There are three saved input files in total for the demo (*testing_mode.in* for the testing mode, *seeds.in* for the normal mode with a fixed seed, and *bonus.in* for showing bonus features). Details of how to run the program with these saved input files are provided in the three sections below. Note that in the following, comments after "//" explains what the saved input tests/demonstrates, and the content in the form "//..." does not actually present in the saved input files.

Testing Mode

The demo for a complete game in testing mode (which demonstrates that the program meets all requirements of the project and how the program handles some invalid inputs whose behaviors are not specified in the project description) can be run with the command line argument *-testing* and saved input file *testing_mode.in* contained in *hydra.zip*, by using the following command:

```
./hydra -testing <testing_mode.in
```

The saved input file *testing_mode.in* is as follows:

? // If the input for the number of players invalid (not a number or a number not greater than 1), the program ignores the input and prints "Invalid input! Input again?"

2 // Input the number of players (which is valid in this case)

Joker // Choose the first head

// Player enters any line (a blank line in this case) to continue play (Note that the line must be a full line, i.e., with a new line character immediately preceding it and ends with a new line character. If inputting from the keyboard, then after the prompt "Player a, it is your turn.", simply inputting 0 or more characters and pressing "Enter" will suffice.)

5 // Choose the value of the card Player 2 holds

h // Choose the suit of the card Player 2 holds (Lowercase character for suit is also accepted. Note that the two inputs for value and suit must be separated by one or more whitespaces. If inputting from the keyboard, choosing the value and suit of the card is typically done by seeing "Card value?", inputting the value, pressing "Enter", seeing "Suit?", inputting the suit, and then pressing "Enter" again. To save space in the following, the inputs for choosing the value and suit of the card are written on the same line.)

0 // Invalid move (the program ignores the move and prints the interactive prompt again)

5 H // Choose the value and suit of the card that Player 2 holds again after an invalid move

1 // Cutting off a head

k D // Choose the value and suit of the new head (lowercase character for value is also accepted)

2 D // Choose the value and suit of the other new head

OK! I am ready! // Player enters any line (a non-blank line in this case) to continue play

1 // Invalid card value, the program ignores the invalid input and repeats the question again

B! // Invalid card value, the program ignores the invalid input and repeats the question again

5 // Valid card value

A // Invalid card suit, the program ignores the invalid input and repeats the question again

S // Valid card suit

2 // Action 1

5 S // Choose the value and suit of the card

2 // Action 2

// Player enters any line (a blank line in this case) to continue play

5 C // Choose the value and suit of the card

2 // Action 2

// Player enters any line (a blank line in this case) to continue play

K C // Choose the value and suit of the card

0 // Action 4

A C // Choose the value and suit of the card

3 // Action 1

// Player enters any line (a blank line in this case) to continue play

A D // Choose the value and suit of the card

3 // Action 2

// Player enters any line (a blank line in this case) to continue play

9 D // Choose the value and suit of the card

0 // Action 4

8 C // Choose the value and suit of the card

0 // Action 5

7 C // Choose the value and suit of the card

0 // Invalid move since the card has been swapped once (what to do in this case is not specified in the project description. My program has the rule that the player must play the card that he/she retrieved by swapping with the reserve to prevent the player from swapping with the reserve back and forth forever. And the program treats this invalid move as other invalid moves, ignoring this move and prompting again.)

7 C // Choose the value and suit of the card

3 // Action 3

// Player enters any line (a blank line in this case) to continue play

J D // Choose the value and suit of the card

MoVe??? // Invalid input for move which is not a number (the program ignores the invalid input and prints “Invalid character!” for each invalid character)

1 // Invalid move which is a number greater than 0 but less than the first current head (the program simply ignores the invalid move and prompts again as specified in the project description)

J D // Choose the value and suit of the card

5 // Invalid move which is a number greater than the number of the last current head (the program simply ignores the invalid move and prompts again as specified in the project description)

J S // Choose the value and suit of the card

2 // Invalid move because there is no valid move with that head and it is also invalid to cut off a head (on Piazza the instructor says that it is up to us to determine whether a player is allowed to cut off a head when the reserve is empty, i.e., the player has the chance to perform Action 4. In my implementation, the player cannot cut off a head if the reserve is empty)

J S // Choose the value and suit of the card

0 // Action 4

Joker // Choose the value and suit of the card

2 // Select a head

8 // Cutting off a head (Joker can be used to cut off a head in my implementation)

