V \leqslant N$) — numer groty

SCORE Podaje liczbę coor złapanych jak dotąd przez Waszą drużynę.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

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

WAIT Czeka do momentu rozpoczęcia kolejnej tury.

Parametry:brak

Dane (od serwera):

Serwer odpowiada jedną linią zawierającą (oddzielone spacjami) napis

- WAITING

oraz liczbę

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

TURN Informuje o numerze aktualnie trwającej tury.

Parametry:brak

Dane (od serwera):

- T ($T \in \mathbb{N}, 0 \leqslant T$) — numer aktualnie trwającej tury

5.3.1 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 Łapanka.

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
100	bad cave identifier
101	there is no tunnel between given caves
102	not enough energy
103	cave is already occupied
104	you have already set cage in this cave
105	bad direction value
106	bad turn number

5.4 Serwery

Poniżej znajduje się zestawienie serwerów wraz z wartościami parametrów rozgrywki

- **TurnDuration** - długość tury w sekundach
- **GraphType** - poglądowa struktura tuneli
- **MaxCalls** - maksymalna liczba poleceń w turze

nazwa	adres:port	**TurnDuration**	**GraphType**	**MaxCalls**
A1	universum.dl24:20000	1	STAR	50
A2	universum.dl24:20001	1	TREELIKE	30
A3	universum.dl24:20002	2	RANDOM	100

5.5 Przykład

Poniżej znajduje się przykładowy zapis komunikacji z serwerem.

klient → serwer	serwer → klient
	`LOGIN`
`login1`	
	`PASS`
`secret`	
	`OK`
`TUNNELS`	
	`OK`
	`4 5`
	`1-2 2-3 1-4 4-3 3-1`
`COORS`	
	`OK`
	`1`
	`3`
`CAGE 1`	
	`OK`
`WAIT`	
	`OK`
	`WAITING 4.36`
	`OK`
`SCORE`	
	`OK`
	`1`
`COORS`	
	`OK`
	`0`

6 Zadanie Rekonstrukcja

6.1 Wprowadzenie

Cywilizacja żukoskoczków słynie z solidarności, ambicji, inteligencji i odwagi, które zapewniły jej dominację w macierzystych okolicach Wszechświata. W tym większe zakłopotanie wprawiają władze czarne owce w zdrowej tkance społeczeństwa - to znaczy wszelkiej maści bumelanci, dezerterzy i wagarowicze. Ministerstwo Propagandy robi co może, żeby ukrywać przed światem przypadki występków przeciwko ogólnie pojętemu porządkowi społecznemu, ale najnowszej katastrofy ukryć nie sposób!

Otóż na jednej ze skolonizowanych planet urodził się wyjątkowo niesforny żukoskoczek. Nie interesował go podbój Wszechświata, nie marzył o karierze pilota, żołnierza, łowcy snejków, ani nawet urzędnika państwowego. Całe dnie spędzał na błahych rozrywkach w rodzaju układania puzzli albo programowania. Kiedy wieść o tym dotarła do Ministersta Dyscypliny Wewnętrznej, sam Minister osobiście skonfiskował wszystkie puzzle należące do żukoskoczka i nakazał mu natychmiastową rekrutację na studia w Akademii Lotnictwa Inwazyjnego. Żukoskoczek jednak nigdy nie dotarł do Akademii - targany zranioną miłością do puzzli, zawiedziony represyjną postawą państwa, poprzysiągł zemstę na społeczeństwie, które go odrzuciło.

I oto pewnego dnia komputery cywilizacji żukoskoczków opanował tajemniczy wirus, który zamienia wszystkie obrazy znalezione na dysku komputera w puzzle, tj. dzieli je na wiele prostokątnych fragmentów, które dodatkowo obraca o 0, 90, 180 lub 270 stopni. *Toż to zwyczajny terrorysta!* - miał wg doniesień prasowych wykrzyknąć Minister Dyscypliny Wewnętrznej na wieść o wirusie.

