CIS 278 (CS1) Programming Methods: C++

Assignment 2: Loops

Skip to Main Content

Learning Objectives

After the successful completion of this learning unit, you will be able to:

- Implement syntactically correct while, do-while, and for loops
- Critically analyze a problem to determine which type of loop will best meet the objectives of effective programming practice to solve the problem.

Assignment 2.1 [45 points]

This assignment is a review of loops. Do not use anything more advanced than a loop, such as programmer-defined functions or arrays or classes.

If you know blackjack, you may be tempted to change the specifications below to make the game more "blackjack-like". Please don't. Be sure to follow these specs exactly and don't try to improve on them. You will be penalized if the specs are not met exactly.

In the card game named 'blackjack' players get two cards to start with, and then they are asked whether or not they want more cards. Players can continue to take as many cards as they like. Their goal is to get as close as possible to a total of 21 without going over. Face cards have a value of 10.

Write a command line game that plays a simple version of blackjack. The program should generate a random number between 1 and 10 each time the player gets a card. Each of the values (1 through 10) must be equally likely. (In other words, this won't be like real black jack where getting a 10 is more likely than getting some other value, because in real black jack all face cards count as 10.) It should keep a running total of the player's cards, and ask the player whether or not it should deal another card. If the player hits 21 exactly, the program should print "Congratulations!" and then ask if the player wants to play again. If the player exceeds 21, the program should print "Bust" and then ask if the player wants to play again. Sample output for the game is written below. Your program should produce the same output.

If you'd like a little refresher on random number generation, see **lesson 7.3.**

```
> First cards: 3, 2

> Total: 5

> Do you want another card? (y/n): y

> Card: 6

> Total: 11

> Do you want another card? (y/n): y

> Card: 7

> Total: 18

> Do you want another card? (y/n): n

> Would you like to play again? (y/n): y

>

> First cards: 10, 2

> Total: 12
> Do you want another card? (y/n): y

> Card: 6

> Total: 18

> Do you want another card? (y/n): y

> Card: 7

> Total: 25

> Bust.

> Would you like to play again? (y/n): n
```

Suggestion

Be sure to use iterative development. Start with a small amount of functionality, and then grow it gradually. This way you can compile and run your program after each statement that you write.

You might start by just generating a single card. The program execution might look like this:

```
> First card: 3
```

Then generate two cards

```
> First cards: 3, 2
```

Next add a variable to store the total, and a statement to show its value:

```
> First cards: 3, 2

> Total: 5
```

Next read in a user response and print out the value that was entered

```
> First cards: 3, 2

> Total: 5

> Do you want another card? (y/n): y

> You entered: y
```

Next you might add a loop, without yet adding the blackjack logic

```
> First cards: 3, 2

> Total: 5

> Do you want another card? (y/n): y

> Do you want another card? (y/n): y

> Do you want another card? (y/n): n
```

Now move the display of the total to the loop

```
> First cards: 3, 2

> Total: 5

> Do you want another card? (y/n): y

> Total: 5
> Do you want another card? (y/n): y

> Total: 5

> Do you want another card? (y/n): n
```

Your next steps might be something like this:

- Generate a new card in each loop and display the value
- Update the total in each loop.
- Check to see if the user busts in each loop
- Wrap the game in loop that handles the Play-Again functionality

Submit Your Work

Name your source code file according to the assignment number (a1_1.cpp, a4_2.cpp, etc.). Execute the program and copy/paste the output that is produced by your program into the bottom of the source code file, making it into a comment. Use the Assignment Submission link to submit the source file. When you submit your assignment there will be a text field in which you can add a note to me (called a "comment", but don't confuse it with a C++ comment). In this "comments" section of the submission page let me know whether the program works as required.

Keep in mind that if your code does not compile you will receive a 0.