This is largely copied from Andrew’s 2014 guide: Quick Guide on Getting Unstuck

A major frustration you might encounter in intro programming is when you stare at a homework problem and have no idea where to start. Or you write some code and it doesn't do what you want, but now what? You work at it for a while, but next thing you know, you've been stuck for hours on the same problem and have little to show for it.

So here's a checklist of things you can do when you're stuck. Experienced programmers do these things almost naturally **(because of how much practice they've had being stuck),** and so ***while they get stuck just as much as you or I*****,** they always know what to do next.

0. Do I understand what the problem is asking?
    a. If not, which part of the problem is confusing me?
        i. Identify the exact sentences/phrases/words/etc.
    b. Check the given examples (from class or office hours). Do they make sense to me?
    c. Can I come up with my own examples? A good indicator that you understand the question is that you can come up with some non trivial examples of how the code works.
    d. How can I solve the problem on paper (and with pictures), BEFORE writing the code?
    e. Can I describe the behavior of the function in one sentence?
    f. If you’re writing a function, what’s the domain and range?
1. What concepts should I use here?
    a. Do I understand the concepts? **Can I explain the concept in English to one of my friends so that they get it in 5 minutes?**
        i. If not, go back and relearn the *specific* concepts that are unclear (through discussion, lab, lecture, etc.) Don't read the entire book in order to solve one problem.
    b. How do I apply the concept to the given problem?
2. Write your code **and test it.**
    a. Use given tests, **BUT ALSO WRITE YOUR OWN TESTS**
        i. Saying "my function works because the doctests pass" is a lot like saying "this airplane will fly because it has wings."
    b. If your code breaks, ask yourself:
        i. Does it throw an error? Which line number? Is it a....
            1. Syntax error? If so, find the syntax bug and fix it.
            2. Logic error? Is it something weird that you don't understand? (E.g. cannot add integer and tuple)
        ii. Why did it do that? Why didn't it do what I expected?

Trace through the code by hand with an example (sample values) you came up with in step 0. **Add calls to print in order to figure out how your function is handling the arguments.**

1. Am I missing a trick?
    - a. Oftentimes you've never seen this type of problem before. This is expected on homework (and this is why homework can take a long time) because if you see it on the homework, then you will be familiar with it on the exam and when you program for fun and profit.
    - b. The key here is just to learn the trick however you need to.
        - i. Stare at it yourself
        - ii. Stare at it with others (peers in the class)
        - iii. Ask on canvas / in office hours what the approach is.
        - iv. Stare at it with the TAs/lab Assistants
    - **c. Once you figure it out, remember the trick so that you can use it next time.**
2. At any point you identify what you're stuck on, you can begin to resolve it.
    - a. Use the tips above. Try things out on the interpreter. Review the lecture/discussion/labs/etc. Do whatever helps you get a better understanding of the problem.
    - b. Once you have something specific that you're stuck on, you can ask other people in the class.
        - **i. Don't be afraid to ask. Everyone gets stuck and feels stupid sometimes. However, you get to choose how you react to it.**
        - **ii. At the same time, it really helps to work with people who are on about the same level in the course.**
    - c. Look at Canvas. Ask questions if yours hasn't come up yet. Be that awesome person who helps answer questions.
    - d. You can ask the TA if all else fails. We are here to help you learn!