CS 4328.001
Operating Systems
Fall 2021
Dylan W. Ray(dwr48), Brett M. Owen(bmo23)
Programming Assignment #1: Pair War Report
Due: *Friday, October 8 @ 11:55 PM*

Design and Implementation:

This C++ program utilizes multithreading via POSIX threads to manage a dealer and players in the implementation of the pair war card game. In the main function of the program, a while loop is created for the purpose of implementing the required number of rounds (3). At the start of each round, a dealer thread is created, given a start routine argument of the function dealer(). This thread begins by shuffling the deck and then deals a card to each player's hand. Dealing cards is done in a round robin fashion starting from the players whose turn it is to begin the round. Fundamentally a player is represented by a global structure which holds an integer array of two values. The deck is a global object, or class. When declared, the deck creates 4 copies of 13 elements from [1, 13], a 52-element integer vector (1,1,1,1, 2,2,2,2, . . . ,13,13,13,13). Member functions of the deck class include shuffle(shuffle()), deal cards(draw()), add cards back to the deck (discard()), and look at the contents of the deck (getContents()). While the dealer thread executes, the main function calls a pthread_join() statement on the thread, immediately after its creation, and waits for it to finish.

When the dealer thread has exited, the main function then creates three player threads, each given a start routine argument of the function player() and an id argument. Next, main calls three pthread_join() statements for each player thread, halting its execution until each thread returns, indicating the round is over. Each player thread launches a while loop that only exits when either the round is won (WIN == true) or a draw case is met (TURN_COUNT == GAME_DRAW, only 10 turns per round are allowed) . Inside the while loop, each thread checks if it's their turn (id == TURN), and then draws a card form the deck. If the values of the two current cards in a player threads hand match, WIN is set to true, and the while loop is broken for each thread. If the player thread didn't win, it puts one of the two cards in its hand back in the deck and increments TURN_COUNT and TURN % 3, ending its own turn. When it's not the player thread's turn, the player thread sits in a busy waiting loop until their turn starts, another player wins, or the round is a draw.

After all the player threads exit, and before the next round starts, in main() The WIN variable is set back to false, TURN_COUNT is set to 0, ROUND_COUNT is incremented, and TURN is set to the value of the just finished ROUND_COUNT. This last step is performed to ensure that the proper player thread starts the next round, player 1->round 1, player 2->round 2, player 3->round 3. The while loop in main starts again, and the next round begins. This will repeat until three total rounds have been played.

Throughout the program's execution, each player's turn outputs to a log file and to the command line. Also, throughout the program, we have implemented the use of a single mutex lock (mutex_multiSeal) to protect writes to global variables, console/log outputs (but only after the turn loop breaks) and calls to the DECK class's member functions. Ideally there

would be one mutex lock for every item of shared memory for the purpose of performance, but one mutex lock will be acceptable for this program even with the slight loss of performance.

Our code style, for the most part, follows typical C++ style guidelines. By convention the log files will always be named "pair_war_log_<seed>.txt" and included at the top of the file will be a title along with the seed used.

Compilation Instructions:

So many of the problems we had when debugging the program were solved when we had removed any optimization flags from our compilation. Adding compiler optimization created hangs in our busy waiting loops. The compiler command we ran which gave us the correct output is this:

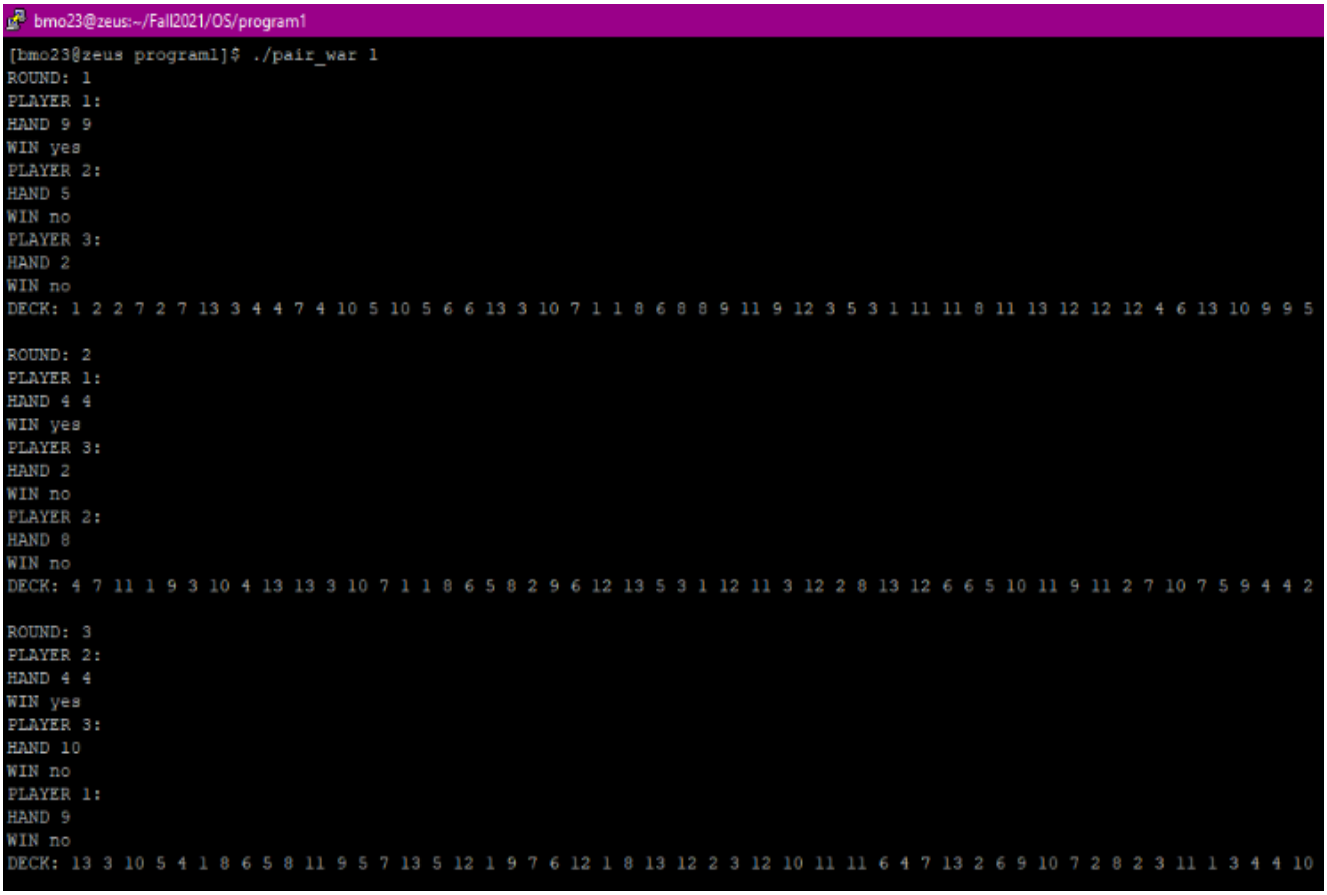
```
$ g++ -pthread -std=c++11 -o pair_war pair_war.cpp Deck.cpp

$ ./pair_war <seed>
```

When compiling ensure that files pair_war.cpp, Deck.cpp, and Deck.h are in the same directory. Also, ensure that only a single integer value is given as the seed, multiple command line arguments or not integer arguments will cause the program to throw and error and terminate. Compiling on the university Linux servers, Zeus/Eros, should perform correctly.

Program execution command line output:

Seed of 1:



Seed of 5:

```
bmo23@zeus:~/Fall2021/OS/program1
[bmo23@zeus program1]$ ./pair_war 5
ROUND: 1
PLAYER 3:
HAND 3 3
WIN yes
PLAYER 1:
HAND 2
WIN no
PLAYER 2:
HAND 1
WIN no
DECK: 4 9 13 4 6 5 11 8 7 6 5 6 7 10 7 13 8 10 4 9 9 4 1 12 8 5 10 2 12 10 11 11 5 3 12 2 13 9 13 11 8 2 1 1 6 12 7 3 3 3 2

ROUND: 2
PLAYER 2:
HAND 5
WIN no
PLAYER 3:
HAND 10
WIN no
PLAYER 1:
HAND 6
WIN no
DRAW
DECK: 10 2 3 9 1 4 6 10 4 9 12 8 5 10 12 3 7 11 11 13 8 1 1 2 8 13 11 1 2 4 8 6 12 11 3 5 9 12 13 13 6 9 2 7 5 4 3 7 7 5 10

ROUND: 3
PLAYER 3:
HAND 5
WIN no
PLAYER 1:
HAND 11
WIN no
PLAYER 2:
HAND 8
WIN no
DRAW
DECK: 13 9 13 7 7 1 11 8 4 1 10 8 13 6 4 2 5 8 10 6 11 6 12 9 12 5 12 3 11 2 7 5 6 1 3 7 10 2 10 4 1 3 12 2 9 3 4 13 9 5 11
```