Ministerstwo Uprawiania Nauki powołało zespół ekspertów mających opracować wydajną metodę rekonstrukcji zniszczonych przez wirus obrazów. Wasza drużyna została zaproszona do udziału w programie.

6.2 Opis problemu

Ministerstwo przygotowało eksperyment polegający na kontrolowanej rekonstrukcji specjalnie opracowanych zestawów (to znaczy takich, które obrazy źródłowe są znane naukowcom Ministerstwa), co pozwoli na weryfikację postępów drużyn.

6.2.1 Obrazy

Przedmiotem rekonstrukcji będą obrazy zapisane przy pomocy formatu JPG. Każdy taki obraz został podzielony na kwadratowe fragmenty regularną siatką o pewnej liczbie kolumn i wierszy. Następnie kwadraty zostały losowo poobracane o różne wielokrotności 90 stopni i losowo przepermutowane na siatce

(to znaczy, że dla dowolnego fragmentu obrazu źródłowego, każda docelowa pozycja na siatce i każda orientacja jest równie prawdopodobna). Każdy taki obraz powstały po obrotach i permutacji fragmentów na siatce został zapisany w formacie JPG i umieszczony w zaszyfrowanym archiwum.

Archiwa z obrazami zostały dołączne do tego zbioru zadań na nośniku przenośnej pamięci masowej (pomarańczowy pendrive).

6.2.2 Przebieg rekonstrukcji

Rekonstrukcja będzie prowadzona wspólnie przez wszystkie drużyny zaproszone do udziału w eksperymencie. Eksperyment będzie przebiegał w rundach, po jednej na obraz. Każda runda zacznie się w momencie ujawnienia hasła do kolejnego archiwum z kawałkami i będzie trwała do momentu pełnej rekonstrukcji źródłowego obrazu.

Kalendarz rund Początki kolejnych rund zaplanowane są równo co ustaloną liczbę minut przez cały czas trwania konkursu. Ze względu na to, że rekonstrukcja każdego z obrazów trwa do skutku, kalendarz może się przesuwać, ale tylko do przodu. To znaczy, i-ta runda rozpocznie się o czasie

max(zaplanowany czas rozpoczęcia, czas zakończenia poprzedniej rundy)

Tury Sama rekonstrukcja będzie przebiegała w turach - w czasie jednej tury każdej drużynie będzie przysługiwało prawo do jednej próby dopasowania fragmentu rekonstruowanego obrazu, tj. wskazania zajmowanej przez dany kawałek pozycji na siatce oryginalnego obrazu i kąta, o jaki należy ten kawałek obrócić. Jeśli próba będzie udana, drużyna otrzyma punkt, a wszystkie drużyny będą mogły uzyskać informację o lokalizacji i orientacji dopasowanego fragmentu.

Pozycja na siatce W czasie rekonstrukcji będziecie znali liczbę kolumn i wierszy regularnej siatki, która podzieliła wejściowy obraz na kwadraty. Kolumny numerujemy w kolejności od 1 do K w kolejności od lewej strony do prawej. Wiersze numerujemy od 1 do W w kolejności od góry do dołu obrazu. Wskazanie pozycji na siatce oznacza wskazanie numeru kolumny i numeru wiersza zajmowanych przez dany fragment na siatce w obrazie źródłowym.

Kąt obrotu Dopasowując fragment należy także określić jego orientację, to znaczy podać o jaki kąt (spośród zbioru możliwości 0, 90, 180, 270) należy obrócić dany fragment obrazu z archiwum zgodnie z ruchem wskazówek zegara z perspektywy osoby stojącej na wprost obrazu.

6.2.3 Punktacja

Każde prawidłowe dopasowanie fragmentu będzie nagradzane $\frac{1}{C}$ punktami, gdzie C oznacza liczbę drużyn, które jednocześnie, tj. w tej samej turze dopasowały dany fragment. Wynik Waszej drużyny będzie równy łącznej ilości punktów za dopasowane fragmenty we wszystkich rundach rozgrywki.

6.3 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 "Rekonstrukcja".

