CSCE 1030 – Homework 6

Due: 11:59 PM on Wednesday, December 6, 2017

PROGRAM DESCRIPTION:

For this C++ program, you will edit your Homework 5 programming assignment to add the functionality to play a simplified version of the classic game of Stratego that will display to the screen. Unless so indicated, all of the requirements in Homework 5 hold in Homework 6. Note that there are some changes to this assignment!

You will take your solution from Homework 5 and edit it as follows:

1. You shall organize your program into three files:
   - **`hw6prgm.h`** will hold the include directives for the necessary libraries, any global constants, any enumerated data types, any type definitions, any structure definitions, and the list of function prototypes (i.e., function declarations).
   - **`hw6main.cpp`** will hold the local include directive for the header file as well as the `main` function for the program.
   - **`hw6func.cpp`** will hold the local include directive for the header file as well as all function definitions (not including `main`, of course) used in the program.
2. You will update the rules of the game to note what happens when equal pieces strike. Note the rules in the SAMPLE OUTPUT.
3. Define a structure (i.e., `struct`) that contains information about each piece associated with each location (i.e., square) on the game board. Specifically, this structure should contain the following three members: (1) an `enum` data type member that holds the rank (i.e., `FLAG`, `BOMB`, `MARSHAL`, etc.) of the piece on the board, (2) an `enum` data type member that holds the color of the piece on the board, and (3) a boolean variable that indicates whether the piece on the board is moveable or not. You will then change the data type of the game board array to be of this structure instead of the `enum` as required in Homework 5.
4. Update the function to display the game board by adding a boolean formal parameter used to indicate whether or not to "reveal" the solution with all pieces visible, or to display the current, active board with the computer's (i.e., `BLUE`) pieces hidden, showing some generic value '`X`' instead. The `RED` player's pieces must, obviously, be revealed for the user to play the game. You will call this function every time to display the updated board after each valid user move.
5. Add a boolean value-returning function to update the board after a valid move (refer to rules about how a moveable piece can move one position up/down or left/right on the board, but a non-moveable piece cannot be moved) so as to return a boolean value to indicate whether or not the game is over (i.e., if the player struck the opponent's `FLAG`, this function would return `true`; otherwise, it would return `false`). You will use this boolean result to determine if the game should continue (i.e., game is not over) or not (i.e., game is over). At a minimum, you will pass the game board as well as position information of the piece that is moving or striking to this function. You will display sufficient and meaningful

information about what is happening for each valid move (see SAMPLE OUTPUT for examples of what type of information is expected).

6. Inside the `main` function, you will add a loop to play the game until either the game is won (e.g., the player successfully struck the opponent's `FLAG`) or the player has selected to forfeit the game by typing "`QQ`" to end the game (e.g., the player no longer has any available moves). For each turn, the player will enter a set of coordinates corresponding to the current position (i.e., row and column) of the piece on the game board the player wants to move. If the player enters an invalid position (e.g., outside of the game board, a non-moveable piece, an empty square, the opponent's piece, etc.), you will provide a meaningful error message that an invalid position was entered and continue to prompt the player to enter the coordinates again until valid coordinates are entered. You will do similarly for the new position of the piece by validating the new position of the piece. If the player enters an invalid position (e.g., outside of the game board, a diagonal move, or a move of more than one space). You may assume that the player enters the position coordinates correctly as a letter and an integer, though one or both values may not be in the valid range defined. Following each turn, you will display an updated version of the board. Ultimately, the game will be over if the player strikes the opponent's `FLAG` (i.e., won the game) or the player forfeits the game by entering "QQ", in a case, for example, when the user has no more moves left. At the end of the game, you will display a meaningful message about whether the player won or lost the game as well as the final updated board with the position of all the pieces revealed.
7. Instead of a static two-dimensional array as was done in Homework 5, you shall declare and create a two-dimensional *dynamic* array in `main` to represent the 5-by-5 board for the `struct` data type you declared to hold the various values that a square can assume. Since the board array is not global, you will pass the board array to the needed functions as a pointer (actually, as a pointer to a pointer). You will need to make sure to return the memory for the two-dimensional array back to the freestore when your program is done using it (at the end).
8. You may use global `enum` types, type definitions, and structures. You may also have a limited number of global constants, such as the size of the game board, but the board array must be declared as a local variable that is then passed to the related functions.