JOKER // Choose the value and suit of the new head (both lowercase and uppercase for each character of Joker will be accepted by the program)

```
A S    // Choose the value and suit of the other new head
       // Player enters any line (a blank line in this case) to continue play
Joker    // Choose the value and suit of the card
-1     // Invalid move which is a number less than the first current (the program simply ignores the invalid
         move and prompts again)
Joker    // Choose the value and suit of the card
4      // Select a head
3      // Invalid value for the joker for the given play (the play is ignored and the turn repeats)
Joker    // Choose the value and suit of the card
3      // Select a head
8      // Invalid value for the joker for the given play (the play is ignored and the turn repeats)
Joker    // Choose the value and suit of the card
3      // Select a head
5      // A valid value for the joker (perform Action 1)
Joker    // Choose the value and suit of the card
0     // A valid value for the joker (perform Action 4)
Joker    // Choose the value and suit of the card
0     // A valid value for the joker (perform Action 5)
Joker    // Choose the value and suit of the card
4      // Select a head
2      // A valid value for the joker (perform Action 2)
       // Player enters any line (a blank line in this case) to continue play
Joker    // Choose the value and suit of the card
3      // Select a head
5      // A valid value (perform Action 2)
       // Player enters any line (a blank line in this case) to continue play
Joker    // Choose the value and suit of the card
4      // Select a head
1      // The input itself is not a card valid value (what to do in this case is not specified in the project
         description, my program will ignore the invalid input and repeat the question "Joker value?" again
         until it receives a valid response. Note that this case has nothing to do about whether the value is
         valid for the given play, as the input itself is not a valid card value at all)
B!     // Also not a valid card value, the program ignores the invalid input and repeats the question again
A      // A valid value (perform Action 1)
Joker    // Choose the value and suit of the card
5      // Select a head
10     // A valid value (perform Action 3)
Joker    // Choose the value and suit of the card
4      // Select a head
A     // A valid value (perform Action 2)
       // Player enters any line (a blank line in this case) to continue play
A D    // Choose the value and suit of the card
4     // Action 2
       // Player enters any line (a blank line in this case) to continue play (The rest of this saved input file
```