INFO Udostępnia podstawowe informacje o rekonstruowanym obrazie.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- *name* (*name* zawiera co najwyżej 50 znaków i są to wyłącznie małe i duże litery alfabetu angielskiego, kropki oraz cyfry łacińskie) — nazwa archiwum z kawałkami rekonstruowanego obrazu
- *pass* (*pass* zawiera co najwyżej 50 znaków i są to wyłącznie małe i duże litery alfabetu angielskiego oraz cyfry łacińskie) — jeśli trwa rekonstrukcja obrazu *pass* jest hasłem do rozpakowania odpowiedniego archiwum, w przeciwnym wypadku (tj. jeśli trwa oczekiwanie na rozpoczęcie nowej rundy - por. paragraf **Kalendarz rund**) ma postać "[WillShowOnNextTurn]".
- K ($K \in \mathbb{N}$) — liczba kolumn siatki obrazu
- W ($W \in \mathbb{N}$) — liczba wierszy siatki obrazu

GUESS Próbuje dopasować jeden z jeszcze niedopasowanych fragmentów we wskazane miejsce na siatce obrazu. Drużyna może w ciągu tury wydać wiele poleceń GUESS, w takim wypadku wykonane zostanie ostatnie polecenie wydane w danej turze. Aby sprawdzić, czy dopasowanie się powiodło, należy w kolejnej turze użyć polecenia CHANGES lub POINTS

Parametry:

- k_p ($k_p \in \mathbb{N}, 1 \leqslant k_p \leqslant K$) — numer kolumny dopasowywanego fragmentu na siatce obrazu z archiwum
- w_p ($w_p \in \mathbb{N}, 1 \leqslant w_p \leqslant W$) — numer wiersza dopasowywanego fragmentu na siatce obrazu z archiwum
- k_{org} ($l_{org} \in \mathbb{N}, 1 \leqslant k_{org} \leqslant K$) — numer kolumny wskazywanego miejsca na siatce oryginalnego obrazu
- w_{org} ($w_{org} \in \mathbb{N}, 1 \leqslant w_{org} \leqslant W$) — numer wiersza wskazywanego miejsca na siatce oryginalnego obrazu
- *ROT* (*ROT* $\in \{0, 90, 180, 270\}$, **opcjonalny**) — kąt obrotu fragmentu (czyli kąt o jaki należy obrócić fragment z archiwum, aby otrzymać oryginalną orientację, zgodne ze specyfikacją z paragrafu **Kąt obrotu**, domyślnie 0)

CHANGES Informuje o fragmentach dopasowanych w wybranej turze. Liczba ostatnich zakończonych tur, o które można pytać jest ograniczona odpowiednim parametrem świata (por. sekcja **Serwery**). W przypadku braku parametru polecenie domyślnie zwróci zmiany dotyczące poprzedniej tury.

Parametry:

- T ($T \in \mathbb{N}$, **opcjonalny**)

Dane (od serwera):
Pierwsza linia:

- *name* (*name* zawiera co najwyżej 50 znaków i są to wyłącznie małe i duże litery alfabetu angielskiego, kropki oraz cyfry łacińskie) — nazwa archiwum z kawałkami obrazu rekonstruowanego w danej turze

Druga linia:

- L ($L \in \mathbb{N}$) — liczba fragmentów pozostałych do dopasowania po danej turze

Trzecia linia:

- X ($X \in \mathbb{N}$) — liczba fragmentów dopasowanych w danej rundzie

Każda kolejna linia opisuje jeden dopasowany w danej rundzie fragment za pomocą oddzielonych spacjami liczb:

