

Правила Cloud Contest 2.

Основной тур

Игра заключается в соревновании ботов. Нужно бегать по полю несколькими ботами сразу и стрелять в ботов противника. Они от этого умирают и вам начисляются очки. Победителем является набравший наибольшее количество очков за определенное число ходов участник.

В одной игре может участвовать 2 и более игроков. У каждого игрока есть 6 ботов. Ход делается одновременно всеми ботами всех участников. Игра проходит на прямоугольном поле $(0..N-1) \times (0..M-1)$ клеток. Каждый бот каждого игрока за 1 ход может ходить вверх-вниз-вправо-влево или стоять на месте. Он не может выходить за границы поля.

На поле могут быть препятствия — стены, клетки по которым нельзя пройти. Два бота не могут встать на одну клетку.

Игрок получает информацию, как стены и его боты расположены относительно границ поля, однако не знает расположение всех ботов противника. Каждый ход ему сообщается только о тех объектах, которые находятся от какого-то из его ботов на расстоянии, не большем заданного числа. Расстояние — это сумма разницы координат по X и разницы координат по Y. Вы можете считать, что каждый ваш бот создаёт вокруг себя область видимости в форме ромба и вы видите врагов только в пределах суммарной области видимости.

Кроме передвижения, каждый бот может раз в ход выстрелить по одной из клеток в пределах дальности стрельбы — то есть цель должна быть от стреляющего не дальше чем на эту величину. Все команды на передвижения и выстрелы выдаются одновременно, однако обрабатываются сначала выстрелы. Если за ход вы и стреляете, и перемещаетесь, ваш выстрел корректен, если цель находилась в пределах дальности стрельбы **до** перемещения. Изначально у каждого бота 3 жизни. Если в бота выстрелили, он теряет одну жизнь. Если это была его последняя, он умирает и через 10 ходов появится с полными жизнями в клетке поля, выбранной случайно равномерно среди всех пустых клеток, наименее простреливаемых его противниками. Тот игрок, чей бот стрелял по нему в этот ход, получает 100 очков. Если таких игроков несколько, очки делятся между всеми ними пропорционально количеству сделанных выстрелов и округляются вниз. Вы не получаете очков за стрельбу по своим ботам, однако можете наносить им урон.

Если бот умирает и он стрелял на этом ходу, его выстрел будет обработан, он мог кого-то убить и принести хозяину очки.

После выстрелов обрабатываются перемещения. Если два или более бота сходили в одну и ту же клетку, то в таком случае они остаются на своих местах.

Протокол обмена данными

Клиент взаимодействует с сервером через сокет соединения. Протокол взаимодействия простой linetext-based протокол, закодированы в json.

Окончание строки "\n", прошу заметить НЕ "\r\n".

Логин

После установки TCP-соединения сервер должен выслать сообщение

```
{"status":"ok", "message":"ready to login"}
```

В ответ клиент должен отправить команду логина

```
{"message_type":"login", "user_token":"123abcd",  
"game_token":"1234abcd"}
```

где user_token — токен для аутентификации пользователя,
game_token — идентификатор игры, в которую логинится бот.

Сервер должен ответить:

```
{"status":"ok", "message":"logged in"} или {"status":"error",  
"message":"invalid user"}
```

После того как все боты подключатся к игре, игра начинается и сервер всем отправляет конфигурацию поля

```
{"user_id":"nick1", "width" : 10, "height" : 10, "bricks":[[1,1],  
[2,2]]}
```

user_id — идентификатор пользователя в этой игре,

width, height — высота и ширина поля (в клетках)

bricks — массив координат, в которых расположены препятствия, каждый элемент этого массива имеет вид [x, y].

Затем сервер отправляет клиентам конфигурацию на начало хода

```
{"turn_no":1, "players":player_hash, "bots":bots_hash,  
"empty":[[2,1],[2,3],[3,2],[3,3]]}
```

здесь

turn_no — номер хода,

player_hash содержит имена игроков и количество набранных ими очков в этой игре и имеет вид

```
{"nick1":133,"nick2":33,"nick3":33}
```

`bots_hash` имеет вид

```
{ "bot1": { "player": "nick1", "position": [1,2], "health": 2, "time": 0,
"can_shoot": [[1,3], [1,1], [0,2], [2,2]] },
"bot2": { "player": "nick2", "position": [10,7], "health": 0, "time": 7,
"can_shoot": [[10,6], [10,8], [9,7], [11,7]] } }
```

ключ хеша это idшники ботов,

`player` — какой игрок владеет этим ботом,

`position` — положение бота на игровом поле,

`health` — количество жизней у этого бота,

`time` — если количество жизней больше нуля, это ноль; иначе, это количество ходов до восстановления бота в случайной точке игрового поля;

`can_shoot` — массив клеток, по которым этот бот может стрелять.

В `bots_hash` перечисляются только те боты, которые находятся в пределах видимости игрока.

`empty` — массив пустых (не занятых ботом или препятствием) клеток, которые находятся в поле зрения игрока.

В ответ бот-клиент должен отправить хеш команд, указав для каждого своего живого бота, в каком направлении сделать шаг, стрелять ли и куда:

```
{ "bot1": { "move": "u", "fire": [3,4] }, "bot2": { "move": "u", "fire": [] } }
```

`move` может принимать значения: u — up (0, -1), d — down (0, 1), l — left (-1, 0), r — right (1, 0), s — stay (0,0). В скобках указаны изменения координат.

`fire` — это координаты клетки, по которой бот хочет стрелять в виде [x,y]. Если бот не намерен стрелять на данном ходу, `fire` принимает значение [].

Игрок должен сделать ход за 0,25 секунды. Если клиент не отправляет команду, то сервер выдает сообщение `{ "status": "error", "message": "turn timeout" }`. При этом считается, что игрок выдал команду по умолчанию `"stay"` всем своим ботам и никем никуда не стрелял.

После того как получены команды от всех ботов, на сервере вычисляется новое состояние мира — новые положения и жизни ботов и очки игроков.

Если соединение с клиентом будет прервано, после установки повторного соединения с клиентом игрок может продолжить игру с текущего хода. Ему снова будет выслана конфигурация поля и конфигурация на начало хода, если эта игра уже началась.