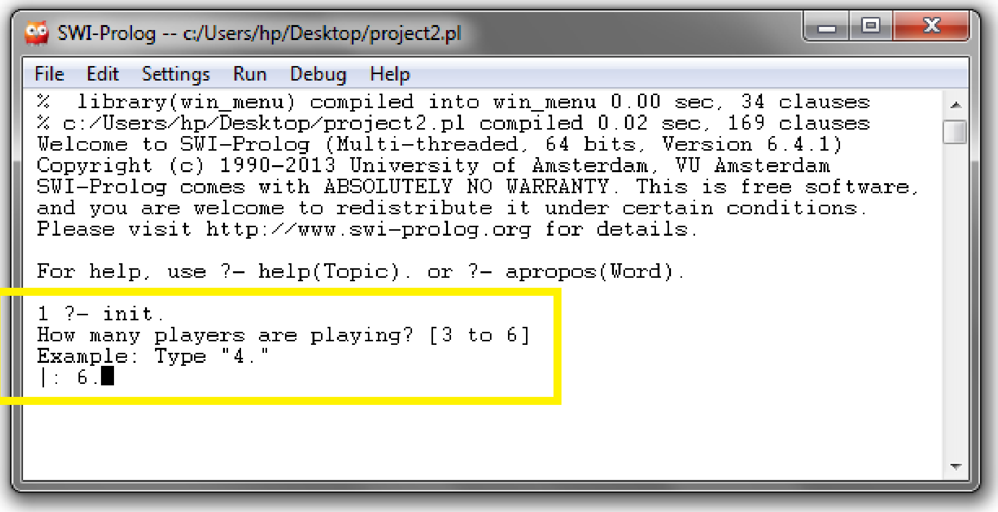
Clue Assistant

This is a project developed in Prolog, and it has the purpose of assisting you while playing the board game **Clue** with others. To use the application, you must first have Prolog installed in your machine. Then, you can execute the project **.pl** file in the Prolog interpreter. Once you do that, a console window will open where you will be able to interact with the application.

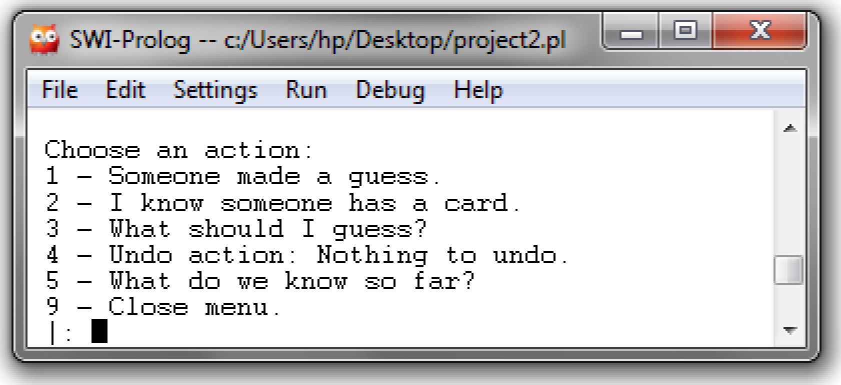
To use it, first you must type **init.** Note that everything you type must be followed by a dot. When calling init, the application will ask you to enter the number of players that are playing the board game with you. Please enter a valid number as suggested by the application.


SWI-Prolog -- c:/Users/hp/Desktop/project2.pl
File Edit Settings Run Debug Help

```
%  library(win_menu) compiled into win_menu 0.00 sec, 34 clauses
% c:/Users/hp/Desktop/project2.pl compiled 0.02 sec, 169 clauses
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 6.4.1)
Copyright (c) 1990-2013 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?- init.
How many players are playing? [3 to 6]
Example: Type "4."
|: 6.
```