- k_p ($k_p \in \mathbb{N}, 1 \leqslant k_p \leqslant K$) — numer kolumny dopasowanego fragmentu na siatce obrazu z archiwum
- w_p ($w_p \in \mathbb{N}, 1 \leqslant w_p \leqslant W$) — numer wiersza dopasowanego fragmentu na siatce obrazu z archiwum
- k_{org} ($k_{org} \in \mathbb{N}, 1 \leqslant k_{org} \leqslant K$) — numer kolumny prawidłowego miejsca na siatce oryginalnego obrazu
- w_{org} ($w_{org} \in \mathbb{N}, 1 \leqslant w_{org} \leqslant W$) — numer wiersza prawidłowego miejsca na siatce oryginalnego obrazu
- ROT ($ROT \in \{0, 90, 180, 270\}$) — kąt obrotu fragmentu (czyli kąt o jaki należy obrócić fragment z archiwum, aby otrzymać oryginalną orientację, zgodne ze specyfikacją z paragrafu **Kąt obrotu**)
- ($Count$) ($Count \in \mathbb{N}$) — liczba drużyn, które dopasowały dany fragment (podana w nawiasach okrągłych)

WAIT Czeka do momentu rozpoczęcia kolejnej tury.

Parametry:brak

Dane (od serwera):

Serwer odpowiada jedną linią zawierającą (oddzielone spacjami) napis

- WAITING

oraz liczbę

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

TURN Informuje o numerze aktualnie trwającej tury.

Parametry:brak

Dane (od serwera):

- T ($T \in \mathbb{N}, 0 \leqslant T$) — numer aktualnie trwającej tury

POINTS Informuje o ilości punktów zgromadzonych przez drużynę.

Parametry:brak

Dane (od serwera):

- P ($P \in \mathbb{R}, 0 \leqslant P$) — ilość punktów zgromadzonych przez drużynę

6.3.1 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 Rekonstrukcja.

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	bad coordinates
102	bad rotation angle
103	bad turn specification
104	game over - no more puzzles to solve
105	turn hasn't finished yet
106	changes not available

6.4 Przykład

Poniżej znajduje się przykładowy zapis komunikacji z serwerem.

klient → serwer	serwer → klient
	`LOGIN`
`login1`	
	`PASS`
`secret`	
	`OK`
`INFO`	
	`OK`
	`littleKitty password123 10 20`
`GUESS 3 15 1 1`	
	`OK`
`CHANGES`	
	`OK`
	`littleKitty`
	`151`
	`1`
	`3 15 1 1 0 (1)`
`POINTS`	
	`OK`
	`27.500`
`TURN`	
	`OK`
	`652`
`WAIT`	
	`OK`
	`WAITING 0.67`
	`OK`

6.5 Serwery

Poniżej znajduje się zestawienie serwerów wraz z wartościami parametrów rozgrywki

- **TurnDuration** - długość tury w sekundach
- **ChangesFlashback** - liczba ostatnich zakończonych tur, o które można pytać przy pomocy komendy CHANGES
- **MaxCalls** - maksymalna liczba poleceń w turze

- **SchedulePadding** - interwały dzielące zaplanowane rozpoczęcia kolejnych rund (por. paragraf **Kalendarz rund**) w minutach
- **StartTime** - godzina uruchomienia serwera

nazwa	adres:port	**Turn Duration**	**Changes Flashback**	**MaxCalls**	**Schedule Padding**	**StartTime**
B1	universum.dl24:20003	1	100	30	10	10:00
B2	universum.dl24:20004	1	300	40	10	14:00
B3	universum.dl24:20005	1	800	50	10	18:00

7 Zadanie Złodzieje i złodzieje

7.1 Wprowadzenie

Znany (z zadania Tour) i lubiany (a jakże) Admirał Żukk w dalszym ciągu administruje planetą w odległym zakątku terytorium żukoskoczków, a jego zwierzchnicy w dalszym ciągu składają mu zapowiedziane, ale niezbyt wyczekiwane wizyty. Kolejne odwiedziny urzędników wypadają za niecały miesiąc, a tymczasem - jak na złość - na planecie Admirała zaczęły narastać problemy wewnętrzne.

