2018-10-24

Lisp Casino Manual

Joseph Allen | jallen6@ramapo.edu

Joseph Allen

Contents

[Bug Report: 2](#_Toc528182585)

[Feature Report: 2](#_Toc528182586)

[Non-Implemented: 2](#_Toc528182587)

[Implemented: 2](#_Toc528182588)

[How to run 2](#_Toc528182589)

[Data-Structures 3](#_Toc528182590)

[Round parameters: 3](#_Toc528182591)

[Log: 3](#_Toc528182592)

[October 3rd- 4th: 3](#_Toc528182593)

[October 5th: 3](#_Toc528182594)

[October 6th: 3](#_Toc528182595)

[October 7th 3](#_Toc528182596)

[October 8th 3](#_Toc528182597)

[October 9th: 4](#_Toc528182598)

[October 11th: 4](#_Toc528182599)

[October 12th: 4](#_Toc528182600)

[October 13th: 4](#_Toc528182601)

[October 14th: 4](#_Toc528182602)

[October 15th: 4](#_Toc528182603)

[October 16th: 4](#_Toc528182604)

[October 17th: 5](#_Toc528182605)

[October 18th 5](#_Toc528182606)

[October 19th 5](#_Toc528182607)

[October 20th: 5](#_Toc528182608)

[October 21st: 5](#_Toc528182609)

[October 22nd: 5](#_Toc528182610)

[Screenshots: 6](#_Toc528182611)

[Load-in: 6](#_Toc528182612)

[Ai Capture: 7](#_Toc528182613)

[Set Input and capture: 7](#_Toc528182614)

[Build being created: 8](#_Toc528182615)

[Capturing a build: 9](#_Toc528182616)

[Scoring (the first list is the human’s pile): 9](#_Toc528182617)

[Yes/No Validation: 10](#_Toc528182618)

[Numeric Validation: 10](#_Toc528182619)

# Bug Report:

* Builds get added to the pile as a build instead of being flattened
* doCycle doesn’t end until the hands are nil and they attempt to play
* In some cases, the computer can draw cards to early, this tends to happen when the file loads in where one player starts with an additional card – ie in the middle of a cycle
* Cards sometimes get added to pile in a list, which breaks the scoring and causes a crash
* Some functions fail to return a properly formatted list due to not being able to flatten input
  + Some sets fail to capture but mark as they succeeded due to this reason. This puts the game into an error state. Game does NOT crash, however there is no way to return to a normal game state

# Feature Report:

## Non-Implemented:

* + - Build Owners
    - Build enforcement rules
      * Such as not trailing
      * Playing target card and not taking the build
      * They can be treated as a normal card for making other bulds
    - Ai doesn’t make build to capture sets
      * There are functions to find sets, however the AI cannot parse the returned data structures. In order for it to do it, it needs to remove nil values, and remove dupes

## Implemented:

* A way for the Ai to find sets it can make, however the AI does not use this knowledge do to failing in its parsing

# How to run

The program should run in any common lisp interpreter, however it was test in clisp. In order to run the program, open a command prompt in the same directory as the file is located, and run with command “clisp Casino.lsp” or enter the clisp interpreter in the directory and execute command “(load :Casino.lsp)

# Data-Structures

## Round parameters:

A list in the form

( firstPlayer deck tableCards humanHand humanPile compHand compPile )

Where:

* firstPlayer is an int, 0 or 1. Human is 0, computer is one
* Deck is a list of cards where new cards get dealt from
* tableCards is a list of Cards that are the table
* HumanHand/ComputerHand which are cards in the hand
* HumanPile/ComputerPile which are cards in the pile for that player

This is the driving “master list” of parameters, which gets passed to functions, either in its entirety, or in pieces.

# Log:

## October 3rd- 4th:

* Finding and setting up a lisp environment

## October 5th:

* Added ability to get Yes or no input from user (1 hour)
* Created skeleton for entering into the tournament and checking score (1 hour)

## October 6th:

* Multiple failed attempts to use lisp for anything productive and decided to do generic practice to reinforce knowledge of lisp syntax, no work was saved from the project

## October 7th

* Implemented ability to check scores and see if anyone won (.75 hours)
* Coin flip implemented (1 hour)
* Created skeleton of round loop, which includes coin flip, and score check (.1 hour)
* Able to generate a deck of cards in sequential order (.2 hours)
  + This was as a list of list symbols