You may assume that all input by the user is of the correct data type, though perhaps out of range. You should contact your instructor if there is any question about what is being asked for.

BONUS OPPORTUNITY: Up to 20 Points

For students who have completed all requirements for this program, you may add the following options to gain additional bonus points to your program score:

- You may add the option to "save" the game (perhaps using a sentinel value instead of the coordinates) so that it can be resumed at a later time. This would entail saving the game board (with all needed information indicated). This information may be written to a file in any format you choose. Then when the program is run again, you can check If the player wants to resume the previous game and, if so, open the file (if it exists) and load the data into memory to continue the game.
- The current requirements of the game assume a "passive" computer (i.e., `BLUE`) player, that is, only the `RED` player moves. You may add logic to allow the computer (i.e., `BLUE`) player to alternate turns with the `RED` player. This may be as simple as adding random, yet valid, moves, but more points will be awarded for more sophisticated moves and intelligence from the `BLUE` player.

Please note that unless you have completed all other requirements for this program no credit will be given for trying this extra credit. In other words, make sure your program is complete before attempting this extra credit.

REQUIREMENTS:

- Your code should be well documented in terms of comments. For example, good comments in general consist of a header (with your name, course section, date, and brief description), comments for <u>each</u> variable, and commented blocks of code.
- Your program source code should be named as detailed in the above requirements.
- Your program will be graded based largely on whether it works correctly on the CSE machines (e.g., `cse01`, `cse02`, …, `cse06`), so you should make sure that your program compiles and runs on a CSE machine.
- **This is an individual programming assignment that must be the sole work of the individual student.**

You may assume that all input will be of the appropriate data type, although the range (e.g., a positive integer) may not be valid. Please pay attention to the SAMPLE OUTPUT for specific details about the flow and input/output of the program.

You shall use techniques and concepts discussed in class – you are not to use global variables (limited constants, structures, `enums`, and type definitions are OK), `goto` statements, or other items specifically not recommended in this class.

DESIGN (ALGORITHM):

On a piece of paper (or word processor), write down the algorithm, or sequence of steps, that you will use to solve the problem. You may think of this as a “recipe” for someone else to follow. Continue to refine your “recipe” until it is clear and deterministically solves the problem. Be sure to include the steps for prompting for input, performing calculations, and displaying output.

CSCE 1030 – Homework 6

Due: 11:59 PM on Wednesday, December 6, 2017

You should attempt to solve the problem by hand first (using a calculator as needed) to work out what the answer should be for a few sets of inputs, especially in how the RED player interacts with the game board. This will also help in designing any loops that will process the two-dimensional array.

Type these steps and any calculations into a document (i.e., Word, text, or PDF) that will be submitted along with your source code. Note that if you do any work by hand, images (such as pictures) may be used, but they must be clear and easily readable. This document shall contain both the algorithm and any supporting hand-calculations you used in verifying your results.