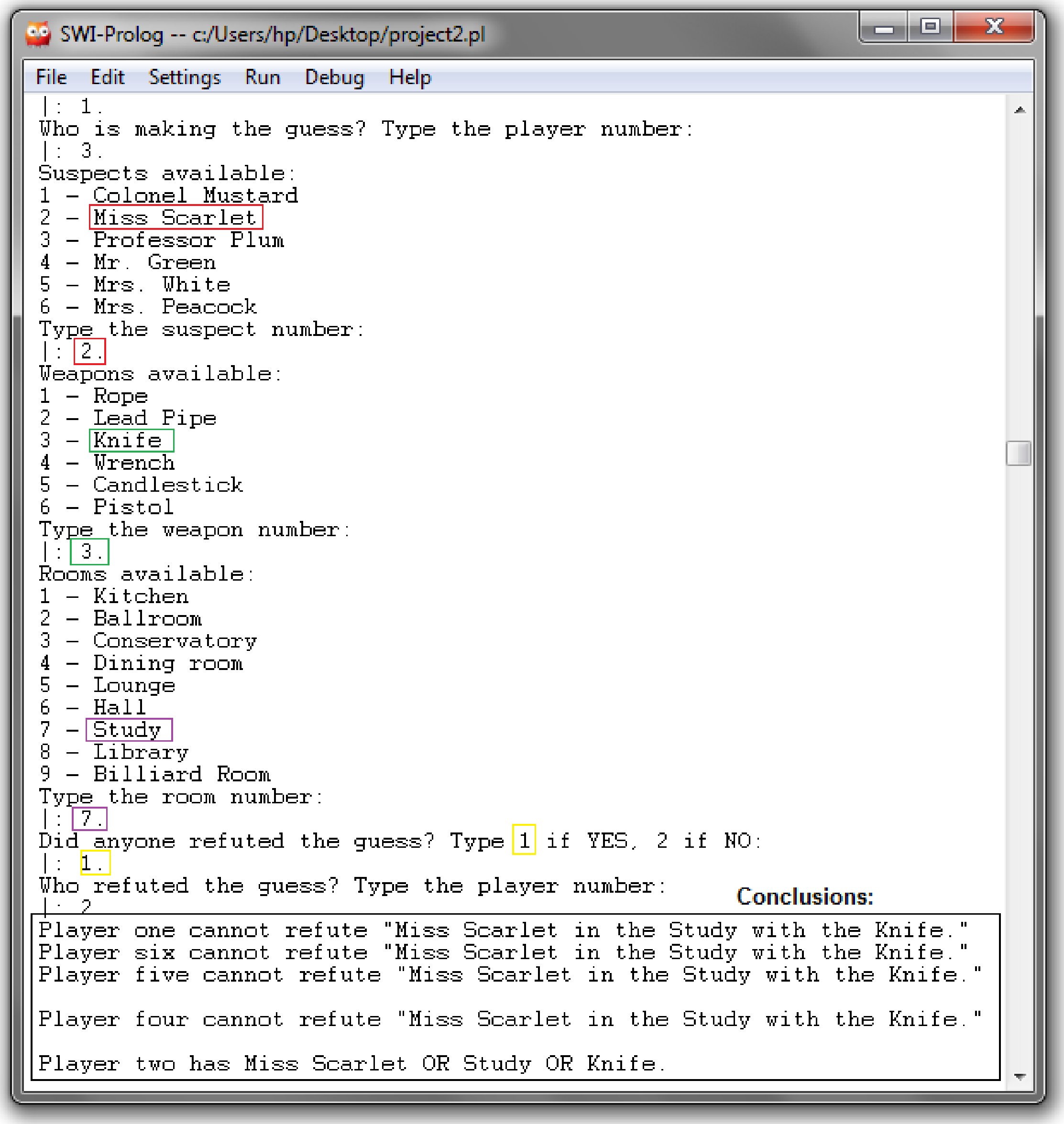
The application can only be initialized once. If you try to initialize again, it will not let you. If you want to restart the game, you can simply close the Prolog interpreter and open it again (it is recommended that you close the menu first).

Once it is initialized, there will be a menu where you can choose from several options. You type one of the numbers displayed in the screen with a period. Then, you press enter. All the input of this application happens by typing numbers.