Ot na przykład - w mieście Gachico szaleje przestępczość - zamaskowane gangi złodziejaszków okradają banki w biały dzień, a także w czarną noc; cała policja wzięła urlopy zdrowotne, mieszkańcy (i bankierzy!) apelują o pomoc. Słowem - sytuacja dojrzała do interwencji. Poza przygotowaniem sprzętu wojskowego i mobilizacją żołnierzy, Admirał chciałby zadbać o odpowiednie przygotowanie taktyczne swojej kadry oficerskiej. W myśl starej zasady *poznaj swojego wroga* sprowadził więc do bazy grę planszową "Złodzieje i złodzieje" (podobną koncepcyjnie do gry "Policjanci i złodzieje", tylko bez policji). Z rozkazu Admirała oficerowie muszą rozegrać między sobą partię gry planszowej, żeby wczuć się w sposób myślenia i działania bankowych złodziei.

Jako młodzi i ambitni oficerowie wojskowi nie możecie przepuścić tak doskonałej okazji do zaimponowania Admirałowi. Do łomów!

7.2 Opis problemu

7.2.1 Szajka

W grze "Złodzieje i złodzieje" będziecie sterować szajką rabusiów napadających na banki. Wasza szajka będzie składać się z wielu złodziejaszków podzielonych na poruszające się po mieście zespoły.

Każdego złodziejaszka opisuje para umiejętności:

- Zręczność - liczba naturalna z przedziału 0-5 oznaczająca umiejętność otwierania bankowych sejfów.
- Udźwig - liczba naturalna z przedziału 0-5 oznaczająca ilość pieniędzy, które może nieść dany złodziejaszek.

Każdy złodziejaszek dołącza do szajki jako rekrut. Aby mógł cokolwiek zrobić, należy go wcześniej wyszkolić. Wyszkolenie sprowadza się do podziału puli pięciu punktów w dowolny sposób pomiędzy zręczność i udźwig złodziejaszka. Ta decyzja jest nieodwracalna, od momentu jej podjęcia rekrut staje się złodziejaszkiem i trafia do Puli Wolnych Złodziejaszków, gdzie będzie oczekiwał na przypisanie go do zespołu.

Każdej tury do Waszej szajki mogą dołączać nowi rekruci. Dokładna liczba dołączających każdej tury rekrutów nie jest (to znaczy nie musi być) stała ani też określona przepisami, ale w każdej turze będzie identyczna dla wszystkich drużyn. Rekrutów nie trzeba szkolić w tej samej turze, w której dołączyli do szajki, dołączający rekruci kumulują się w oczekiwaniu na wyszkolenie.

7.2.2 Zespoły

Złodziejaszki poruszają się i pracują w zespołach o dowolnej niezerowej liczbie członków. W czasie gry zespoły mogą być tworzone i rozwiązywane, z zachowaniem następujących zasad:

- Zespół może zostać utworzony spośród dowolnego niepustego podzbioru Puli Wolnych Złodziejaszków Waszej szajki i w momencie utworzenia zostaje umieszczony w Dziupli (czyli w ustalonym segmencie miasta będącym bazą wypadową złodziejaszków, por. sekcja **Miasto**)
- Zespół może zostać rozwiązany tylko jeśli aktualnie znajduje się w Dziupli, a jego członkowie trafiają do Waszej Puli Wolnych Złodziejaszków.
- Liczba zespołów w Waszej szajce nie może przekroczyć maksymalnej wartości określonej w parametrach świata (por. sekcja **Serwery**)

Tworzenie i rozwiązywanie zespołu dokonuje się natychmiast w momencie wydania takiego polecenia.

7.2.3 Miasto

Miasto zbudowane jest na planie prostokątnej siatki kwadratowych segmentów. Siatka ma wymiary **N** wierszy (numerowanych od 1 do **N**) na **M** kolumn (numerowanych od 1 do **M**) i składa się z **M** x **N** jednostkowych segmentów. Dokładnie jeden segment w mieście jest, wspólną dla wszystkich szajek, Dziuplą. Pozostałe segmenty dzielą się na banki i zwykłe segmenty, nie zawierające żadnych interesujących budynków. Lokalizacja Dziupli jest znana, lokalizacje banków drużyny muszą ustalić samodzielnie.