is mainly just for playing the game until one player wins, so there are not many things to pay attention to. The three places that need attention are marked red.)

```
A D    // Choose the value and suit of the card
0      // Action 4
A D    // Choose the value and suit of the card
0      // Action 5
4 H    // Choose the value and suit of the card
3      // Action 1
3 H    // Choose the value and suit of the card
3      // Action 1
       // Player enters any line (a blank line in this case) to continue play
2 H    // Choose the value and suit of the card
3      // Action 1
A H    // Choose the value and suit of the card
3      // Action 1
K H    // Choose the value and suit of the card
3      // Action 3
       // Player enters any line (a blank line in this case) to continue play
Q H    // Choose the value and suit of the card
3      // Action 1
J H    // Choose the value and suit of the card
3      // Action 1
10 H    // Choose the value and suit of the card
3      // Action 1
       // Player enters any line (a blank line in this case) to continue play
9 H    // Choose the value and suit of the card
3      // Action 1
8 H    // Choose the value and suit of the card
3      // Action 1
7 H    // Choose the value and suit of the card
3      // Action 1
       // Player enters any line (a blank line in this case) to continue play
6 H    // Choose the value and suit of the card
3      // Action 1
5 H    // Choose the value and suit of the card
3      // Action 1
4 H    // Choose the value and suit of the card
3      // Action 1
       // Player enters any line (a blank line in this case) to continue play
2 D    // Choose the value and suit of the card
0      // Action 4
Q C    // Choose the value and suit of the card
4      // Action 3
K S    // Choose the value and suit of the card
```

```
3      // Cutting off a head
J S      // Choose the value and suit of the new head
10 S    // Choose the value and suit of the other new head
       // Player enters any line (a blank line in this case) to continue play
J D    // Choose the value and suit of the card
4      // Action 1
10 D    // Choose the value and suit of the card
4      // Action 1
9 D    // Choose the value and suit of the card
4      // Action 1
8 D    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
7 D    // Choose the value and suit of the card
4      // Action 1
6 D    // Choose the value and suit of the card
4      // Action 1
5 D    // Choose the value and suit of the card
4      // Action 1
4 D    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
3 D    // Choose the value and suit of the card
4      // Action 1
2 D    // Choose the value and suit of the card
4      // Action 1
9 S    // Choose the value and suit of the card
5      // Action 1
8 S    // Choose the value and suit of the card
5      // Action 1
       // Player enters any line (a blank line in this case) to continue play
5 S    // Choose the value and suit of the card
0      // Action 4
7 S    // Choose the value and suit of the card
5      // Action 1
6 S    // Choose the value and suit of the card
5      // Action 1
5 S    // Choose the value and suit of the card
5      // Action 1
       // Player enters any line (a blank line in this case) to continue play
4 S    // Choose the value and suit of the card
5      // Action 1
3 S    // Choose the value and suit of the card
5      // Action 1
```

```
2 S    // Choose the value and suit of the card
5      // Action 1
A S    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
K S    // Choose the value and suit of the card
4      // Action 3
Q S    // Choose the value and suit of the card
4      // Action 1
J S    // Choose the value and suit of the card
4      // Action 1
10 S    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
10 C    // Choose the value and suit of the card
4    // Action 2
       // Player enters any line (a blank line in this case) to continue play
4 D    // Choose the value and suit of the card
0      // Action 4
9 C    // Choose the value and suit of the card
4      // Action 1
8 C    // Choose the value and suit of the card
4      // Action 1
7 C    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
6 C    // Choose the value and suit of the card
4      // Action 1
5 C    // Choose the value and suit of the card
4      // Action 1
4 C    // Choose the value and suit of the card
4      // Action 1
3 C    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
2 C    // Choose the value and suit of the card
4      // Action 1
A C    // Choose the value and suit of the card
4      // Action 1
K C    // Choose the value and suit of the card
4      // Action 3
Q C    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
```

```
J C    // Choose the value and suit of the card
4      // Action 1
10 C    // Choose the value and suit of the card
4      // Action 1
9 C    // Choose the value and suit of the card
4      // Action 1
8 C    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
7 C    // Choose the value and suit of the card
4      // Action 1
6 C    // Choose the value and suit of the card
4      // Action 1
5 C    // Choose the value and suit of the card
4      // Action 1
4 C    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
4 H    // Choose the value and suit of the card
4      // Action 2
       // Player enters any line (a blank line in this case) to continue play
3 C    // Choose the value and suit of the card
4      // Action 1
2 C    // Choose the value and suit of the card
4      // Action 1
A C    // Choose the value and suit of the card
4      // Action 1
K C    // Choose the value and suit of the card
4      // Action 3
       // Player enters any line (a blank line in this case) to continue play
K H    // Choose the value and suit of the card
4      // Action 2
       // Player enters any line (a blank line in this case) to continue play
10 C    // Choose the value and suit of the card
4      // Action 1
9 C    // Choose the value and suit of the card
4      // Action 1
8 C    // Choose the value and suit of the card
4      // Action 1
7 C    // Choose the value and suit of the card
4      // Action 1
       // Player enters any line (a blank line in this case) to continue play
7 C    // Choose the value and suit of the card
4      // Action 2
```

```
     // Player enters any line (a blank line in this case) to continue play
6 C   // Choose the value and suit of the card
4    // Action 1
5 C   // Choose the value and suit of the card
4    // Action 1
4 C   // Choose the value and suit of the card
4    // Action 1
3 C   // Choose the value and suit of the card
4    // Action 1
     // Player enters any line (a blank line in this case) to continue play
2 C   // Choose the value and suit of the card
4    // Action 1
A C   // Choose the value and suit of the card
4    // Action 1
J C   // Choose the value and suit of the card
4    // Action 3
10 C   // Choose the value and suit of the card
4    // Action 1
     // Player enters any line (a blank line in this case) to continue play
K S   // Choose the value and suit of the card (discard pile is shuffled to form a new draw pile)
0     // Action 4
K S   // Choose the value and suit of the card
4     // Cutting off a head
5 D     // Choose the value and suit of the new head
5 D     // Choose the value and suit of the other new head
     // Player enters any line (a blank line in this case) to continue play
9 C   // Choose the value and suit of the card
7    // Action 1
8 C   // Choose the value and suit of the card
7    // Action 1
7 C   // Choose the value and suit of the card
7    // Action 1
6 C   // Choose the value and suit of the card
7    // Action 1
5 C   // Choose the value and suit of the card
7    // Action 1
     // Player enters any line (a blank line in this case) to continue play
K S   // Choose the value and suit of the card
0     // Action 4
K S   // Choose the value and suit of the card
5     // Cutting off a head
J H    // Choose the value and suit of the new head
8 S    // Choose the value and suit of the other new head
     // Player enters any line (a blank line in this case) to continue play
```

9 C // Choose the value and suit of the card
0 // Action 4
9 C // Choose the value and suit of the card
10 // Action 1
K H // Choose the value and suit of the card
6 // Cutting off a head
3 S // Choose the value and suit of the new head
(discard pile is shuffled to form a new draw pile to create new heads)
4 S // Choose the value and suit of the other new head
// Player enters any line (a blank line in this case) to continue play
5 D // Choose the value and suit of the card
7 // Action 2
// Player enters any line (a blank line in this case) to continue play
5 S // Choose the value and suit of the card
8 // Action 2 (Player 1 wins and the program quits)