SWI-Prolog -- c:/Users/hp/Desktop/project2.pl
File Edit Settings Run Debug Help

```
Choose an action:
1 - Someone made a guess.
2 - I know someone has a card.
3 - What should I guess?
4 - Undo action: Nothing to undo.
5 - What do we know so far?
9 - Close menu.
|: 
```

Option 1: when anyone in the game makes a guess (including you), choose this option. It will ask you what the guess was, who made it and who refuted it (if someone did). The player who refuted should have at least one of the 3 cards from the guess. The players between the player who guessed and the player who refuted cannot have any of those 3 cards suggested. To summarize, the application will take conclusions about what cards other players can't have or might have based on the information you provided. If any conclusion taken was already taken before (such as if I already know that a player can't have those cards), then the conclusion will be ignored and not stored again. All the conclusions that were not ignored will be shown in the screen after you entered all the input requested.



```
|: 1.
Who is making the guess? Type the player number:
|: 3.
Suspects available:
1 - Colonel Mustard
2 - Miss Scarlet
3 - Professor Plum
4 - Mr. Green
5 - Mrs. White
6 - Mrs. Peacock
Type the suspect number:
|: 2.
Weapons available:
1 - Rope
2 - Lead Pipe
3 - Knife
4 - Wrench
5 - Candlestick
6 - Pistol
Type the weapon number:
|: 3.
Rooms available:
1 - Kitchen
2 - Ballroom
3 - Conservatory
4 - Dining room
5 - Lounge
6 - Hall
7 - Study
8 - Library
9 - Billiard Room
Type the room number:
|: 7.
Did anyone refuted the guess? Type 1 if YES, 2 if NO:
|: 1.
Who refuted the guess? Type the player number:
|: 2
```

Conclusions:

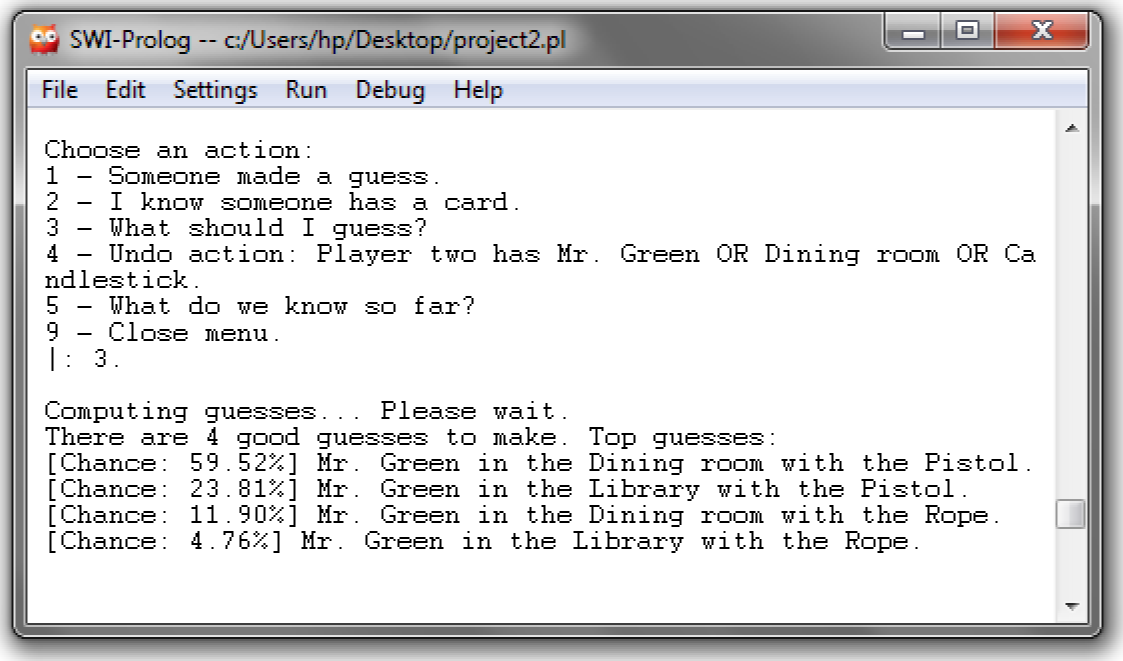
```
Player one cannot refute "Miss Scarlet in the Study with the Knife."
Player six cannot refute "Miss Scarlet in the Study with the Knife."
Player five cannot refute "Miss Scarlet in the Study with the Knife."

Player four cannot refute "Miss Scarlet in the Study with the Knife."

Player two has Miss Scarlet OR Study OR Knife.
```

