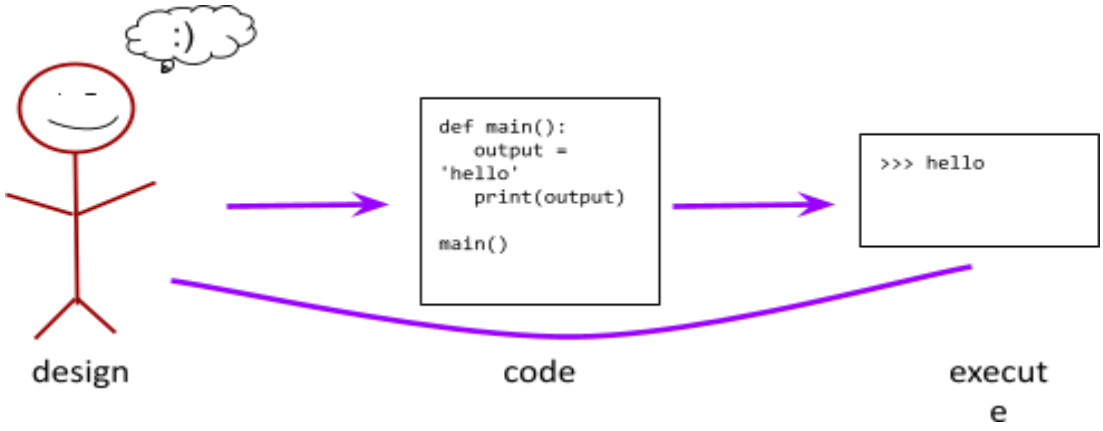
DS2000
Fall 2021
Handout: How to Approach a Programming Homework

Python code is executed with an *interpreter*. It parses the source code (Python statements) and turns it into machine code (0s and 1s) in an on-demand kind of way. It reads a Python statement, executes it on the computer, and moves on to the next one.

We suggest you install Anaconda and use Spyder to do your Python programming. If you like and use another editor, that's totally fine, but we only "officially" support Spyder so you may be on your own if you choose another option.

How To Code
Follow these steps, and repeat regularly. Write only *a teeny tiny little bit* of code before you run it. Make sure you can run all the way through without errors, fix them if you need to and go back and write a teeny tiny bit more code.



Getting Started
Read the assignment, the whole thing. Keep a pen handy and jot down anything that's confusing or might need a little extra attention.

1. Design
Once you've read the assignment, start to put together your **pseudocode**. Pseudocode doesn't need to follow a particular format, but it does need to be somewhere between English and Python. Bullet points are usually helpful; make them specific and clear without actually writing code.

For example, if the assignment asks you to get two numbers from the user and report their sum, your pseudocode might look something like this:

- Ask user to enter two numbers
- Save user input in two variables
- Add two numbers together and store result in a third variable
- Print third variable

Finally, write a few test cases. You know what your program needs to do, and ***before you start coding*** make sure you know what to expect. That way, when you go to execute your code later on, you'll understand right away whether it's working correctly.

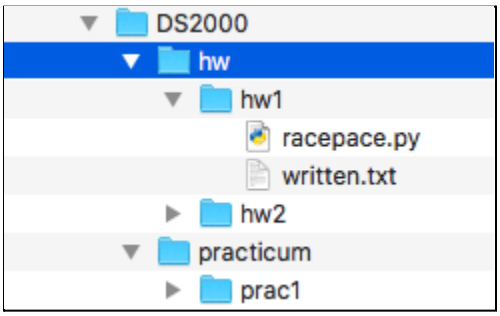
In the above example, we might say:

- Inputs: 9, 5. Output: 14
- Inputs: 10, -1. Output: 9
- Inputs: -3, -4. Output: -7
- Inputs: 0, 0. Output: 0

You might feel like you can skip this step, especially early in the course, but don't! Get into the habit of designing your solution now, while it's straightforward, and it'll be that much easier when your programs become more sophisticated.

