PIC 10A Section 2 - Homework #6 (due Monday, February 14, by 11:59 pm)

You should upload each .cpp (and .h file) separately and submit them to CCLE before the due date/time! Your work will otherwise not be considered for grading. Do not submit a zipped up folder or any other type of file besides .h and .cpp.

Be sure you upload files with the precise name that you used in your editor environment otherwise there may be linker and other errors when your homeworks are compiled on a different machine. Also be sure your code compiles and runs on Visual Studio 2022.

CARD RANGE GAME

This homework is aimed at getting you to use random numbers, vectors, and to read and write functions!

At the end of this homework, you should submit 3 files: **Range.h** that declares and documents functions you are to write, **Range.cpp** that defines those functions, and **Game.cpp** that has your main routine.

Do not write a class. At this point in the course, classes have not been covered. You only need to write functions.

This is a one-player game that vaguely-ish resembles BlackJack. There are 3 identical rounds where a player attempts for the value of all cards in their hand to be within a fixed range. This range is randomly set each round.

Each card of a standard deck of 52 cards has a value associated with it.

- The cards numbered 2-10 are worth that many points; for example a 4 of any suit has a value of 4.
- All Jacks (J), Queens (Q), and Kings (K) have value 11.
- An Ace (A) is has value 12 if it is Spades and otherwise it has value 1.

In each round, a number from 17 to 22 inclusive is randomly chosen as a lower bound for the value of the user's hand. The player aims for the total value of all cards in their hand to *reach or exceed the given target value*, while at the same time ensuring the *total*

value of their hand does not exceed 22!

At the start of every round, there is a full deck of 52 cards, represented by a **std::vector<Card>** (see the **Card.h** and **Card.cpp** files). Cards are dealt by a card being randomly selected from the deck and removed from it.

During a round, the target lower bound should be displayed and the user dealt a card if they have chosen so (except for the first card that will always be dealt). The total value of cards in the user's hand should be displayed along with the cards in their hand. Then,

- if the value of their hand is within the desired range, they earn 1 point and are told so.
- if the value of their hand exceeds 22, they earn -1 points and are told so.
- if the value of their hand is less than the target value, the player may choose whether or not to be dealt another card by choosing 'y' for "yes" or 'n' for "no".

If they choose to be dealt no more cards, they do not earn or lose any points, they are told they earned 0 points, and the next round, if there is one, commences.

A round should end if the user is within range, without them having to choose not to be dealt more cards!

When the game is over, the user's total score should be displayed.

The specifics:

Download the **Card.h** and **Card.cpp** files and include them in your workspace. Do not modify these files in any way. You **must use the Card** class in this assignent; some of the other functions included may also be of use to you. You will not submit either **Card.h** or **Card.cpp** in the end as they will be automatically included when the homework is tested.

Write a **Game.cpp** that implements the game described above. The desired input and a sample outputs are provided.

You **must define and use** the following functions in nontrivial ways in your code (this is where **Range.h** and **Range.cpp** that you must write come into play):

- **cardValue**, which can be passed a **Card** as input and returns the card's value as an **int**;
- **dealCard**, which accepts an **std::vector<Card>** representing the deck of undealt cards, chooses a card at random and removes it from the deck, and returns that selected **Card**;
- **showHand**, which accepts an **std::vector<Card>** representing the player's hand and an **int** for the total value of cards in their hand, and which displays their hand value and cards in hand to the console;
- **exceeds**, which accepts two arguments, an **int** representing the max possible hand value allowed and an **int** representing the user's hand value, returning **true** only if the user's hand value has exceeded the maximum possible hand value allowed; and
- **inRange**, which accepts three arguments, an **int** representing the target lower bound for a hand value; an **int** representing the maximum possible hand value allowed; and an **int** representing the user's hand value, returning **true** only if the user's hand value is within the target range.

The desired format is below:

Cards 2-10 are worth their numeric value.
J, Q, K have a value of 11.
An A has a value of 12 as a spade and 1 for other suits.
Over 3 rounds, you will try to reach or exceed a target value, without going over 22!
If you go over, you get -1 points; if you are within range, you get +1 points; otherwise you get 0 points.

FOR EACH OF 3 ROUNDS [
  UNTIL THE USER WINS OR LOSES OR CHOOSES TO ACCEPT NO MORE CARDS [
    Target lower bound: [VALUE CHOSEN]
    Hand value is: [CORRECTLY COMPUTED HAND VALUE]
    Hand is: [LIST OF CARDS]
    Deal more? y/n: [USER ENTERS 'y' or 'n']
  ]
  You got [SOME NUMBER] [POINT/POINTS]
]
Your total score is [CORRECT TOTAL] [POINT/POINTS].

The [] are not part of the display; they are just used to group the logic together. There might be nested loops, etc. Only one of "point" or "points" should display, depending on the number of points earned/displayed.

```
Cards 2-10 are worth their numeric value.
J, Q, K have a value of 11.
An A has a value of 12 as a spade and 1 for other suits.
Over 3 rounds, you will try to reach or exceed a target value, without going over 22!
If you go over, you get -1 points; if you are within range, you get +1 points; otherwise you get 0 points.

Target lower bound: 22
Hand value is: 8
Hand is: 8[Hearts]
Deal more? y/n: y
Target lower bound: 22
Hand value is: 19
Hand is: 8[Hearts] Q[Hearts]
Deal more? y/n: y
Target lower bound: 22
Hand value is: 20
Hand is: 8[Hearts] Q[Hearts] A[Hearts]
Deal more? y/n: y
Target lower bound: 22
Hand value is: 21
Hand is: 8[Hearts] Q[Hearts] A[Hearts] A[Diamonds]
Deal more? y/n: y
Target lower bound: 22
Hand value is: 28
Hand is: 8[Hearts] Q[Hearts] A[Hearts] A[Diamonds] 7[Spades]
You got -1 points.

Target lower bound: 17
Hand value is: 5
Hand is: 5[Spades]
Deal more? y/n: y
Target lower bound: 17
Hand value is: 15
Hand is: 5[Spades] 10[Spades]
Deal more? y/n: y
Target lower bound: 17
Hand value is: 17
Hand is: 5[Spades] 10[Spades] 2[Diamonds]
You got 1 point.

Target lower bound: 22
Hand value is: 1
Hand is: A[Diamonds]
Deal more? y/n: y
Target lower bound: 22
Hand value is: 13
Hand is: A[Diamonds] A[Spades]
Deal more? y/n: y
Target lower bound: 22
Hand value is: 17
Hand is: A[Diamonds] A[Spades] 4[Spades]
Deal more? y/n: y
Target lower bound: 22
Hand value is: 20
Hand is: A[Diamonds] A[Spades] 4[Spades] 3[Diamonds]
Deal more? y/n: n
You got 0 points.


Your total score is 0 points.
```