The example in the illustration above is a 6-players game. Player 3 made the guess and Player 2 refuted it. Every game goes in a circle where you are player 1. The first player to check your guess is player 2, then player 3 and so on. Thus, if player 3 made the guess and player 2 refuted, that means player 4, player 5, player 6, and player 1 could not refute the guess. That also means player 2 has at least one of the cards since he/she refuted the guess. This is exactly what the application captures and stores it in its database (if not there yet).

This is how most of the game information is added to the application. You can call this option as much as you like. Once the guess is made, the application returns to the main menu. If for some reason an error happened anywhere in the application, you can always call the main menu again by typing **menu.**

Option 2: when you know someone has a card (including you), choose this option. This option will ask you who has the card and which card is it. Remember that you are player 1. So, you might want to call that once cards are dealt to let the application know about your own cards. This will add the information to the database only if the information is not there already. This is a useful option in case a player revealed a card to you or if you are a psychic.

Option 3: when it's your turn to make a guess, call this option before choosing a room to move. This option will analyse everything the application knows about the other players to suggest you some decent guesses that you can make. Based on the guesses provided and how far you can move, you can choose to which room you will go. Suggestions with the same probability will vary more in the rooms, so that you have more options in case you can't go to a specific room. Each suggestion shows its probability of matching the cards in the envelope. If there's only one suggestion (with 100% probability), you should make an accusation right away because that is the answer! If there are more than one suggestion, they will be sorted based on their probability. Only the top 40 suggestions will be shown in the screen.


SWI-Prolog -- c:/Users/hp/Desktop/project2.pl
File Edit Settings Run Debug Help

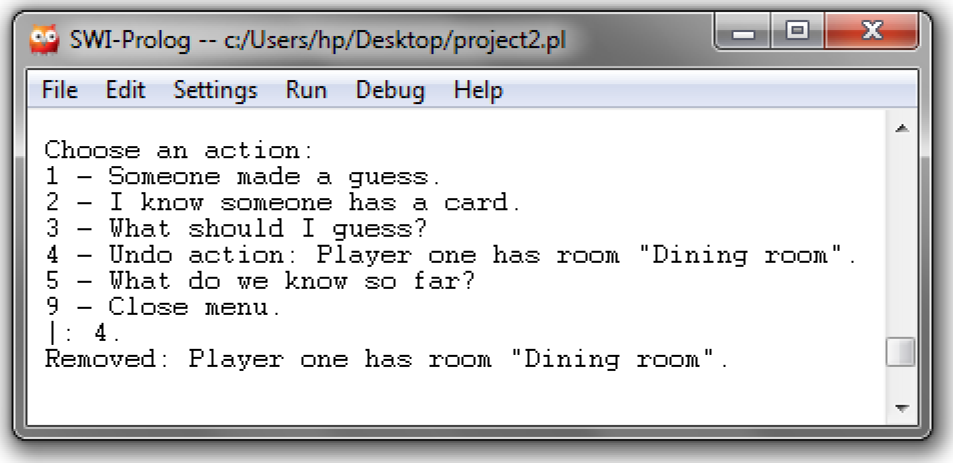
```
Choose an action:
1 - Someone made a guess.
2 - I know someone has a card.
3 - What should I guess?
4 - Undo action: Player two has Mr. Green OR Dining room OR Ca
ndlestick.
5 - What do we know so far?
9 - Close menu.
|: 3.

Computing guesses... Please wait.
There are 4 good guesses to make. Top guesses:
[Chance: 59.52%] Mr. Green in the Dining room with the Pistol.
[Chance: 23.81%] Mr. Green in the Library with the Pistol.
[Chance: 11.90%] Mr. Green in the Dining room with the Rope.
[Chance: 4.76%] Mr. Green in the Library with the Rope.
```