## October 8th

* + - Can deal four cards to each hand and remove them from the deck (.5 hours)
    - Created function to print out the board (.5 hours)
    - Able to prompt the user for the action they want to take and can validate them (1 hour)

## October 9th:

* + - Rewrote the main loop to allow for it to be recursively called (2 hours)
    - Implemented trailing for the human player (.75 hours)

## October 11th:

* Program is able to shuffle the deck after it is created (1 hours)
* Able to trail any card in the hand instead of the first (1.5 hours)
  + This required use of two helper functions, 1 to get the numeric input and validate it, and the other to remove the card from the hand
* Can get a card’s symbol (.1 hour)
* Create skeleton for capturing (getting the user input) (.5 hour)

## October 12th:

* Fixed getNumericInput not validating in some cases, mainly strings (.5 hours)
* Added a function to find and remove matching cards (1.5 hours)
* Added a doPlayerMove helper function which is responsible for managing the pile and is a helper function to doCycle (.25 hours)

## October 13th:

* Able to trail from both hands (.25 hours)
* Added check for empty hands, which will allow for card re dealing (.2 hours)
* Fixed an infinite loop which occurred if the hand contained a card of nil before being redealt (.5 hours)

## October 14th:

* Program is able to read in a save file (.75 hour)
* Created functions to parse the data that is loaded in (.2 hours)
* Able to load into the board with the save data (1.5 hours)

## October 15th:

* Converted cards and all functions to uses symbols instead of lists (2 hours)

## October 16th:

* Program prompts again on invalid input (.2 hours)
* Changed user input to be one higher than the index for usability (.2 hours)
* Added helper functions to convert symbol to number and back again (1 hour)
* Able to find sets programmatically in ideal circumstances (2 hours)

## October 17th:

* Fixed formatting of the sets (1 hour)
* Had to track down functions which were missed in the conversation of symbols to lists (.75 hours)

## October 18th

* Updated return values of multiple functions to be consistent (1.25 hours)
* Added Skeletons for set capture functions (1 hour)

## October 19th

* Display the menu before each turn, and have it be different for each player (.75 hours)
* Program can exit from user choice (.1 hours)
* Program can open and write some data to the save file (1 hour)
* Able to get multiple inputs from the user (1 hours)
* Able for human to create a build (1 hour)

## October 20th:

* Builds remove the cards which were consumed in the build (2 hours)
* Able to check if a card is in a list (.1 hour)
* Able to get indices that are not selected for a table (1 hour)
* Able to make a build of N cards (1.25 hours)
* Can get the card Symbol of a build (2 hours)
* Can now capture builds (.75 hours)

## October 21st:

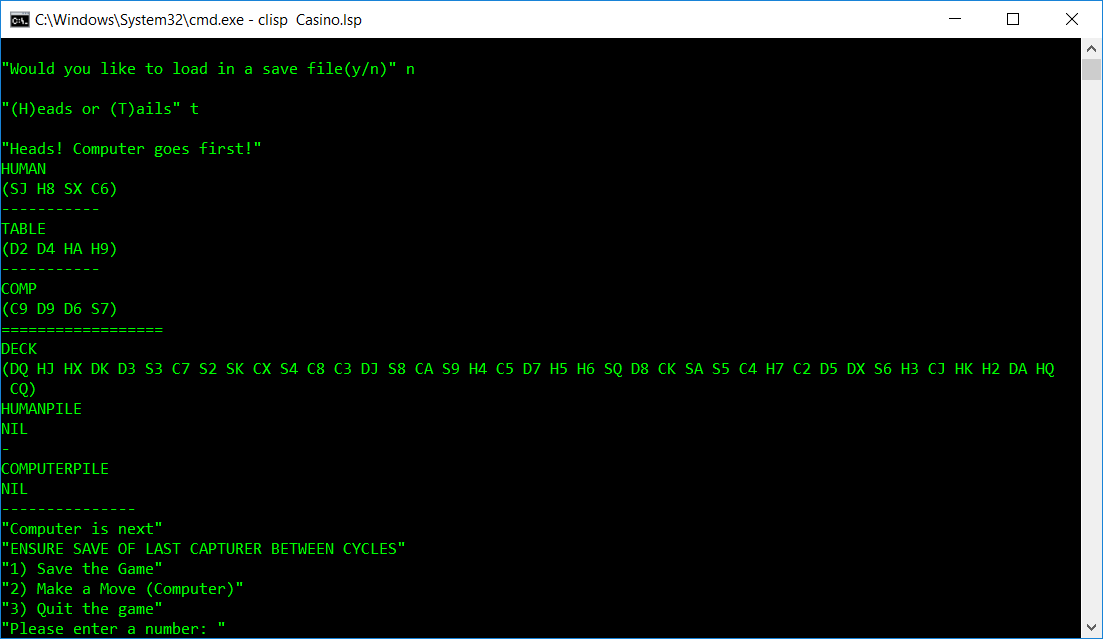
* Changed save file format (.2 hours)
* Added all scoring helper functions (1.5 hours)
* Tournament now properly loops for new rounds (.5 hours)
* Reorganized the file and added comments per the coding standards (2.25 hours)

