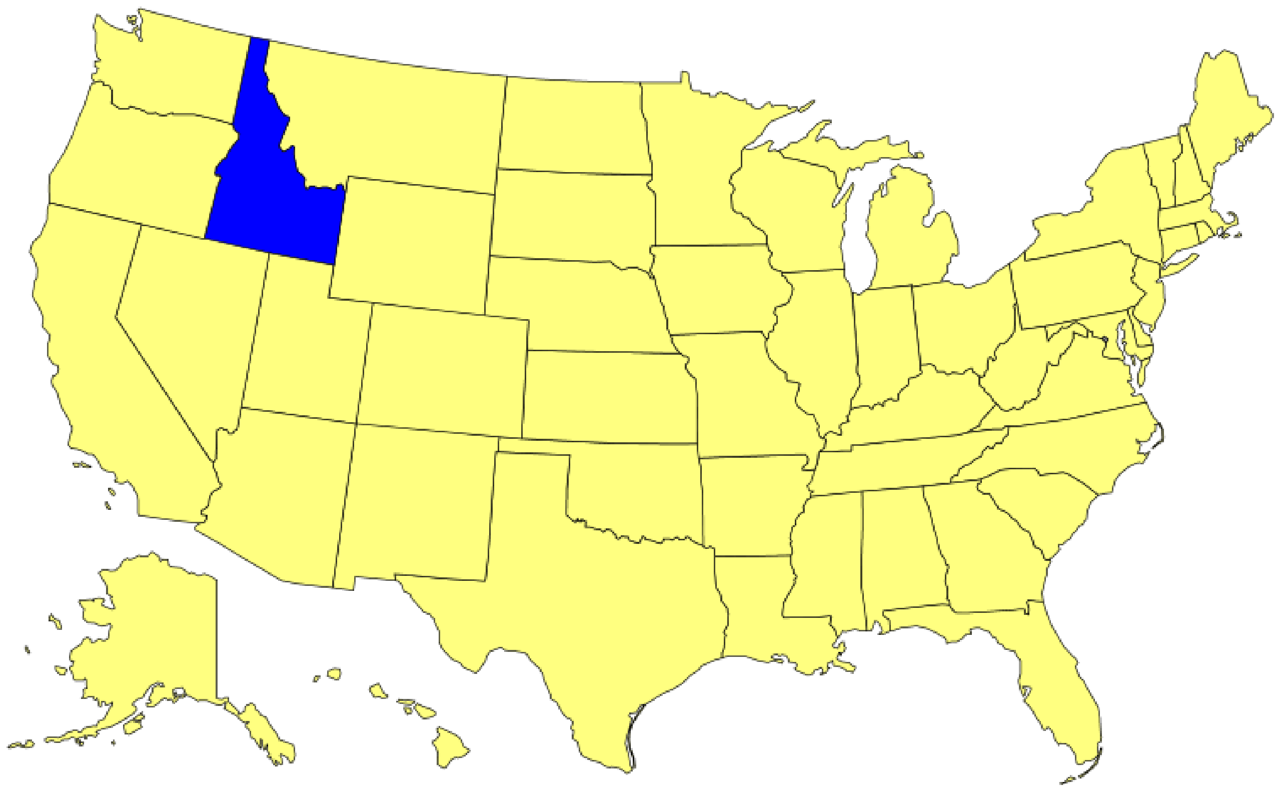
EGR 125 Due date: _________________
Introduction to Engineering Methods (C++)
File: N125P4D

Programming Assignment #4: Class StateGame

Software is commonly used to create educational games. In this project you will create a ***Name The State Game*** that could be used to test the user's knowledge of geography and other information about the 50 United States.



Play the *Name The State Game*!

Clue #1: It is the 14th largest state.