Zespoły złodziejaszków mogą swobodnie poruszać się po mieście. Z każdego segmentu w mieście zespół może przejść do dowolnego spośród jego czterech sąsiadów - górnego, dolnego, lewego i prawego, pod warunkiem że dany sąsiad istnieje.

Przejście zespołu z segmentu do sąsiedniego segmentu zajmuje jedną turę. Konkretnie, jeśli polecenie przejścia do sąsiedniego segmentu zostanie wydane w **N**-tej turze, to w turze **N**+1 dany zespół będzie w segmencie docelowym.

7.2.4 Banki

Każdy bank zajmuje dokładnie jeden, ustalony segment miasta i jest opisywany przez trzy liczby:

- Poziom zabezpieczeń - liczba naturalna z przedziału 0 - 100, oznaczająca minimalną sumę zręczności członków zespołu potrzebną, aby zespół mógł się włamać do danego banku.
- Dzienny zysk - liczba naturalna z przedziału 0 - 5 oznaczająca następujący każdej tury przyrost ilości pieniędzy w bankowym sejfie.
- Maksymalna ilość pieniędzy - liczba naturalna z przedziału 0 - 100 oznaczająca wartość, po osiągnięciu której ilość pieniędzy w banku przestaje się zwiększać. Jeśli później ilość pieniędzy w sejfie ulegnie obniżeniu na skutek udanego włamania, ponownie zacznie się zwiększać do osiągnięcia maksymalnej ilości.

Liczba banków w mieście, ich pozycje i cechy **nie** są jawne, zdobycie tych informacji będzie istotnym elementem pracy szajki.

Rozbudowa W miarę upływu czasu, banki mogą decydować się na rozbudowę swoich placówek. W wyniku rozbudowy parametry banku (niektóre lub wszystkie) zostają zwiększone, zawsze jednak będzie to zmiana na wartości większe, które nadal będą spełniać podane wyżej ograniczenia (liczby naturalne w przedziale 0-100 lub 0-5). Każdy bank może rozbudowywać się wielokrotnie, a szajki nie są w żaden sposób informowane o rozbudowach.

7.2.5 Złodziejski proceder

Napady na banki Zespół znajdujący się w segmencie w którym jest bank, może dokonać napadu. Napad odbywa się w czasie rzeczywistym (tj. natychmiast, w szczególności w tej samej turze, w której zostało wydane takie polecenie). W wyniku napadu:

- Jeśli suma zręczności członków zespołu jest mniejsza od poziomu zabezpieczeń banku, napad jest nieudany i drużyna jest o tym informowana (ale nie poznaje dokładnej wartości poziomu zabezpieczeń banku).
- W przeciwnym wypadku napad się udaje, a zespół zdobywa ilość pieniędzy równą mniejszej z dwóch wartości: pozostałemu udźwigowi zespołu i ilości pieniędzy w skarbcu banku. Drużyna jest informowana o powodzeniu napadu i ilości zdobytych pieniędzy, a także ilości środków pozostałych w skarbcu (jeśli takowe w skarbcu pozostają, tj. jeśli dany zespół nie miał wystarczająco dużo wolnego udźwigu, aby zabrać wszystkie).

Nieudany napad na bank nie ma żadnych dodatkowych negatywnych konsekwencji, w szczególności zespół nie traci ew. przenoszonych pieniędzy uzyskanych w poprzednich udanych napadach.

Transport łupów Każdy zespół może przenosić ilość łupów z przedziału od 0 do sumy udźwigu jego członków. Pieniądze te mogą być gromadzone w czasie pojedynczego napadu, lub kilku kolejnych, tak długo, jak długo zrabowane pieniądze nie osiągną sumarycznego udźwigu zespołu.

Pieniądze z napadów trafiają do puli pieniędzy zdobytych przez szajkę, kiedy tylko dany zespół wejdzie do Dziupli. Wszystkie środki zrabowane przez dany zespół są wtedy automatycznie zapisywane na koncie drużyny a sam zespół odzyskuje pełny udźwig równy sumie udźwigów jego członków.

7.2.6 Punktacja

Wynik Waszej drużyny będzie równy łącznej wartości zdeponowanych w Dziupli pieniędzy.