Seed of 25:

```
bmo23@zeus:~/Fall2021/OS/program1
[bmo23@zeus program1]$ ./pair_war 25
ROUND: 1
PLAYER 2:
HAND 2 2
WIN yes
PLAYER 3:
HAND 2
WIN no
PLAYER 1:
HAND 7
WIN no
DECK: 3 8 3 12 8 4 4 4 2 5 4 5 11 6 6 6 7 1 5 13 5 1 3 12 10 7 9 9 6 10 13 10 11 9 11 11 12 1 10 12 9 13 13 7 8 3 1 8 2 2 2

ROUND: 2
PLAYER 3:
HAND 4 4
WIN yes
PLAYER 1:
HAND 5
WIN no
PLAYER 2:
HAND 12
WIN no
DECK: 4 11 4 7 3 5 6 10 13 12 8 9 12 1 11 13 7 3 9 9 13 2 6 5 3 9 11 11 1 1 6 12 10 10 13 7 8 3 8 1 2 2 2 7 10 8 6 5 4 4 5

ROUND: 3
PLAYER 3:
HAND 8
WIN no
PLAYER 2:
HAND 13
WIN no
PLAYER 1:
HAND 12
WIN no
DRAW
DECK: 10 11 13 4 3 9 4 13 9 6 5 2 9 2 11 1 1 8 8 10 12 12 7 13 3 1 2 3 9 11 7 6 8 6 5 4 5 12 1 7 10 5 6 10 4 11 7 3 2 8 13
```

Seed of 50:

```
bmo23@zeus:~/Fall2021/OS/program1
[bmo23@zeus program1]$ ./pair_war 50
ROUND: 1
PLAYER 1:
HAND 4
WIN no
PLAYER 2:
HAND 1
WIN no
PLAYER 3:
HAND 2
WIN no
DRAW
DECK: 5 6 13 5 4 1 7 6 11 8 6 5 9 7 10 8 8 3 13 9 9 7 6 10 10 10 2 1 5 11 11 12 1 12 3 7 4 13 12 2 13 11 12 2 3 9 4 8 3 4 1

ROUND: 2
PLAYER 2:
HAND 5
WIN no
PLAYER 1:
HAND 5
WIN no
PLAYER 3:
HAND 11
WIN no
DRAW
DECK: 7 2 2 8 3 3 9 9 7 6 11 10 10 2 7 5 11 4 12 1 12 6 10 4 13 3 12 6 11 8 9 13 4 4 8 1 10 8 2 5 12 1 9 13 3 7 6 13 1 5 5

ROUND: 3
PLAYER 3:
HAND 13
WIN no
PLAYER 1:
HAND 6
WIN no
PLAYER 2:
HAND 2
WIN no
DRAW
DECK: 2 6 5 2 9 13 9 9 6 10 4 12 3 12 6 13 8 10 8 7 4 2 8 1 12 7 13 5 1 10 3 3 4 5 10 11 5 8 11 4 12 1 9 3 7 1 11 11 7 13 6
```

Seed of 100:

```
bmo23@zeus:~/Fall2021/OS/program1
[bmo23@zeus program1]$ ./pair_war 100
ROUND: 1
PLAYER 3:
HAND 3 3
WIN yes
PLAYER 1:
HAND 9
WIN no
PLAYER 2:
HAND 2
WIN no
DECK: 1 4 4 4 8 1 8 5 9 9 10 5 13 7 3 7 6 8 7 8 7 6 10 1 10 10 2 9 11 11 11 11 12 12 12 12 2 3 13 13 13 4 1 2 6 6 5 5 3 3 9

ROUND: 2
PLAYER 3:
HAND 5 5
WIN yes
PLAYER 2:
HAND 12
WIN no
PLAYER 1:
HAND 4
WIN no
DECK: 1 13 7 9 9 10 10 13 7 3 2 8 2 7 8 6 4 13 6 10 1 4 9 5 3 11 11 11 11 12 12 6 3 13 12 8 4 1 8 2 10 2 6 3 9 5 1 7 5 5 12

ROUND: 3
PLAYER 3:
HAND 4
WIN no
PLAYER 1:
HAND 8
WIN no
PLAYER 2:
HAND 8
WIN no
DRAW
DECK: 7 8 1 4 13 6 10 1 4 13 8 3 11 11 5 12 2 5 6 3 11 12 2 2 4 10 2 6 11 6 3 9 1 12 7 12 5 10 9 9 13 7 3 1 10 5 7 13 9 4 8
```