Clue #2: The state capital ends in the letter e

Clue #3: It is in the following time zone(s): PST/MST

Clue #4: It is bordered by the following states: MT,NV,OR,UT,WA,WY

Arkansas? X

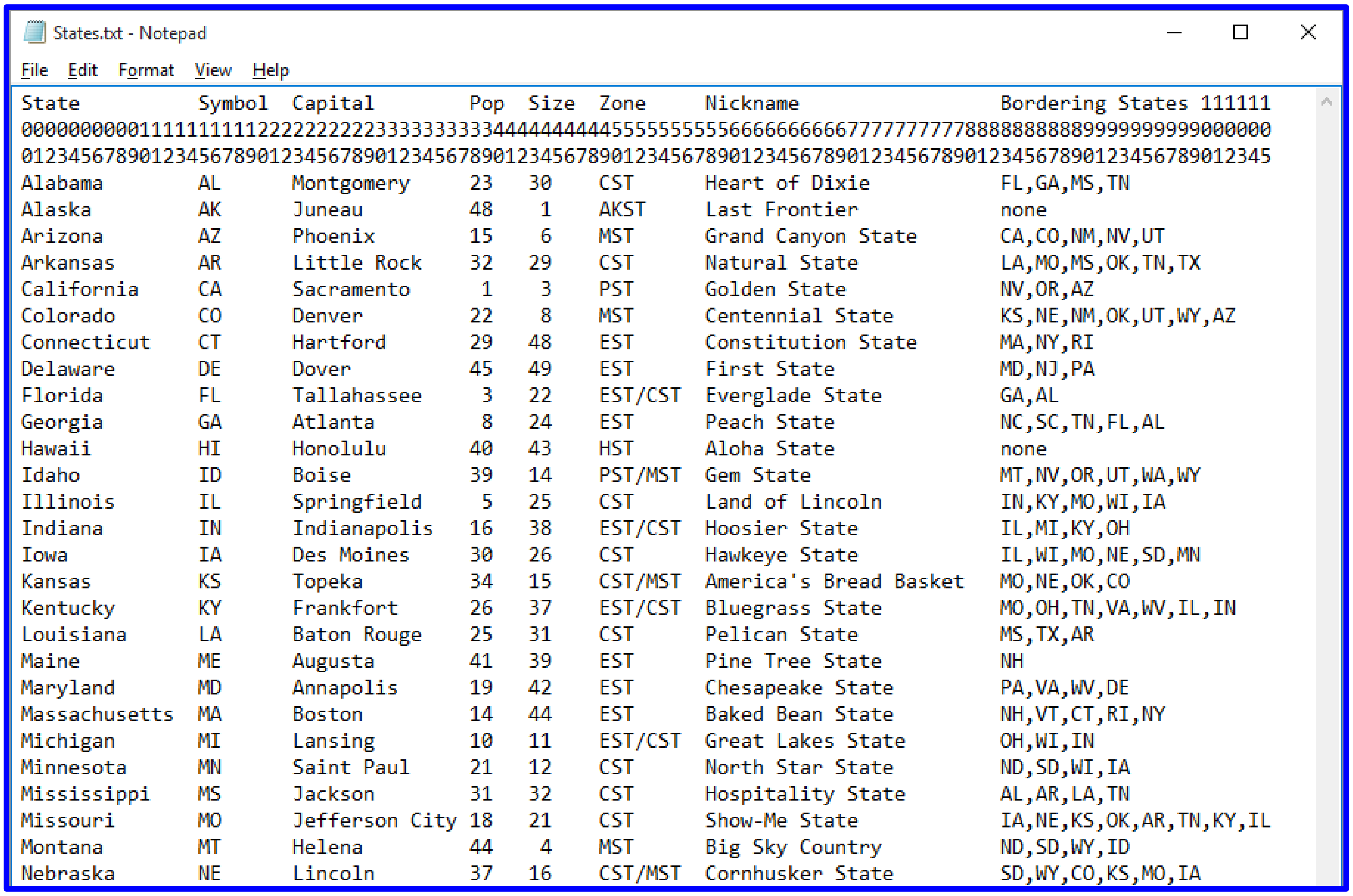
Louisiana? X

South Dakota? X

Idaho? ✓ Correct!

State Information

A data file named ***states.txt*** is available on the course website with information about each state.
Part of the data file is shown below:



```
State          Symbol  Capital        Pop  Size  Zone     Nickname                 Bordering States 111111
0000000000111111111122222222223333333333444444444455555555556666666666777777777788888888889999999999000000
0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345
Alabama        AL      Montgomery     23   30    CST      Heart of Dixie           FL,GA,MS,TN
Alaska         AK      Juneau         48    1    AKST     Last Frontier            none
Arizona        AZ      Phoenix        15    6    MST      Grand Canyon State       CA,CO,NM,NV,UT
Arkansas       AR      Little Rock    32   29    CST      Natural State            LA,MO,MS,OK,TN,TX
California     CA      Sacramento      1    3    PST      Golden State             NV,OR,AZ
Colorado       CO      Denver         22    8    MST      Centennial State         KS,NE,NM,OK,UT,WY,AZ
Connecticut    CT      Hartford       29   48    EST      Constitution State       MA,NY,RI
Delaware       DE      Dover          45   49    EST      First State              MD,NJ,PA
Florida        FL      Tallahassee     3   22    EST/CST  Everglade State          GA,AL
Georgia        GA      Atlanta         8   24    EST      Peach State              NC,SC,TN,FL,AL
Hawaii         HI      Honolulu       40   43    HST      Aloha State              none
Idaho          ID      Boise          39   14    PST/MST  Gem State                MT,NV,OR,UT,WA,WY
Illinois       IL      Springfield     5   25    CST      Land of Lincoln          IN,KY,MO,WI,IA
Indiana        IN      Indianapolis   16   38    EST/CST  Hoosier State            IL,MI,KY,OH
Iowa           IA      Des Moines     30   26    CST      Hawkeye State            IL,WI,MO,NE,SD,MN
Kansas         KS      Topeka         34   15    CST/MST  America's Bread Basket   MO,NE,OK,CO
Kentucky       KY      Frankfort      26   37    EST/CST  Bluegrass State          MO,OH,TN,VA,WV,IL,IN
Louisiana      LA      Baton Rouge    25   31    CST      Pelican State            MS,TX,AR
Maine          ME      Augusta        41   39    EST      Pine Tree State          NH
Maryland       MD      Annapolis      19   42    EST      Chesapeake State         PA,VA,WV,DE
Massachusetts  MA      Boston         14   44    EST      Baked Bean State         NH,VT,CT,RI,NY
Michigan       MI      Lansing        10   11    EST/CST  Great Lakes State        OH,WI,IN
Minnesota      MN      Saint Paul     21   12    CST      North Star State         ND,SD,WI,IA
Mississippi    MS      Jackson        31   32    CST      Hospitality State        AL,AR,LA,TN
Missouri       MO      Jefferson City 18   21    CST      Show-Me State            IA,NE,KS,OK,AR,TN,KY,IL
Montana        MT      Helena         44    4    MST      Big Sky Country          ND,SD,WY,ID
Nebraska       NE      Lincoln        37   16    CST/MST  Cornhusker State         SD,WY,CO,KS,MO,IA
```