7.3 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 "Złodzieje i złodzieje".

DESCRIBE_CITY Podaje wymiary miasta i lokalizację Dziupli.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- N ($N \in \mathbb{N}$) — liczba wierszy planu miasta
- M ($M \in \mathbb{N}$) — liczba kolumn planu miasta

Druga linia:

- R ($N \in \mathbb{N}$, $1 \leqslant R \leqslant N$) — numer wiersza segmentu zawierającego Dziuplę
- C ($M \in \mathbb{N}$, $1 \leqslant C \leqslant M$) — numer kolumny segmentu zawierającego Dziuplę

COUNT_RECRUITS Podaje liczbę dostępnych rekrutów.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- X ($X \in \mathbb{N}$) — liczba dostępnych rekrutów

TRAIN_RECRUIT Trenuje jednego rekruta, czyniąc go złodziejaszkiem. Drużyna musi posiadać niezerową liczbę rekrutów.

Parametry:

- Z ($Z \in \mathbb{N}$, $0 <= Z <= 5$) — zręczność przyszłego złodziejaszka
- U ($U \in \mathbb{U}$, $0 <= U <= 5$, $Z + U = 5$) — udźwig przyszłego złodziejaszka

Dane (od serwera):
Pierwsza linia:

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

LIST_THIEVES Podaje listę złodziejaszków w szajce.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- X ($X \in \mathbb{N}$) — liczba złodziejaszków

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

- ID ($ID \in \mathbb{N}$) — unikalny numer porządkowy złodziejaszka
- Z_Z ($Z_Z \in \mathbb{N}$) — zręczność złodziejaszka
- Z_U ($Z_U \in \mathbb{N}$) — udźwig złodziejaszka
- Z_T ($Z_T \in \mathbb{C}$) — numer porządkowy zespołu do którego należy Złodziejaszek, lub -1, jeśli jest w Puli Wolnych Złodziejaszków

MAKE_TEAM Tworzy zespół z podanych złodziejaszków. Każdy z podanych złodziejaszków musi znajdować się w Puli Wolnych Złodziejaszków, tj. nie być przypisanym do żadnego zespołu.

Parametry:

- X ($X \in \mathbb{N}$) — liczba złodziejaszków, którzy mają wejść w skład tworzonego zespołu
- Ciąg X (oddzielonych spacjami) liczb ID_i ($ID_i \in \mathbb{N}$) — numery porządkowe złodziejaszków, którzy mają wejść w skład tworzonego zespołu

Dane (od serwera):
Pierwsza linia:

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

LIST_TEAMS Podaje listę zespołów złodziejaszków.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- X ($X \in \mathbb{N}$) — liczba zespołów

Każda kolejna linia opisuje jeden zespół za pomocą oddzielonych spacjami liczb:

- ID ($ID \in \mathbb{N}$) — unikalny numer porządkowy zespołu
- R ($R \in \mathbb{N}$) — numer wiersza segmentu, w którym znajduje się zespół
- C ($C \in \mathbb{N}$) — numer kolumny segmentu, w którym znajduje się zespół
- L ($L \in \mathbb{N}$) — ilość pieniędzy przenoszona przez zespół

DESCRIBE_TEAM Opisuje wybrany zespoł złodziejaszków.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer porządkowy zespołu, który ma zostać opisany

Dane (od serwera):
Pierwsza linia:

- ID ($ID \in \mathbb{N}$) — unikalny numer porządkowy opisywanego zespołu
- R ($R \in \mathbb{N}$) — numer wiersza segmentu, w którym znajduje się opisywany zespół
- C ($C \in \mathbb{N}$) — numer kolumny segmentu, w którym znajduje się opisywany zespół
- L ($L \in \mathbb{N}$) — ilość pieniędzy przenoszona przez opisywany zespół

DISMANTLE_TEAM Rozwiązuje podany zespół. Podany zespół musi przebywać w Dziupli.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer porządkowy zespołu, który ma zostać rozwiązany

MOVE_TEAM Wydaje podanemu zespołowi polecenie przejścia do podanego segmentu. Podany segment musi być bezpośrednim sąsiadem segmentu, w którym przebywa podany zespół. Przejście do nowego segmentu odbywa się zaraz po zakończeniu obecnej tury. Jeśli w jednej turze zostanie wydanych wiele poprawnych poleceń MOVE_TEAM w stosunku do tego samego zespołu, wykonane zostanie ostatnie z nich.