## October 22nd:

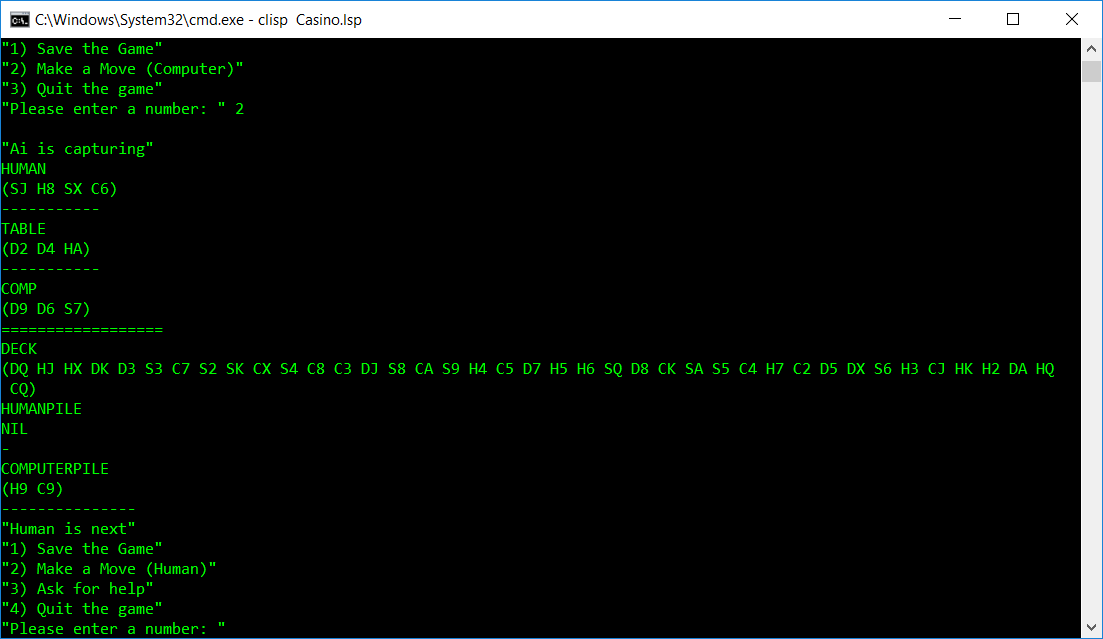
* Flip coin returns numeric instead of symbol (.25 hours)
* Fixed bug in doCycle which caused the hands to switch in certain conditions (1 hour)
* All info is added to the saveFile