Note that rows 1-3 provide headings and column numbers. The other rows are organized as follows:

• State	- columns 0 - 14
• State Symbol	- columns 15 - 16
• …	
• Bordering States	- columns 83 - (end of line)

Clues – The user will be provided a series of 9 clues, in random order, from the following list. The blanks below will contain the appropriate information obtained from the data file. The program must use these 9 clues.

- ***The first letter of the state is ___.***
- ***The last letter of the state is ___.***
- ***The first letter of the state capital is ___.***
- ***The last letter of the state capital is ___.***
- ***The state's rank in population, where 1 is the largest, is ___.***
- ***The state's rank in size (or area), where 1 is the largest, is ___.***
- ***The state is in the following time zone(s): ___***
- ***The state's nickname is ___;***
- ***The state is bordered by the following states: ___.***

<u>Rules for playing the *Name The State Game* (single user option):</u>

1) The program will read the information for one ***<u>random</u>*** state from the data file.
2) The program will provide an introductory message and ask the user if they want a summary of the rules.
3) The program will present the user with the 9 clues above in ***<u>random</u>*** order.
4) The program will keep score. A perfect score is 100. If the user correctly guesses the state after the first clue, their score is 100.
5) After each clue the user will be given an option of guessing the state or asking for another clue. The clues should be numbered (from 1 to 9) as they are presented so that the user knows how many clues they have used.
6) Each time the user makes an incorrect guess, 5 points will be deducted from their score.
7) Each time the user asks for another clue, 5 points will be deducted from their score.
8) Display the score after every clue or guess.
9) The game will end if their score drops to 0 without correctly guessing the state.
10) After the last clue the user should be given the option of guessing the state or giving up. If they give up, display the name of the state.
11) When the user guesses a state, they should be able to enter the full state name or the state symbol using any mixture of uppercase or lowercase letters.

<u>Sample game output:</u> (The output doesn't need to look exactly like this.)

```
Welcome to the Name The State Game!
Would you like a summary of the rules (Y or N)?  N
Let's get started.  A state has been randomly selected.  Here is your first clue:

Clue #1:  The state's rank in size (or area), where 1 is the largest, is 14
Current Score:  100
Would you like to guess the state (1) or do you want another clue (2)? 2

Clue #2:  The last letter of the state capital is e
Current Score:  95
Would you like to guess the state (1) or do you want another clue (2)?   1
Enter your guess (name or symbol):  LA
Incorrect.  Current Score:  90
Would you like to guess the state (1) or do you want another clue (2)?   2

Clue #3:  The state is in the following time zone(s): PST/MST
Current Score:  85
Would you like to guess the state (1) or do you want another clue (2)?  2

Clue #4:  The state is bordered by the following states: MT,NV,OR,UT,WA,WY
Current Score:  80
Would you like to guess the state (1) or do you want another clue (2)?  1
Enter your guess (name or symbol):  IDAHO

Correct!  Congratulations!  Your final score is 80
```

Program Requirements:

1) Creating a class: The program should define **class StateGame** (or pick your own name) according to the following class diagram.
   - Use the data members indicated and member functions indicated. Additional data members and member functions can be added if you wish.
   - All data members must be private. All member functions must be public.
   - Once the state information has been accessed using ***ReadLine( ),*** do not open or read from the data file again. All state information should then be accessed using accessor functions (***Get…( )***), not by reading the file again. Do not define State, Symbol, etc., in main. Access them from the class.
   - Do not display values (using cout, for example) from any member function. Access the data using accessor functions and display any desired values within main.
   - Use a member functions to get the Score and to update the Score. Do not define Score in main.
   - Include a constructor function which initializes Score to 100.
   - Use separate header and implementation files for the class (StateGame.h and StateGame.cpp)