The probability of a card to be in the envelope is based on the normalization of the probabilities of each player having that card. Then, it's taken into consideration that the envelope has only 1 weapon, 1 room, and 1 suspect. A second normalization happens to give the final probability. When computing the probability of each suggestion, the application first analyses the raw data provided. Then, it tries to take logical conclusions from that data.

Those are some of the conclusions taken:

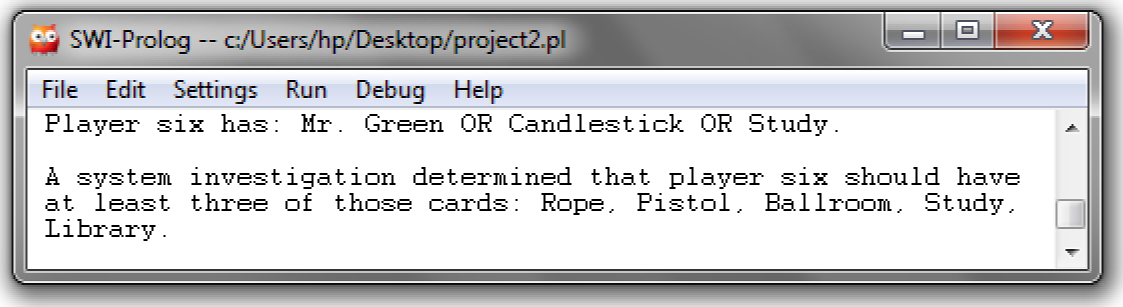
1. A player cannot have a card if someone else has it.
2. There cannot be a card in the envelope if a player has it.
3. If all players in game cannot have a card, that card must be in the envelope.
4. A player that refutes a guess has at least one of the 3 cards from that guess. If other players have 2 of those 3 cards, then we conclude this player must have that third card.
5. At most 5 suspects, 8 rooms and 5 weapons can be present among the players. That means if the players have already 5 suspects, they cannot have the last one, which must be in the envelope. The same is valid for rooms or weapons.
6. The application computes the minimum number of cards a player can have in hands, which is the floor of 18 divided by the number of players in game. So, if for example a player has 3 or 4 cards but can only have 3 possible cards, that means those 3 cards must belong to this player, which means everyone else cannot have them.

Option 4: mistakes happen in every game. If someone made a mistake and you ended up entering wrong data, no worries because you can always undo it. Choose this option if some information you entered was wrong. For example, suppose you typed that player 3 refuted the guess, but it was actually player 2! You can undo your actions. The way this works is that the application keeps track of a stack of all the actions made. Thus, you can undo every single action you made since the beginning of the game!


SWI-Prolog -- c:/Users/hp/Desktop/project2.pl
File Edit Settings Run Debug Help

```
Choose an action:
1 - Someone made a guess.
2 - I know someone has a card.
3 - What should I guess?
4 - Undo action: Player one has room "Dining room".
5 - What do we know so far?
9 - Close menu.
|: 4.
Removed: Player one has room "Dining room".
```

Option 5: choose option 5 to see everything the game has recorded so far. It shows the number of players in game, which cards we know each player has, and all the raw information captured from the information you provided on previous actions. It also shows a logical conclusion of what cards each player might have. This conclusion shares a portion of the code used on option 3 (described above).


SWI-Prolog -- c:/Users/hp/Desktop/project2.pl
File Edit Settings Run Debug Help

```
Player six has: Mr. Green OR Candlestick OR Study.

A system investigation determined that player six should have
at least three of those cards: Rope, Pistol, Ballroom, Study,
Library.
```