# Screenshots:

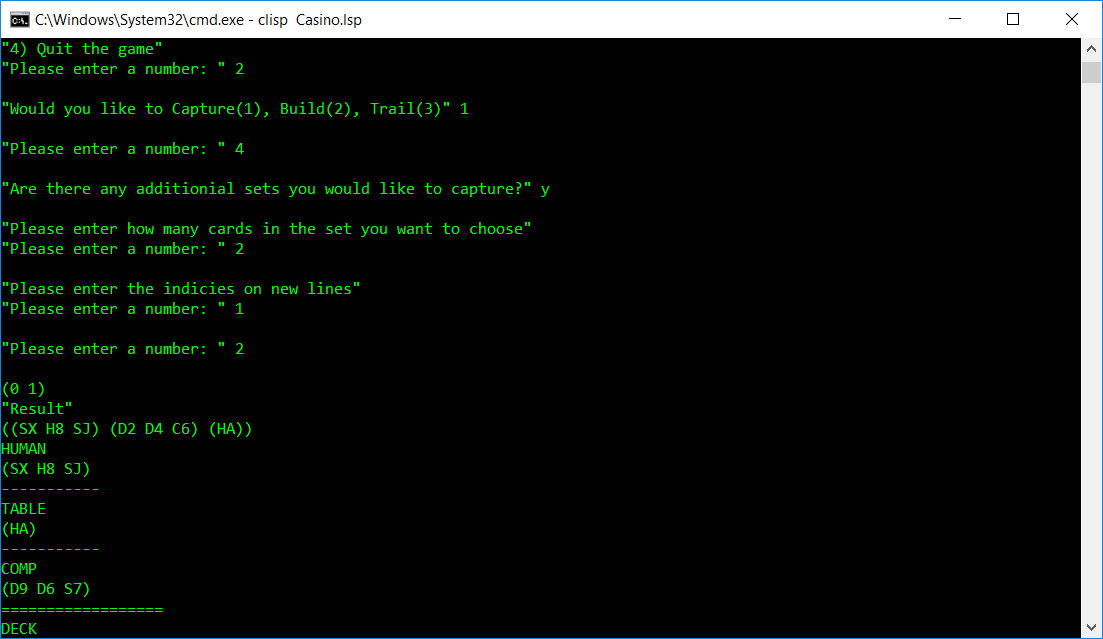
## Load-in:

