CLUE AI FRAMEWORK

RULES

Clue is a turn-based game of solving who committed a murder. The competition rules are based on the original board game, but they are simplified slightly to make programming a bot less of a headache. Essentially, there will be a board with rooms scattered about it. Players will be given a list of weapons and players as well. The goal of the game is to determine the room in which a murder was committed, the weapon used, and the murderer.

At the beginning of the game, all players will be told what the board is. It will always be a square, but the rooms placed on the board will have randomized positions and names. A randomized room, weapon, and player will be chosen to be used in the murder. Players will be dealt cards, each one containing either a room, weapon, or player that was not a part of the murder. There will be at least 2 players per game. All players will start in the room with 0 as the id.

On each player's turn, players will be told how far they can move, which will be a random number between 1 and 12, along with how long they have to determine their move. Players may move to any space within *n* spaces, where *n* is the random number generated that represents their dice roll. Players may also opt to not move. After moving, if the player is in a room, the player must guess who committed the murder, the weapon used, and the room in which the murder was committed. The player and weapon guessed may be any player and weapon in the game, but the room guessed must be the room in which the player is currently located in. If the player ever submits an invalid move, the player will automatically lose and be removed from the game.

In order to win a game, a player must submit an accusation in place of their move. An accusation is the same as a guess, except players do not have to be located in the room in which they think the murder occurred. If a player gets their accusation wrong, that player will lose and be removed from the game. If a player gets their accusation correct, then that player wins the game.

COMMUNICATING WITH THE ENGINE

Communication with the engine will be done through standard input and output channels, so for example, with Java, the engine will send information to your bot through System.in and your bot will send information to the engine through System.out. Because standard in and out are used, this competition is language agnostic so long as the language to run your bot is installed on the device that will run the competition. The competition will be run on a Linux-based system, likely a Raspberry Pi running Arch Linux. Below is a table representing all possible communications from the engine to your bot.

Text	**Meaning**	**Notes**
ID [id]	The number that the engine will use to represent your bot.	[id] will be an integer greater than or equal to 0.
Playercount [count]	The number of players that are currently in the game.	[count] will be an integer greater than or equal to 2.
Board [width] [height]	The size of the board. This will be followed by a 2-D table of numbers representing the locations of rooms on the board with zeros representing empty spaces and positive numbers representing 1+the id of the room in that location.	[width] and [height] will be equal integers greater than or equal to 5.
Name	The engine is asking your bot for its name.	Print out your bot's name following this request.
Roomname [id] [name]	The id and name of a new room. There will always be $\sqrt{board\ area}$ rooms.	[id] will be an integer greater than or equal to 0. No id values will be repeated. [name] will be a string that may contain spaces.
Weaponname [count] [id] [name]	The total number of weapons along with the id and name of a new weapon.	[count] will be an integer greater than or equal to 15. [id] will be an integer greater than or equal to 0. No id values will be repeated. [name] will be a string that may contain spaces.
Personname [count] [id] [name]	The total number of people along with the id and name of a new person.	[count] will be an integer greater than or equal to 15. [id] will be an integer greater than or equal to 0. No id values will be repeated. [name] will be a string that may contain spaces.
Card [type] [id]	A card that is being dealt to you.	[type] will be either a 0, 1 or 2. 0 represents a person, 1 a weapon, and 2 a room. [id] will be an integer greater than or equal to 0.
Move [time] [range]	The engine is asking your bot what it would like to do on its turn. Should be followed by an appropriate move command.	[time] will be an integer greater than 0 representing the number of nanoseconds your bot will have to generate its next move. [range] will be the number of spaces your bot may move on this turn.
Opponent [id] [y] [x]	The engine is telling your bot about an opponent's new location.	[id] will be the id of the opponent that just moved. [y] will be your opponent's new y-coordinate on the board. [x] will be your opponent's new x-coordinate on the board.

Disprove [player] [person] [weapon] [room]	The engine is asking your bot if it can refute the given accusation or guess. Should be followed by a disprove command.	[player] will be the id of the player that made this accusation. [person] will be the id of the person being accused. [weapon] will be the id of the weapon being accused. [room] will be the id of the room in which the murder is believed to have occurred in.
Card From [player] [type] [id]	A bot has disproved your guess. Note that your bot may not use cards from this command in its responses to disprove commands.	[player] will be the id of the player that disproved you. [type] will be either a 0, 1 or 2. 0 represents a person, 1 a weapon, and 2 a room. [id] will be an integer greater than or equal to 0.
Disproved [player]	A bot has disproved the previous guess.	[player] will be the id of the player that disproved the guess.
Accusation [player] [person] [weapon] [room]	A bot has made a false accusation.	[player] will be the id of the player that made this accusation. [person] will be the id of the person being accused. [weapon] will be the id of the weapon being accused. [room] will be the id of the room in which the murder is believed to have occurred in.
Win [id]	A bot has won the game.	[id] will be the id of the winning bot.