Class StateGame	
Member Data: - State: string - Symbol: string - Capital: string - Population: string (or int) - Size: string (or int) - TimeZone: string - NickName: string - BorderingStates: string - Score: int	Comments: All data members must be private (-)
Member Functions: + StateGame( ); + ReadLine( ); void + GetState( ): string + GetSymbol( ): string + etc (other member functions)	Comments: Constructor – Create new game and initialize Score to 100 Open file, read random line, extract State, Symbol, etc. Use Get…( ) ***accessor*** functions to access all member data Add other member functions, including functions to get the Score and to update the Score.

2) Main program: The main program should make one game object and make use of member functions to complete the game.
   - Be sure to implement the ***Rules for playing the Name The State Game*** previously listed.
   - Study the ***Sample Game Output*** shown previously. Your output does not to look exactly like this example, but the example illustrates many of the requirements.
   - Be sure to follow the guidelines listed under ***Creating a class*** above.

Testing your Program

Since the program selects random states and uses a random question order, exact test conditions can't be specified. Instead, run your program and include the results for the following:

- Case 1: Run the program for a state where you made no incorrect guesses but required 3 or more clues before getting the correct answer. The user should enter the entire state name with a mixture of uppercase and lowercase letters. The user should also request a summary of the rules.

- Case 2: Run the program for a state where you made no incorrect guesses but required 3 or more clues before getting the correct answer. The user should enter the state symbol.
- Case 3: Run the program for a state where you guessed a state after every clue, but still needed 3 or more clues before getting the correct answer.
- Case 4: Run the program for a state where you used all 9 clues and gave up.
- Case 5: Run the program for a state where you got the correct answer on the first try (this might take a while!)
- Case 6: Run the program for a state where you made so many guesses that the score dropped to 0 and the program stopped (but you didn't give up).

Programming Hints:

1. **Random Numbers** – Our text introduces functions to produce random numbers. They may not be covered in this courses unless they are needed for a project. They are needed in this project, so here is a summary:
   - **rand( )** – This function produces a pseudo-random number between 0 and 32767 (compiler dependent). It is pseudo-random because it produces the same sequence of number each time it is executed on a given computer. This occurs because the random number is generated using a seed which has a default value of 1.
   - **srand(time(0))** - To make **rand( )** produce a random number, we need to first execute **srand(time(0))** which generates a new seed at the current time.
   - Example:

```
# include <ctime>  // needed for the time( ) function
# include <cstdlib> // needed for the rand( ) and srand( ) functions
using namespace std;
int main()
{
      srand(time(0));         // include this or else rand( ) produces the same number each run
      int Number1 = rand( )%10;         // produces a random number from 0 to 9
      int Number2 = rand( )%30;         // produces a random number from 0 to 29
      …
}
```

   - So if you want to read a random number of lines from a file with 50 lines (1 per state), then ….

2. **Shuffling a list** – We can see how to produce one random number, but how about shuffling a list? Recall that one of the benefits of using the ***vector class*** is that there are many built-in functions, including one named ***random_shuffle( ).***
Since we have 9 questions that we would like to present randomly, suppose the numbers 1-9 are placed in a vector named QuestionOrder. So the original vector contents are:

1	2	3	4	5	6	7	8	9

Executing the command below will result in a shuffled vector.
random_shuffle(QuestionOrder.begin(), QuestionOrder.end())
One ***possible*** result for the shuffled vector is shown below.

7	3	1	9	2	6	4	5	8

<u>Extra Credit Suggestions</u>: (for a maximum of 10 additional points on the program grade)

1. The basic program described above is a 1-player game. Also give the user the option of a ***<u>2-player game</u>***.
   - The 2 players will share the same keyboard.
   - Ask the users to enter their first names and use their names in addressing them.
   - Randomly select which player starts first.
   - The first player gets the 1st clue and then must enter a state.
   - If second player gets the 2nd clue and then must enter a state.
   - The first player gets the 3rd clue and then must enter a state.
   - This pattern continues until one of the players correctly guesses the state.
   - Both players can see all clues displayed and all states that have been guessed.
2. Keep track of the game results in an output file. Each time a game is finished, append the state and score to the end of the file. Also tell the user how many games they have played (by reading the file) and their average score. Example:

   ```
   Virginia 95
   Connecticut 75
   Hawaii 100
   Florida 85
   ```

   You have played 4 games. Your average score is 88.75

   Show the output of the program after playing 5 games (all with non-zero scores) and also show the contents of the output data file.
3. Use your imagination!