Lab 9
Boolean logic

29 October 2024

More functions this week, this time with a focus on writing and using Boolean functions and expressions.

First you'll be writing a function with three parameters, all integers, that determines whether *any two* of the given numbers could be added to get the third.

1. Before you go further in the design process, think carefully about how many test cases you'll need at a minimum, and write out (or type in) some good ones.
2. Follow the process from last week's lab and class to set up `.h`, `.u`, and `.cpp` files that have (for now) just the stuff for this function, and a readme for the directory. (There will be more functions later in the lab.)
3. If you haven't already, type your test cases into the `.u` file.
4. Write the function in the `.cpp` file, test it, and debug it.

I'll be circulating around the lab to answer questions. If you're stuck on some part of the preview, ask me about that (and while you're waiting for me to get to you, look at the next sections). If you're not stuck but haven't finished the preview, work on that now. If you're done with the preview, continue on to the next section.

More boolean stuff

Even more than last week, this one is a bunch of disjointed tasks that are loosely grouped as "involving booleans". Don't get overly wrapped up in the silly cover story for each one as long as the function performs correctly.[1]

[1] Several of these are adapted from problems at codingbat.com.

For each function, write good test cases. Since we've been talking more about this in class, I'm stepping up a little on the test case requirements: to be good test cases with good coverage, you should make sure to cover things like edge cases (boundary conditions) where relevant, and all meaningfully different outcomes and logic paths.

- Write a function that computes a slot machine jackpot based on the three given digits: if all three are different, the payoff is $10. If all are the same, the payoff is $100 for 7s or $50 for 1s, and $25 for other digits. Otherwise (if there's one pair and a mismatch), the payoff is $0.
- Write a function that decides whether the children in a daycare will be allowed to play outside based on the given temperature and whether it's currently summertime. Most of the year, they'll be permitted to play if the temperature is between 65 and 85 degrees (inclusive); when it's not summertime they've brought coats and can play outside if temperatures are as low as 50.

  Note that in this case, the second parameter is itself a `bool` value!
- Write a function that chooses whether to answer a phone based on three given conditions: whether it's before 9am, whether it's your mom calling, and whether you're asleep. In the early morning, you only answer if it's your mom calling, but other times you'll answer no matter who's calling. If you're asleep, though, you can't answer, regardless of anything else.
- Write a function that evaluates words for appropriateness for a certain kind of themed puzzle: a given word is appropriate if it starts or ends with the given letter, but not both.

Handing in and rubric

Hand in as `lab9`. Due 4pm next Monday.

RUBRIC

1 Attendance at lab with preview done or question written down
Part 1 (add two to get third)
1 Comment, header, compiles
1 Test cases
1 Correct definition
Rest of lab
1½ slot machine
- **½** TCs
- **½** ... full coverage
- **½** ... & definition correct

1½ playing outside TCs, coverage, def'n
1½ answer the phone TCs, coverage, def'n
1½ puzzle words TCs, coverage, def'n

AI policy and frequent submission

(no substantive change from the Lab 3 version of this policy)

Some use of generative AI is fine, but you a) should not paste this assignment or type it verbatim into the AI prompt, and b) should not be asking the AI for the whole program all at once. (Just like you can ask for help from a human, but should not have them write the whole program for you!) If you get help from an AI chat OR from a person, you should note that in a program comment near whatever you got from them.

Relatedly, I expect that you'll run the `handin` program relatively often, and I *require* that you do so at least 2–3 times over the course of working on the lab. As a rule of thumb, hand it in after completing each 1–2 points on the rubric. Submissions that jump straight to a final, (near-)correct version with no intervening submissions along the way *will receive little or no credit* for that part.

This is a new policy that I'm experimenting with; let me know if you have any feedback.

This document was written and prepared without the use of generative AI.