Seed for Random Generation

The program can take an optional command line argument, as a number, that represent a seed for random generation. The demo for the program with a fixed seed 315 can be run with the command line arguments *315* and saved input file *seeds.in* contained in *hydra.zip*, by using the following command:

```
./hydra 315 <seed.in
```

The saved input file *seed.in* is as follows (Since the above demo in testing mode has already shown that the game meets all requirements, this demo is mainly for showing that the random generation also works. The saved input *seeds.in* does not play to the end of the game, but should be enough to show that the game works properly in normal mode with random generation):

3 // Input the number of players
// Player enters any line (a blank line in this case) to continue play
1 // Action 1
// Player enters any line (a blank line in this case) to continue play
1 // Select a head
A // A valid value for the joker (perform Action 2)
// Player enters any line (a blank line in this case) to continue play
1 // Action 3
// Player enters any line (a blank line in this case) to continue play
1 // Action 1
// Player enters any line (a blank line in this case) to continue play
1 // Action 1
// Player enters any line (a blank line in this case) to continue play
1 // Action 3
// Player enters any line (a blank line in this case) to continue play
1 // Cutting off a head
// Player enters any line (a blank line in this case) to continue play
0 // Action 4

```
0	// Action 5
3	// Action 2
	  // Player enters any line (a blank line in this case) to continue play
3	// Action 1
0	// Action 4
	  // Player enters any line (a blank line in this case) to continue play
3	// Action 1
3	// Action 1
	  // Player enters any line (a blank line in this case) to continue play
2	// Action 1
0	// Action 4
	  // Player enters any line (a blank line in this case) to continue play
3	// Action 1
0	// Action 4
	  // Player enters any line (a blank line in this case) to continue play
0	// Action 4
2	// Cutting off a head
```

Extra Credit Features

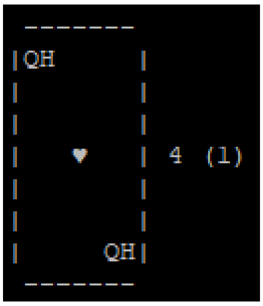
There are four extra credit features in total:

1. The entire project is completed, without leaks, and without explicitly managing memory: all memory management is handled via STL containers and smart pointers.

2. Proper grammar: the enhanced game says the player is holding “an A” or “an 8” instead of “a A” or “a 8”.

3. House rule: the game with enhancements allows players to cut off a head even if there is a valid play.

4. Better display: the “table” is drawn more nicely than a list of heads using ASCII. Each head is printed in the following form:



```
 -------
|QH     |
|       |
|       |
|   ♥   | 4 (1)
|       |
|       |
|     QH|
 -------
```

In the above example, “Q” is the value of the card, “H” is the suit of the card, the heart symbol in the middle is the symbol representing the suit of the card, “4” is the number of the head, and “1” is the total number of cards in the head. Cards of suite S, H, C, or D are displayed with corresponding suit symbols in the middle, and jokers are displayed with the character J in the middle.

The first enhancement can be witnessed by checking the code. The other three enhancements can be witnessed by running the program with the command line arguments *-testing* and *-enablebonus* and saved input file *bonus.in* contained in *hydra.zip*, by using the following command:

./hydra -testing -enablebonus <bonus.in

The saved input file *bonus.in* is as follows (it does not play to the end of the game, but should be enough to demonstrate all the enhancements):

2 // Input the number of players

Joker // Choose the value and suit of the new head (note how this head is displayed)

// Player enters any line (a blank line in this case) to continue play

8 C // Choose the value and suit of the card (the game will say the player is holding "an 8C")

1 // Cutting off a head

5 H // Choose the value and suit of the new head (note how this head is displayed)

J S // Choose the value and suit of the other new head (note how this head is displayed)

// Player enters any line (a blank line in this case) to continue play

A D // Choose the value and suit of the card (the game will say the player is holding "an AD")

3 // Action 1 (note how the head is displayed after placing the new card on)

5 C // Choose the value and suit of the card

2 // Action 2 (note how the head are displayed after placing the new card on)

// Player enters any line (a blank line in this case) to continue play

10 H // Choose the value and suit of the card

2 // Cutting off a head by inputting the number of the first active head (note that the player has somewhere else to play the card, i.e., the player can place the card on the head of number 3 by performing Action 3, or place the card into the reserve by performing Action 4, but the game with enhancements allows players to cut off a head even if there is a valid play)

K H // Choose the value and suit of the new head

3 S // Choose the value and suit of the other new head

In summary, the program can take three command-line arguments (i.e., -testing, -enablebonus, and/or a seed for random generation). These three command-line arguments can be in any order. For any invalid command-line argument, the program will print "Invalid Command Line Argument!" and simply ignore the invalid argument.