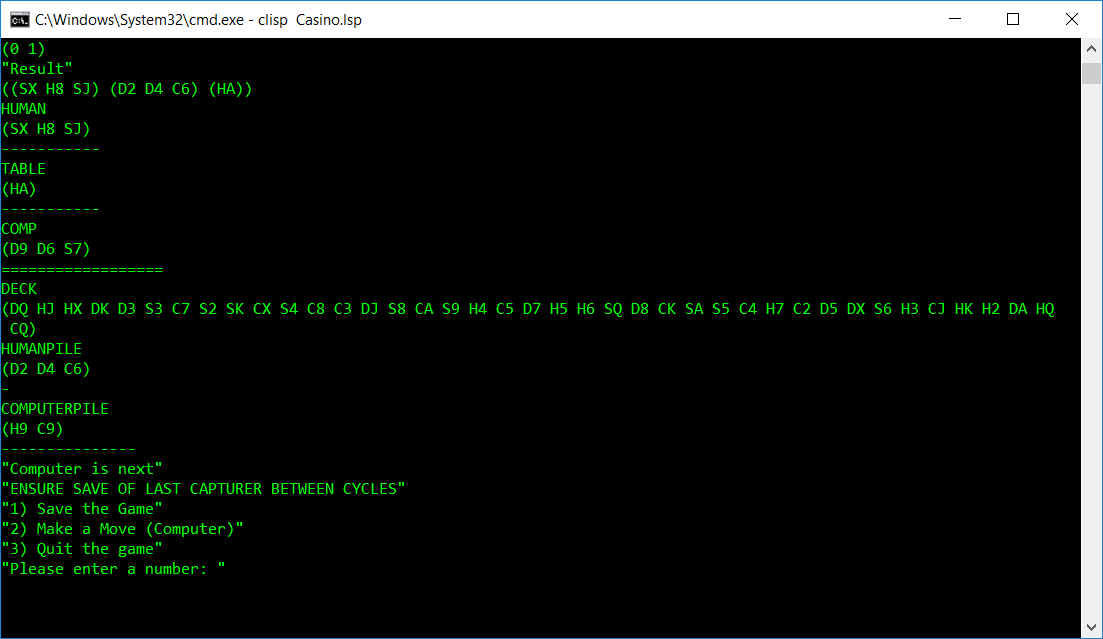


## Ai Capture:

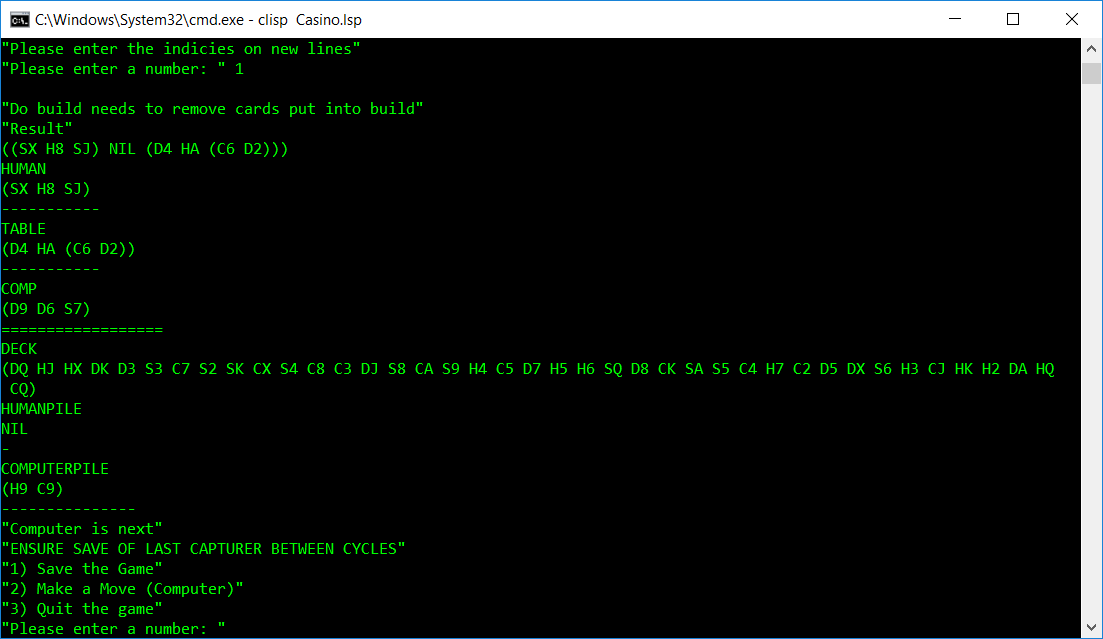


## Set Input and capture:

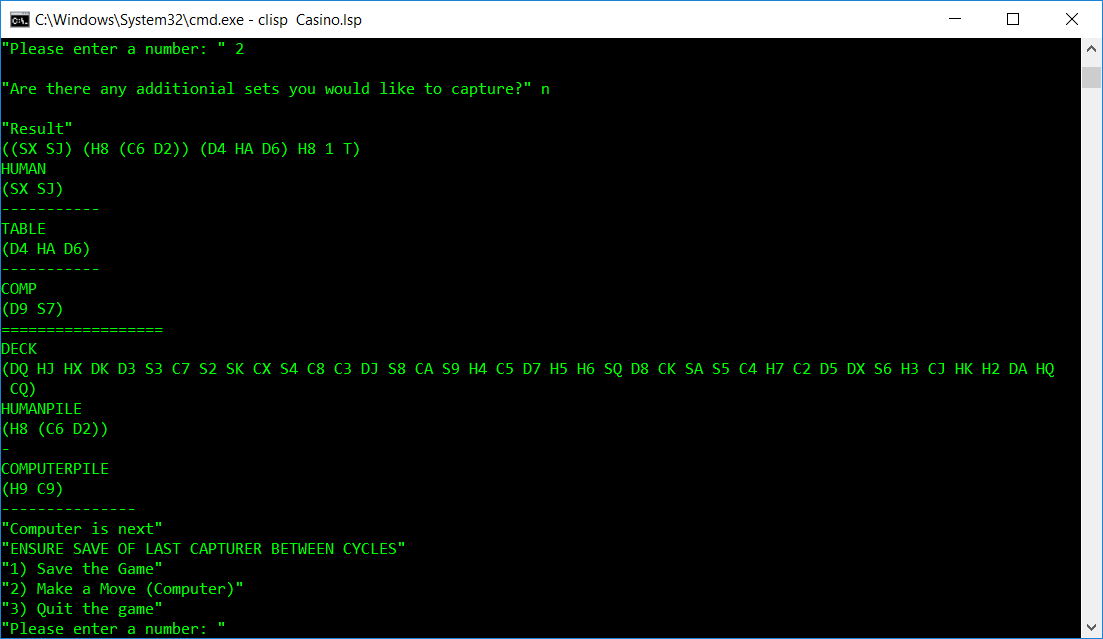




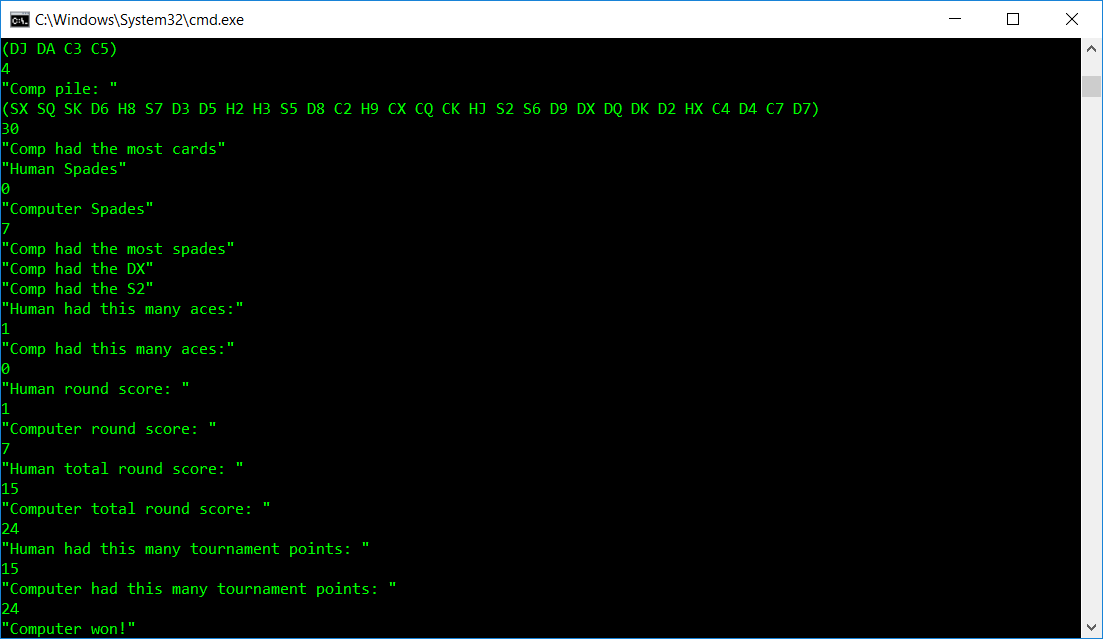
