HW1

Tuesday, January 26, 2021 6:56 PM

Q1) The goals of this course is to make the most fun and rewarding course experience I'll have (students have). This is acheived by technical objectives that include programming skills and for job referrals and interviews. The objective of the programming skills is to find and be prepared for interview opportunities that demand the following skills for both hardware and software teams of tech companies.

Some skills to learn are:

- OOP (using C++ &/or C).
- some basic algorithms
- Data structures
- Scripting using Python for automation & verification tasks.
- Several software management techniques (including versioning, testing, etc.).

For job interviews, the HW/labs are designed to help prepare for interviews. Also professor will to help prepare for interviews. Also professor will introduce alot of tips.

There is also a goal for those interested in research and/or startup. There will be loads of extra credit designed to help for this purpose.

Q2) I am expected to:

- stay safe
- follow academic honesty policies of USC.
- Attend every Zoom lecture and discussion sessions.
- Take notes during lecture/discussion sessions or while watching the videos later.
- Participate in discussions by asking questions & providing answers/arguments.
- Aim to top the class
- Review any posted material in advance to corresponding lecture
- Do all mandatory parts and then doing extra credit parts.
- Have video on
- [illegible] notes & announcements on BB.

- Have [illegible]
- Check notes & announcements on BB regularly.
- Keep course material confidential
- Inform instructors ASAP of any course-related issues

Q4) First take in 10 digit # as long. Then run a for loop that loops through all 10 digits. Have a variable that keeps track of the sum of all the representations of the digits in binary form. Also, every time for loop iterates, the sum variable should be added by the 10s representation of that binary digit.

i.e.

$$\begin{array}{cccccccccc} 2^9 & 2^8 & 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ \underline{1} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} \\ \uparrow & & & & & & & & & \end{array}$$

add to sum variable by $1 \times 2^9 = 2^9$ for the 1st iteration.

After for loop is done, print sum.

Note that this assumes that the user enters in valid 10-digit binary # (i.e. no # like 999 as it does not exist in binary).

it does not exist in binary).

Q6) This is in short using Fibonacci #s.

$1 \quad 1 \quad 2 \quad 3 \quad 5 \quad 8 \quad 13 \quad 21 \quad \ldots$

this is the "0th" stair so to speak in that there is 1 way to climb 0 stairs by staying there in place. It sounds confusing so it is ok to just say that the 1st stair starts at the # circled in black.

To put this in code, we can use recursion. The reasoning is because to reach the nth stair, one has to be at the n-1 th or n-2 th stair. Then it goes that you get the total # of ways by adding # ways of how to get n-1 th & n-2 th stair.

So you can make a function that defines the Fibonacci sequence such that @ $n=0 \Rightarrow 0$, $n=1 \Rightarrow 1$ and for $n>2$, function returns sum of functions$(n-1)$ and function$(n-2)$.

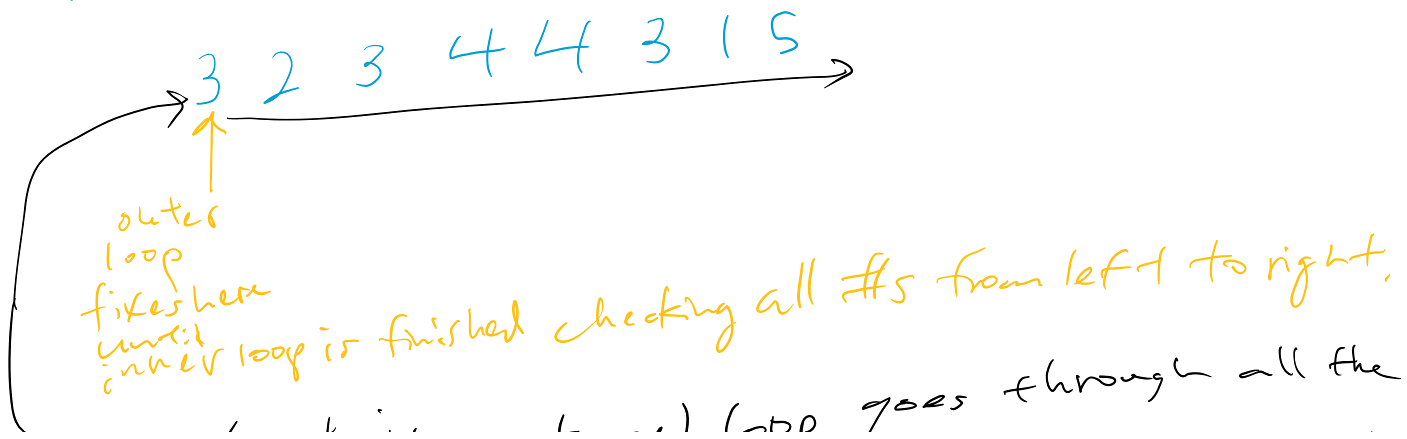
Then create another function that returns the

Then create another function that returns the nth term of the fibonacci sequence.
Then in main function (assuming C/C++), output the function (not the fibonacci one directly) result.

Q3) Assume user is smart enough to supply 200 valid heights (i.e. no negative heights, all heights are NOT the same, no NaN or ∞ heights, etc.).
Then, once you get the input heights via input iostream or cin/cout method, put all 200 heights into an array/vector of length 200. Next, use either a sort function or a sorting algorithm to sort the 200 heights in ascending order (i.e. first element is lowest height, last element is tallest height).
Then after array is sorted, take the first element and stick it into a variable and then [illegible] into the array at the [illegible]

element and stick it
append this value into the array at the
end of the array. Then, delete the first
element of the array.
We can also choose to make another array
of 200 and then fill values from 2-200 to
positions 1-199 of the new array. The last
element of the new array will be filled w/ the
value we put in the variable as aforementioned.
This new array is the desired result of
the 200 hights and then you are done.

Q5) First, take in list of #'s as a single
string. Then make 3 nested for loops. The
outer 2 for loops check each substring by
fixing one loop to one digit.


3 2 3 4 4 3 1 5
outer loop fixes here until inner loop is finished checking all #s from left to right.
goes through all the

inner loop is

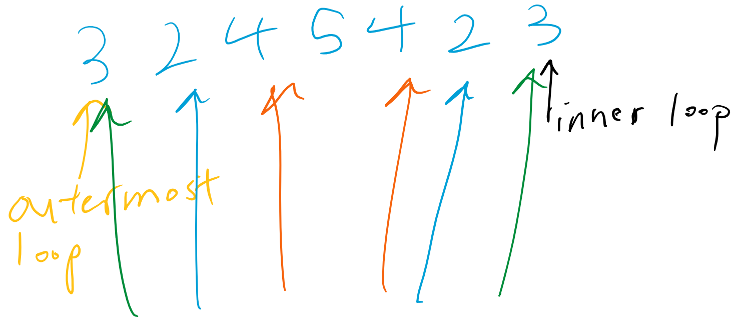
inner (not innermost one) loop goes through all the #s to see if there is a palindromic sub list.

Innermost loop is used to check/validate whether the substring using the indexes by the outer 2 loops is a valid palindrome. This can be done by using a flag and after validation, if flag still stays that the substring is a valid palindrome, you keep track of starting index (outermost for loop) & the length of the substring.

After all for loops are done, you can simply get a substring of the original list of #s and then print it to the console.

Innermost for loop checks by doing following:

Say the substring is:



Innermost loop checks each color pair and

Inner most loop checks each color pair and if it matches, it keeps flag at a high such that the program knows it is a valid palindrome (e.g. flag = 1).