SAMPLE OUTPUT (input shown in **bold green**):

```
$ ./a.out
              +----------------------------------------------+
              |       Computer Science and Engineering       |
              |        CSCE 1030 - Computer Science I        |
              |   Student Name     EUID    euid@my.unt.edu   |
              +----------------------------------------------+

           W e l c o m e   t o   1 0 3 0   S t r a t e g o

------------------------------------------------------------------------
This program will set up a 5x5 game board for a 1030 version of the game
of Stratego. One player will compete against the computer, each assigned
10 total pieces consisting of the following:
  1 FLAG (F)
  3 BOMB (B)
  1 MARSHAL (1) or GENERAL (2)
  3 COLONEL (3), MAJOR (4), CAPTAIN (5), LIEUTENANT (6), or SERGEANT (7)
  1 MINER (8)
  1 SPY (S)

GENERAL RULES:
--------------
For the most part, the game will follow the standard Stratego rules, al-
though there are some exceptions.
1. Both players (BLUE and RED) will have all of their 10 game pieces as-
   signed randomly with the only requirement being that the FLAG must be
   placed in the back row. RED will start the game first.
2. Higher ranked pieces can capture lower ranked pieces in the following
   order: 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> S, meaning that 1 (the
   MARSHAL) can remove 2 (the GENERAL) and so forth. The MINER (8) piece
   may strike a BOMB and remove it to occupy the now unoccupied space. A
   SPY (S), although the lowest ranked piece, may remove the MARSHAL (1)
   or the GENERAL (2). When pieces have equal rank, then both pieces are
   removed.
3. The FLAG and BOMBs are not moveable while all of the other pieces may
   move one square at a time forward, backward, or sideward, but not di-
   agonally to an open square.
4. A player must either move or strike on his/her turn.
5. The game ends when a player strikes his/her opponent's flag.
------------------------------------------------------------------------
```

```
Initializing game board...
Assigning BLUE pieces to board...
Assigning RED  pieces to board...
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |           |
D | 8 5 B B 1 |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): S1
Error: Invalid row location S. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C14
Error: Invalid coordinates C14. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D8
Error: Invalid column location 8. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D3
Error: Invalid piece or not moveable at D3. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C2
Error: Invalid piece or not moveable at C2. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B2
Error: Invalid piece or not moveable at B2. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D2
RED MOVE: Enter new coordinates of piece at D2: S1
Error: Invalid row location S. Try again...
RED MOVE: Enter new coordinates of piece at D2: F2
Error: Invalid row location F. Try again...
RED MOVE: Enter new coordinates of piece at D2: C25
Error: Invalid coordinates C25. Try again...
RED MOVE: Enter new coordinates of piece at D2: C3
Error: Invalid piece move D2 to C3. Try again...
RED MOVE: Enter new coordinates of piece at D2: B2
Error: Invalid piece move D2 to B2. Try again...
RED MOVE: Enter new coordinates of piece at D2: C2
RED 5 move from D2 to C2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |   5       |
D | 8   B B 1 |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C3
Error: Invalid piece or not moveable at C3. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D4
Error: Invalid piece or not moveable at D4. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C2
RED MOVE: Enter new coordinates of piece at C2: B3
Error: Invalid piece move C2 to B3. Try again...
RED MOVE: Enter new coordinates of piece at C2: B2
RED 5 at C2 captured by BLUE 3 at B2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |           |
D | 8   B B 1 |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D5
RED MOVE: Enter new coordinates of piece at D5: C5
RED 1 move from D5 to C5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |         1 |
D | 8   B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C5
RED MOVE: Enter new coordinates of piece at C5: C4
RED 1 move from C5 to C4.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |       1   |
D | 8   B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C4
RED MOVE: Enter new coordinates of piece at C4: C3
RED 1 move from C4 to C3.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |     1     |
D | 8   B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C3
RED MOVE: Enter new coordinates of piece at C3: C2
RED 1 move from C3 to C2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |   1       |
D | 8   B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C2
RED MOVE: Enter new coordinates of piece at C2: B2
RED 1 at C2 capture BLUE 3 at B2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X 1 X X X |
C |           |
D | 8   B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B2
RED MOVE: Enter new coordinates of piece at B2: A2
RED 1 at B2 capture BLUE 8 at A2.
    1 2 3 4 5
  +-----------+
A | X 1 X X X |
B | X   X X X |
C |           |
D | 8   B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): A2
RED MOVE: Enter new coordinates of piece at A2: A3
RED 1 at A2 blown up by BLUE B at A3.
    1 2 3 4 5
  +-----------+
A | X   X X X |
B | X   X X X |
C |           |
D | 8   B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D1
RED MOVE: Enter new coordinates of piece at D1: C1
RED 8 move from D1 to C1.
    1 2 3 4 5
  +-----------+
A | X   X X X |
B | X   X X X |
C | 8         |
D |     B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): c1