## Build being created:



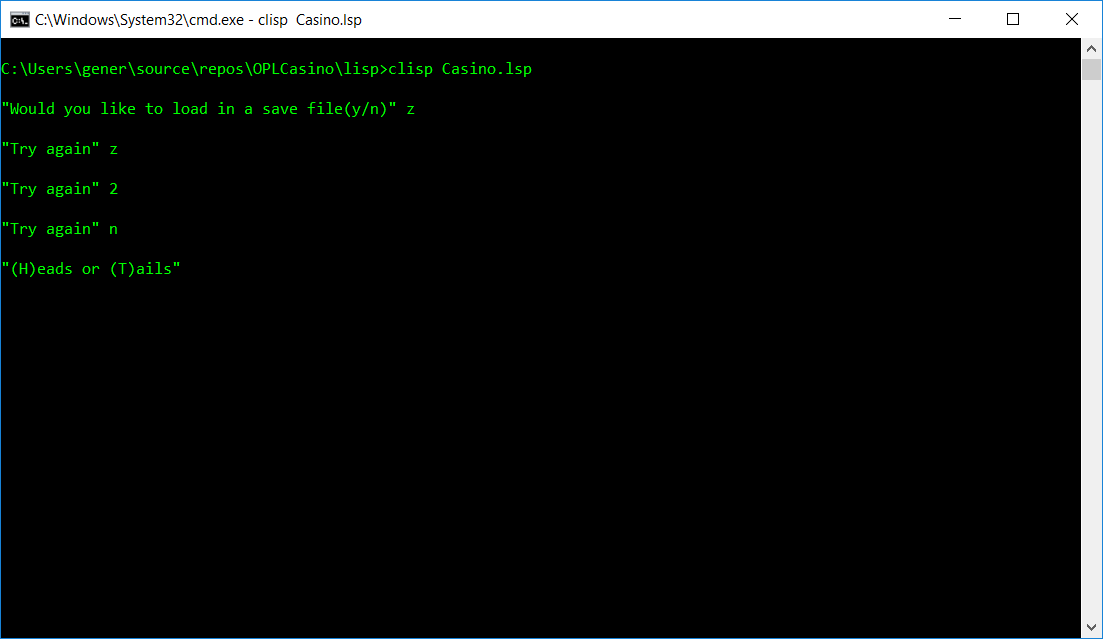
## Capturing a build:



## Scoring (the first list is the human’s pile):



## Yes/No Validation:



## Numeric Validation:

