In class lab for January 11, 2021

The *checksum* of an integer is the sum of the digits in the integer remainder 10 (or mod 10)[1]. For example, lets says we had the integer 46183. The checksum would be 4 + 6 + 1 + 8 + 3 = 22, then 22 mod 10 = 2. So the checksum of 46183 is 2. The reason checksums are useful is for error checking network communication. For example, if we were to transmit the integer and its checksum over the internet as the pair (46183, 2) then the receiver could verify that the transmission did not contain an error by verifying the checksum.

For example, lets say Alice wants to send Bob the integer 46183, she would send the pair (46183, 2). If there was an error in the transmission and Bob received (46983, 2), here the 1 got changed to a 9 because of electromagnetic interference, Bob could detect that there was a problem by computing the checksum of 46983 and noticing that it was not 2, because 4 + 6 + 9 + 8 + 3 = 30 and 30 mod 10 is 0, not 2. He would then notify Alice that there was an error and ask her to resend. This trick also works if there is an error in the checksum digit, but not if there were an error in *both* the ineteger and the checksum digit (or if the were multiple errors in the integer).

1. **Part 1:** In your personal CS220 repo write a **`main.c`** that has two functions **`main`** and **`checksum`**. The function **`checksum`** should take a non-negative integer **`n`** as a parameter and return the checksum of **`n`**. Write a simple **`main`** function that tests your function on 46183 and verifies that it computes 2.
2. **Part 2:** Modify the **`main`** function so that you prompt for and read an integer from the user (on the keyboard), call your **`checksum`** function, and then prints the checksum of the integer read. Make sure to use nice prompts, output and comment your functions.
3. **Part 3:** Break you **`main.c`** file up into two files, one named **`main.c`** that only contains the main function, and the other **`checksum.c`** that contains the checksum function. Use separate compilation to compile and the run your program.
4. **Part 4:** Push your two files up to GitHub and verify by logging on to GitHub in your brower and check that both files are there.
5. **Part 5:** If time allows. Modify your checksum function so that it appends the checksum digit as the least significant digit to the original integer. For example, **`checksum(46183)`** would return 461832 instead of just 2.
6. **Part 6:** If time allows. Modify your main function so that it repeatedly prompts for, reads an integer from the keyboard, and prints the checksum until they enter a -1.

1 This is one definition of a checksum. There are many different kinds of checksums.