Error: Invalid row location c. Try again...
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C1
RED MOVE: Enter new coordinates of piece at C1: C2
RED 8 move from C1 to C2.
    1 2 3 4 5
  +-----------+
A | X   X X X |
B | X   X X X |
C |   8       |
D |     B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C2
RED MOVE: Enter new coordinates of piece at C2: B2
RED 8 move from C2 to B2.
```

```
    1 2 3 4 5
  +-----------+
A | X   X X X |
B | X 8 X X X |
C |           |
D |     B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B2
RED MOVE: Enter new coordinates of piece at B2: A2
RED 8 move from B2 to A2.
    1 2 3 4 5
  +-----------+
A | X 8 X X X |
B | X   X X X |
C |           |
D |     B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): A2
RED MOVE: Enter new coordinates of piece at A2: A3
RED 8 at A2 defuse BLUE B at A3.
    1 2 3 4 5
  +-----------+
A | X   8 X X |
B | X   X X X |
C |           |
D |     B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): A3
RED MOVE: Enter new coordinates of piece at A3: A4
RED 8 at A3 capture BLUE S at A4.
    1 2 3 4 5
  +-----------+
A | X     8 X |
B | X   X X X |
C |           |
D |     B B   |
E | 3 B 6 S F |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): A4
RED MOVE: Enter new coordinates of piece at A4: A5
RED 8 at A4 capture BLUE F at A5. Congratulations!
    1 2 3 4 5
  +-----------+
A | 4     8 F |
B | B   B 3 2 |
C |           |
D |     B B   |
E | 3 B 6 S F |
  +-----------+
$ ./a.out
              +----------------------------------------------+
              |       Computer Science and Engineering       |
              |        CSCE 1030 - Computer Science I        |
              |   Student Name     EUID    euid@my.unt.edu   |
              +----------------------------------------------+

           W e l c o m e   t o   1 0 3 0   S t r a t e g o

------------------------------------------------------------------------
This program will set up a 5x5 game board for a 1030 version of the game
of Stratego. One player will compete against the computer, each assigned
10 total pieces consisting of the following:
  1 FLAG (F)
  3 BOMB (B)
  1 MARSHAL (1) or GENERAL (2)
  3 COLONEL (3), MAJOR (4), CAPTAIN (5), LIEUTENANT (6), or SERGEANT (7)
  1 MINER (8)
  1 SPY (S)

GENERAL RULES:
--------------
For the most part, the game will follow the standard Stratego rules, al-
though there are some exceptions.
1. Both players (BLUE and RED) will have all of their 10 game pieces as-
   signed randomly with the only requirement being that the FLAG must be
   placed in the back row. RED will start the game first.
2. Higher ranked pieces can capture lower ranked pieces in the following
   order: 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> S, meaning that 1 (the
   MARSHAL) can remove 2 (the GENERAL) and so forth. The MINER (8) piece
   may strike a BOMB and remove it to occupy the now unoccupied space. A
   SPY (S), although the lowest ranked piece, may remove the MARSHAL (1)
   or the GENERAL (2). When pieces have equal rank, then both pieces are
   removed.
3. The FLAG and BOMBs are not moveable while all of the other pieces may
   move one square at a time forward, backward, or sideward, but not di-
   agonally to an open square.
4. A player must either move or strike on his/her turn.
5. The game ends when a player strikes his/her opponent's flag.
------------------------------------------------------------------------

Initializing game board...
Assigning BLUE pieces to board...
Assigning RED  pieces to board...
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |           |
D | B 8 7 7 S |
E | 2 7 F B B |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D5
RED MOVE: Enter new coordinates of piece at D5: C5
RED S move from D5 to C5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |         S |
D | B 8 7 7   |
E | 2 7 F B B |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C5
RED MOVE: Enter new coordinates of piece at C5: B5
RED S at C5 captured by BLUE 6 at B5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |           |
D | B 8 7 7   |
E | 2 7 F B B |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D4
RED MOVE: Enter new coordinates of piece at D4: C4
RED 7 move from D4 to C4.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |       7   |
D | B 8 7     |
E | 2 7 F B B |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C4
RED MOVE: Enter new coordinates of piece at C4: B4
RED 7 at C4 captured by BLUE 5 at B4.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |           |
D | B 8 7     |
E | 2 7 F B B |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): QQ
RED player forfeits game. Please play again soon!
    1 2 3 4 5
  +-----------+
A | S F 2 B B |
B | 4 8 B 5 6 |
C |           |
D | B 8 7     |
E | 2 7 F B B |
  +-----------+
$ ./a.out
              +----------------------------------------------+
              |       Computer Science and Engineering       |
              |        CSCE 1030 - Computer Science I        |
              |   Student Name     EUID    euid@my.unt.edu   |
              +----------------------------------------------+
              W e l c o m e   t o   1 0 3 0   S t r a t e g o

------------------------------------------------------------------------
This program will set up a 5x5 game board for a 1030 version of the game
of Stratego. One player will compete against the computer, each assigned
10 total pieces consisting of the following:
  1 FLAG (F)
  3 BOMB (B)
  1 MARSHAL (1) or GENERAL (2)
  3 COLONEL (3), MAJOR (4), CAPTAIN (5), LIEUTENANT (6), or SERGEANT (7)
  1 MINER (8)
  1 SPY (S)

GENERAL RULES:
--------------
For the most part, the game will follow the standard Stratego rules, al-
though there are some exceptions.
1. Both players (BLUE and RED) will have all of their 10 game pieces as-