Table 1. A table of all possible interactions from the engine to your bot.

Follows	**Text**	**Meaning**	**Notes**
Name	[name]	Your bot's name.	[name] will be the name of your bot, which may contain spaces.
Move [time] [range]	*See Table 3.*	Your bot's move.	*See Table 3.*
Disprove [person] [weapon] [room]	[-1, 0, 1, or 2]	Your bot is not in agreement with the player's accusation or guess.	Print -1 if your bot has none of the given cards. Otherwise, print the type of the card that is incorrect. If your bot submits -1 even though your bot knows that something about the guess is incorrect, your bot will be removed from the game.

Table 2. A table of all possible interactions from your bot to the engine.

Text	**Meaning**	**Notes**
Move [y] [x]	Your bot would like to move to (x, y) on the board.	If your move is invalid, you will be removed from the game and lose.
Move [y] [x] [player] [weapon] [room]	Your bot would like to move to (x, y) on the board. It would then like to guess that [player] committed the murder using [weapon] in [room].	If your move is invalid, you will be removed from the game and lose.
Guess [player] [weapon] [room]	Your bot would like to guess that [player] committed the murder using [weapon] in [room].	
Accusation [player] [weapon] [room]	Your bot would like to accuse that [player] committed the murder using [weapon] in [room].	If your accusation is incorrect, your bot will lose. If your accusation is correct, your bot will win. This is the only way to win assuming that all bots make legal moves.

Table 3. A table of all possible move commands from your bot to the engine.

SAMPLE GAME

Below is a sample game between a bot and the engine. Normal text will represent messages from the engine to the bot, and bolded text will represent messages from the bot to the engine.

```
ID 0
Playercount 1
Board 10 10
0 0 0 0 0 0 0 0 0 0
10 0 8 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
```

```
0 0 0 0 0 0 0 0 0 6
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 3 0 0 0
0 5 0 0 0 0 7 0 0 0
0 0 0 0 0 0 2 0 0 0
1 0 0 9 0 0 0 0 0 0
4 0 0 0 0 0 0 0 0 0
Name
```
HumanPlayer
```
Roomname 0 Krakowiak's Lab
Roomname 1 3rd Floor Landing
Roomname 2 WorldStud Room
Roomname 3 Boardroom
Roomname 4 Grisham's Room
Roomname 5 L Waddell's Room
Roomname 6 Comp Lab B
Roomname 7 Nikki's Room
Roomname 8 Spanish Room
Roomname 9 Seward's Lab
Weaponname 6 0 computer keyboard
Weaponname 6 1 paper
Weaponname 6 2 laptop
Weaponname 6 3 car
Weaponname 6 4 fire extinguisher
Weaponname 6 5 door knob
Personname 11 0 Director Alderdice
Personname 11 1 Dean Gregory
Personname 11 2 Kim McKean
Personname 11 3 Dr. Rhuele
Personname 11 4 Dr. Kostopulos
Personname 11 5 Dr. Leigh
Personname 11 6 Brian Isbell
Personname 11 7 Dr. Oatsvall
Personname 11 8 Bryan Adams
Personname 11 9 Señor Mac
Personname 11 10 James Katowich
Card 0 4
Card 0 2
Card 0 0
Card 0 9
Card 0 3
Card 0 6
Card 0 8
Card 0 1
Card 0 7
Card 0 5
Card 1 9
Card 1 5
Card 1 11
Card 1 6
Card 1 4
Card 1 2
Card 1 0
Card 1 1
Card 2 5
Card 2 3
Card 2 8
Card 2 9
Card 2 7
Card 2 2
Card 2 0
Card 2 6
Card 2 1
Move 100000000000 12
Move 8 3 8 4 9
Disprove 8 4 9
Disprove 0
Move 100000000000 2
Move 9 2
Move 100000000000 12
Move 9 0 10 3 4
Disprove 10 3 4
Disprove -1
Move 100000000000 7
Accusation 10 3 4
Win 0
```

TESTING YOUR BOT

Once you have written your bot, you may execute your bot against any other bots. The engine will be written in Java, so you must have Java installed in order to test your bot. In order to test your bot, simply execute the engine code using the command to execute your bot along with all other bots it will compete against as a command-line argument. You will also be provided with an interface to play games against the bots as a human along with a basic bot. Here is the sample code to play the game with 2 human players.

```
java -jar Clue.jar “java -jar Human.jar Name1” “java -jar Human.jar Name2”
```