Alesha Ray
Due 10/4/2020

Pair-War

(Report)

Overview

This implementation of the pair-war game was written in an Object Oriented format. Classes were created for the Card Deck, the Dealer, the Players, the Global variables shared by all objects, and the threads needed to create the dealer and the players.

Starting with the **pThread_class** Class, we need this class in order to create the dealer and the players. This class includes: a default constructor; a virtual destructor which will allow all subclasses of this class to be destructed properly and not result in undefined; a function to start [or create] the thread (*start_thread*); a function to (*wait*) [or join] which makes the calling thread wait until execution of target thread has terminated; a virtual function that the thread will (*run*), which will be implemented in the subclasses of the this parent class; and a (*run_helper*) which calls the runner, and will be passed into the the function which starts the thread.

Next, there is the **Deck** class which handles the logic for the deck of cards. This class is implemented in a way that allows using a 1-dimensional array as one would use a deck of cards. Cards can be popped (*pop*) from the top of the deck, pushed (*push*) onto the bottom of the deck, (*set*) into a specific position within the deck, and received (*get*) from a specific position within the deck. The Deck class also allows a way to populate (*load_deck*) the deck with cards, get the number of cards (*size*) in the deck, view the values of the current cards on screen (*display_deck*), and print the value of the current card to the file (*print_deck*).

The **Dealer** has very few jobs. The only things the dealer is responsible for are to shuffle (*shuffle_cards*) and deal (*deal_cards*) the cards. This class, because it is a subclass of the pthread_class, also has a function to (*run*) the thread. Inside of the run function, the Dealer needs to access the deck (*access_deck*) once it becomes available for use.

The **Player**s are also subclasses of the pthreads_class and as such, this class also has functions to (*run)* and access the deck (*access_deck*) when it becomes available for use. The player's hand will technically only consist of one card. When the player draws a card (*draw_card*), they immediately compare the cards (*compare_cards*) and if they don't have a winning hand, they discard one of their cards (*discard_card*). This means the player will only need to keep one card in their hand at a time, therefore the Player class offers the (*set_hand*) and (*get_hand*) functions which allows for one card [int value] to be set and one card [int value] to be received. The Player class provides (*set_ID*) and (*get_ID*) methods so that it is easy to dictate and determine who currently has access to the deck of cards.

Since the point of the game is to implement threads, there is some data that will be shared across all functions; these are defined in the **globals** class. The deck can only be

accessed by one person at a time. Since the dealer and the players are created from two separate classes, this requires locking mechanisms: the (mutex_deck_availability_to_player) variable and the (mutex_deck_availability_to_dealer) variable. All threads will be required to meet the conditions to lock and unlock the deck using the (cond_deck_availability) variable. The dealer will then need to know when a player has met the condition of winning a round and the player will need a way of informing the dealer of this information, and thus the (cond_winner_found) variable is available. The other global variables are used to access the file that will be written to, the number of rounds being played, the current round, the indicator of which players turn it is to access the deck, the randomization seed, and the winner found variable.

Lastly, there is a **main** application that runs the program. The main application initializes the dealer and player objects (threads). The program first opens the file that will be written to and sets the seed for randomization. For each round, the winner_found flag is reset and the round is then played.

Alesha Ray
Due 10/4/2020

Run Instructions

The program is provided with a makefile that makes running the application simple. The makefile takes in a **seed** as a variable and then passes it into the main program as the the 2nd (index 1) argument. That seed is then used to randomize the random number generator. The makefile also allows for quick cleanup.

Making and running the program:

```
make run seed=value used as seed
```

This will create the binary directory, and place the executable file inside. It will then change into that directory and run the file. If

It is very important to follow the above example exactly, as the wrong seed may be entered into the makefile otherwise. If you run “make run” without a seed, the program will result in a “segmentation fault”.

- There are ***no spaces*** between the word “seed”, the “=” symbol, and the subsequent seed that will be passed in.
- After some trial and error, it was determined that any seed less than 1 million gives inconclusive results. For this reason, the program implements logic that automatically adds 1 million to all seeds that do not meet the minimum requirements.
- While several seeds were used to exhaust testing efforts, the seed most commonly used during testing was as follows:

```
seed=$(($(date +%s%N)/1000000))
```

- This seed takes the current date and time, converts it to the amount of nanoseconds that have elapsed since January 1, 1970, and divides by 1 million, so that the number will always fit into the “int” primitive type.
- This ensures that your seed will always be a different value.

Cleaning up the created files

```
make clean
```

This will remove all class object files, the log file created as a result of the program, the contents of the binary directory and the binary directory itself.

Alesha Ray
Due 10/4/2020

Screen Results

```
$ make run seed=$(($(date +%s%N)/1000000))
g++ -c src/main.cpp -o src/main.o -std=c++0x -I include
g++ -c lib/pthreads_class.cpp -o lib/pthreads_class.o -std=c++0x -I include
g++ -c lib/Deck.cpp -o lib/Deck.o -std=c++0x -I include
g++ -c lib/Player.cpp -o lib/Player.o -std=c++0x -I include
g++ -c lib/globals.cpp -o lib/globals.o -std=c++0x -I include
g++ -c lib/Dealer.cpp -o lib/Dealer.o -std=c++0x -I include
mkdir bin &&    ar rvs bin/cpukit.a lib/pthreads_class.o lib/Deck.o lib/Player.o lib/globals.o lib/Dealer.o
ar: creating bin/cpukit.a
a - lib/pthreads_class.o
a - lib/Deck.o
a - lib/Player.o
a - lib/globals.o
a - lib/Dealer.o
g++ -o bin/main src/main.o -pthread -L lib bin/cpukit.a
./bin/main 1601855535350

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

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

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

Alesha Ray
Due 10/4/2020

```
$ make run seed=$[[$(date +%s%N)/1000000]]
g++ -c src/main.cpp -o src/main.o -std=c++0x -I include
g++ -c lib/pthreads_class.cpp -o lib/pthreads_class.o -std=c++0x -I include
g++ -c lib/Deck.cpp -o lib/Deck.o -std=c++0x -I include
g++ -c lib/Player.cpp -o lib/Player.o -std=c++0x -I include
g++ -c lib/globals.cpp -o lib/globals.o -std=c++0x -I include
g++ -c lib/Dealer.cpp -o lib/Dealer.o -std=c++0x -I include
mkdir bin &&    ar rvs bin/cpukit.a lib/pthreads_class.o lib/Deck.o lib/Player.o lib/globals.o lib/Dealer.o
ar: creating bin/cpukit.a
a - lib/pthreads_class.o
a - lib/Deck.o
a - lib/Player.o
a - lib/globals.o
a - lib/Dealer.o
g++ -o bin/main src/main.o -pthread -L lib bin/cpukit.a
./bin/main 1601856644016

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

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

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

Alesha Ray
Due 10/4/2020

```
% make run seed=$((%(date +%s%N)/1000000))
g++ -c src/main.cpp -o src/main.o -std=c++0x -I include
g++ -c lib/pthreads_class.cpp -o lib/pthreads_class.o -std=c++0x -I include
g++ -c lib/Deck.cpp -o lib/Deck.o -std=c++0x -I include
g++ -c lib/Player.cpp -o lib/Player.o -std=c++0x -I include
g++ -c lib/globals.cpp -o lib/globals.o -std=c++0x -I include
g++ -c lib/Dealer.cpp -o lib/Dealer.o -std=c++0x -I include
mkdir bin &&    ar rvs bin/cpukit.a lib/pthreads_class.o lib/Deck.o lib/Player.o lib/globals.o lib/Dealer.o
ar: creating bin/cpukit.a
a - lib/pthreads_class.o
a - lib/Deck.o
a - lib/Player.o
a - lib/globals.o
a - lib/Dealer.o
g++ -o bin/main src/main.o -pthread -L lib bin/cpukit.a
./bin/main 1601856709622

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

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

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

Alesha Ray
Due 10/4/2020

```
$ make run seed=$(($(date +%s%N)/1000000))
g++ -c src/main.cpp -o src/main.o -std=c++0x -I include
g++ -c lib/pthreads_class.cpp -o lib/pthreads_class.o -std=c++0x -I include
g++ -c lib/Deck.cpp -o lib/Deck.o -std=c++0x -I include
g++ -c lib/Player.cpp -o lib/Player.o -std=c++0x -I include
g++ -c lib/globals.cpp -o lib/globals.o -std=c++0x -I include
g++ -c lib/Dealer.cpp -o lib/Dealer.o -std=c++0x -I include
mkdir bin &&    ar rvs bin/cpukit.a lib/pthreads_class.o lib/Deck.o lib/Player.o lib/globals.o lib/Dealer.o
ar: creating bin/cpukit.a
a - lib/pthreads_class.o
a - lib/Deck.o
a - lib/Player.o
a - lib/globals.o
a - lib/Dealer.o
g++ -o bin/main src/main.o -pthread -L lib bin/cpukit.a
./bin/main 1681856813391

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

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

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

Alesha Ray
Due 10/4/2020

```
$ make run seed=$((`date +%s%N`/1000000))
g++ -c src/main.cpp -o src/main.o -std=c++0x -I include
g++ -c lib/pthreads_class.cpp -o lib/pthreads_class.o -std=c++0x -I include
g++ -c lib/Deck.cpp -o lib/Deck.o -std=c++0x -I include
g++ -c lib/Player.cpp -o lib/Player.o -std=c++0x -I include
g++ -c lib/globals.cpp -o lib/globals.o -std=c++0x -I include
g++ -c lib/Dealer.cpp -o lib/Dealer.o -std=c++0x -I include
mkdir bin &&    ar rvs bin/cpukit.a lib/pthreads_class.o lib/Deck.o lib/Player.o lib/globals.o lib/Dealer.o
ar: creating bin/cpukit.a
a - lib/pthreads_class.o
a - lib/Deck.o
a - lib/Player.o
a - lib/globals.o
a - lib/Dealer.o
g++ -o bin/main src/main.o -pthread -L lib bin/cpukit.a
./bin/main 1601856947621

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

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

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