   signed randomly with the only requirement being that the FLAG must be
   placed in the back row. RED will start the game first.
2. Higher ranked pieces can capture lower ranked pieces in the following
   order: 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> S, meaning that 1 (the
   MARSHAL) can remove 2 (the GENERAL) and so forth. The MINER (8) piece
   may strike a BOMB and remove it to occupy the now unoccupied space. A
   SPY (S), although the lowest ranked piece, may remove the MARSHAL (1)
   or the GENERAL (2). When pieces have equal rank, then both pieces are
   removed.
3. The FLAG and BOMBs are not moveable while all of the other pieces may
   move one square at a time forward, backward, or sideward, but not di-
   agonally to an open square.
4. A player must either move or strike on his/her turn.
5. The game ends when a player strikes his/her opponent's flag.
------------------------------------------------------------------------

Initializing game board...
Assigning BLUE pieces to board...
Assigning RED  pieces to board...
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |           |
D | B 7 6 B 2 |
E | F 3 B 8 S |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D5
RED MOVE: Enter new coordinates of piece at D5: C5
RED 2 move from D5 to C5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X X |
C |         2 |
D | B 7 6 B   |
E | F 3 B 8 S |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C5
RED MOVE: Enter new coordinates of piece at C5: B5
RED 2 at C5 capture BLUE 8 at B5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X 2 |
C |           |
D | B 7 6 B   |
E | F 3 B 8 S |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B5
RED MOVE: Enter new coordinates of piece at B5: A5
RED 2 at B5 blown up by BLUE B at A5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X   |
C |           |
D | B 7 6 B   |
E | F 3 B 8 S |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): E5
RED MOVE: Enter new coordinates of piece at E5: C5
Error: Invalid piece move E5 to C5. Try again...
RED MOVE: Enter new coordinates of piece at E5: D5
RED S move from E5 to D5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X   |
C |           |
D | B 7 6 B S |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D5
RED MOVE: Enter new coordinates of piece at D5: C5
RED S move from D5 to C5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X   |
C |         S |
D | B 7 6 B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C5
RED MOVE: Enter new coordinates of piece at C5: B5
RED S move from C5 to B5.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X X S |
C |           |
D | B 7 6 B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B5
RED MOVE: Enter new coordinates of piece at B5: B4
RED S at B5 capture BLUE 2 at B4.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X S   |
C |           |
D | B 7 6 B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B4
RED MOVE: Enter new coordinates of piece at B4: A4
RED S at B4 blown up by BLUE B at A4.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X     |
C |           |
D | B 7 6 B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D2
RED MOVE: Enter new coordinates of piece at D2: C2
RED 7 move from D2 to C2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X     |
C |   7       |
D | B   6 B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C2
RED MOVE: Enter new coordinates of piece at C2: B2
RED 7 at C2 blown up by BLUE B at B2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X     |
C |           |
D | B   6 B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D3
RED MOVE: Enter new coordinates of piece at D3: C3
RED 6 move from D3 to C3.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X X     |
C |     6     |
D | B     B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C3
RED MOVE: Enter new coordinates of piece at C3: B3
RED 6 at C3 capture BLUE S at B3.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X 6     |
C |           |
D | B     B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B3
RED MOVE: Enter new coordinates of piece at B3: A3
RED 6 at B3 captured by BLUE 4 at A3.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X       |
C |           |
D | B     B   |
E | F 3 B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): E2
RED MOVE: Enter new coordinates of piece at E2: D2
RED 3 move from E2 to D2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X       |
C |           |
D | B 3   B   |
E | F   B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): D2
RED MOVE: Enter new coordinates of piece at D2: C2
RED 3 move from D2 to C2.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X       |
C |   3       |
D | B     B   |
E | F   B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C2
RED MOVE: Enter new coordinates of piece at C2: C3
RED 3 move from C2 to C3.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X       |
C |     3     |
D | B     B   |
E | F   B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): C3
RED MOVE: Enter new coordinates of piece at C3: B3
RED 3 move from C3 to B3.
    1 2 3 4 5
  +-----------+
A | X X X X X |
B | X X 3     |
C |           |
D | B     B   |
E | F   B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): B3
RED MOVE: Enter new coordinates of piece at B3: A3
RED 3 at B3 capture BLUE 4 at A3.
    1 2 3 4 5
  +-----------+
A | X X 3 X X |
B | X X       |
C |           |
D | B     B   |
E | F   B 8   |
  +-----------+
RED MOVE: Enter current coordinates of piece (e.g., D2, or QQ to quit): A3
RED MOVE: Enter new coordinates of piece at A3: A2
RED 3 at A3 capture BLUE F at A2. Congratulations!
    1 2 3 4 5
  +-----------+
A | 6 F 3 B B |
B | 3 B       |
C |           |
D | B     B   |
E | F   B 8   |
  +-----------+
```