Parametry:

- ID ($ID \in \mathbb{N}$) — numer porządkowy zespołu
- R ($R \in \mathbb{N}$) — numer wiersza segmentu, do którego zespół ma przejść
- C ($C \in \mathbb{N}$) — numer kolumny segmentu, do którego zespół ma przejść

DESCRIBE_LOCATION Zwraca informację o segmencie, w jakim znajduje się podany zespół.

Parametry:

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

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

- NEST — jeśli podany zespół znajduje się w Dziupli
- BANK — jeśli podany zespół znajduje się w banku
- REGULAR — jeśli podany zespół znajduję się w zwykłym segmencie

ROB_BANK Wydaje podanemu zespołowi polecenie dokonania napadu na bank. Podany zespół musi znajdować się w banku.

Parametry:

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

Dane (od serwera):
Jeśli napad na bank będzie udany, serwer zwróci jedną linię zawierającą (oddzielone spacjami) napis

- SUCCESS

i dwie liczby naturalne:

- R ($R \in \mathbb{N}$) — ilość zrabowanych pieniędzy
- L ($L \in \mathbb{N}$) — ilość pieniędzy, które pozostały w skarbcu

W przeciwnym wypadku serwer zwróci jedną linię zawierającą napis

- FAILURE

GET_SCORE Podaje ilość pieniędzy dostarczonych do Dziupli przez Waszą drużynę.

Parametry:brak

Dane (od serwera):

Pierwsza linia:

- N ($N \in \mathbb{N}$) — ilość zdeponowanych pieniędzy

WAIT Czeka do momentu rozpoczęcia kolejnej tury.

Parametry:brak

Dane (od serwera):

Serwer odpowiada jedną linią zawierającą (oddzielone spacjami) napis

- WAITING

oraz liczbę

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

7.3.1 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 Złodzieje i złodzieje.

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	no more recruits available
102	invalid skill values
103	no more thieves available
104	duplicate identifier detected
105	invalid identifier
106	unavailable identifier
107	invalid location
108	distance too great
109	too many teams

7.4 Przykład

Poniżej znajduje się przykładowy zapis komunikacji z serwerem.

klient → serwer	serwer → klient
	`LOGIN`
`login1`	
	`PASS`
`secret`	
	`OK`
`DESCRIBE_CITY`	
	`OK`
	`10 20`
	`1 1`
`COUNT_RECRUITS`	
	`OK`
	`2`
`TRAIN_RECRUIT 0 5`	
	`OK`
	`1`
`TRAIN_RECRUIT 5 0`	
	`OK`
	`2`
`LIST_THIEVES`	
	`OK`
	`2`
	`1 0 5 -1`
	`2 5 0 -1`
`MAKE_TEAM 2 1 2`	
	`OK`
	`3`
`MOVE_TEAM 1 2 1`	
	`OK`
`WAIT`	
	`WAITING 16.2`
	`OK`

7.5 Serwery

Poniżej znajduje się zestawienie serwerów wraz z wartościami parametrów rozgrywki

- **N** - liczba wierszy miasta
- **M** - liczba kolumn miasta
- **TurnDuration** - długość tury w sekundach
- **MaxTeams** - maksymalna liczba zespołów w szajce
- **MaxCalls** - maksymalna liczba poleceń w turze

nazwa	adres:port	**N**	**M**	**TurnDuration**	**MaxTeams**	**MaxCalls**
C1	universum.dl24:20006	1000	1000	10	50	200
C2	universum.dl24:20007	2000	3000	15	60	220
C3	universum.dl24:20008	500	2000	10	50	200