SUBMISSION:

Your program will be graded based largely upon whether it works correctly on the CSE machines, so you should make sure your program compiles and runs on the CSE machines.

Your program will also be graded based upon your program style. This means that you should use comments (as directed), meaningful variable names, and a consistent indentation style as recommended in the textbook and in class.

We will be using an electronic homework submission on Blackboard to make sure that all students hand their programming projects on time. You will submit both (1) the program source code files and (2) the algorithm design document to the **Homework 6** dropbox on Blackboard by the due date and time.

CSCE 1030 – Homework 6

Due: 11:59 PM on Wednesday, December 6, 2017

Note that this project must be done individually. Program submissions will be checked using a code plagiarism tool against other solutions, including solutions found on the Internet, so please ensure that all work submitted is your own.

Note that the dates on your electronic submission will be used to verify that you met the due date and time above. All homework up to 24 hours late will receive a 50% grade penalty. Later submissions will receive zero credit, so hand in your best effort on the due date.

As a safety precaution, do not edit your program (using `vim` or `nano`) after you have submitted your program where you might accidentally re-save the program, causing the timestamp on your file to be later than the due date. If you want to look (or work on it) after submitting, make a copy of your submission and work off of that copy. Should there be any issues with your submission, this timestamp on your code on the CSE machines will be used to validate when the program was completed.

General Guidelines (for ALL of your programming assignments):

- Your program's output should initially display the department and course number, your name, your EUID, and your e-mail address.
- Use meaningful variable names.
- Use appropriate indentation.
- Use comments, including a program header. Example program header:

```
/*
 ============================================================================
 Name        : homework2.cpp
 Author      : Mark A. Thompson
 Version     :
 Copyright   : 2015
 Description : The program performs simple arithmetic operations based on in-
               put from the user.
 ============================================================================
 */
```

- Add a header to each function. Example function header:

```
/*
============================================================================
 Function    : deposit
 Parameters  : a double representing account balance and a double represent-
               ing the deposit amount
 Return      : a double representing account balance after the deposit
 Description : This function computes the account balance after a deposit.
============================================================================
 */
```