2. Write Code

We suggest that you have a `DS2000` directory on your computer, and within that you should have a directory for practicum, one for homework, and maybe a third one for sample code or getting some practice on your own. The structure of your directories would look like this on a Mac:



When you're ready to begin coding, open up Spyder. Choose `File > New File` from the menu. Type the Python main function outline before you do anything else. The rest of your code will go within the main function *definition* (at the top), before the *call* to the main function (at the bottom).

```
'''
    DS2000
    Fall 2021
    HW1 - variables and operations
'''

def main():
    # your code goes here!

main()
```

In the snippet above, we start with a block comment enclosed in three single-quotes at either end. This is at the very top of the file, with a description of what the program is.

You should have comments throughout the rest of your program, either block-quotes like we have above, or using one crosshatch (`#`) per line. Be generous but not overwhelming with your comments. Rule of thumb:

> **I should be able to understand your program *just* from reading your comments.**

Start coding up the pseudocode you already have. You don't need to go exactly in order, but choose a few bullet points and put them right there into Python.

Your pseudocode probably didn't specify variable names, so you'll need to do that here. Make sure your variable names are:

- concise
- descriptive
- begin with a lowercase letter

Write just a few lines, with comments, and then move on to step three. Seriously, like 3 lines of code. Maybe 4. Then run it. If you make a habit of running and testing your code after just a few changes, it's much MUCH easier to debug!

3. Execute Your Code

Save your program in the correct directory with a .py extension. Now you're ready to run it through the Python interpreter. The interpreter runs with `Function+F5` (or click the "play" button).

Did you get some errors? **DON'T PANIC!!!** It's no big deal, and not even a sign that you did something wrong. There is no programmer, no matter how smart and experienced, who writes a whole program without errors. We all get them, we all deal with them. Read the errors and try to figure out what went wrong.

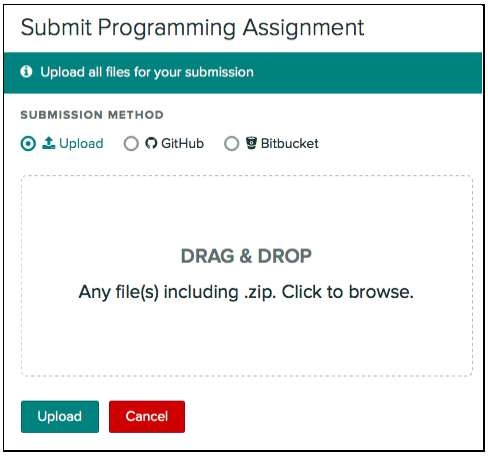
Reading and understanding errors takes some time, too. We get better at it over time. Be patient.

Did your program do what you expected? Great! Now go back to step one, because you've only written 3 or 4 lines of code and you'll need to finish up the program. :)

Submitting Your Homework

We'll be using Gradescope for homework assignments (and quizzes). Go to https://www.gradescope.com/courses/279891 and use entry-code **2R5BVP** to enroll (if necessary; this course is linked on Canvas and Gradescope already, so if you sign up for Gradescope with your Northeastern email — as you should — things ought to work automagically).

Click on the assignment you want to submit and you'll see a window like this, click inside the window to browse the files on your computer:



Select the file(s) to upload (use ctrl-click or command-click to select more than one file): The "autograder" will run first. This just checks to make sure the upload was successful so don't get too excited if it checks out! (Although if it *fails* then you really need to resubmit!) You can continue to make new submissions right up until the deadline.

Academic Integrity
While students are encouraged to discuss course materials, no plagiarism/copying is allowed on homework. In particular,

- You may not copy anyone else's code under any circumstances. This includes online sources.
- You may not permit any other student to see any part of your program.
- You may not permit yourself to see any part of another student's program.
- You may not post a public question to Piazza that contains any part of your code.

You may consult online resources as part of your course work, but you may not copy code from online sources. If you get an idea of how to solve a problem from an online source, include a short citation in a code block at the top of your .py file:

```
'''
DS2000
HW2
MyName
Today's Date

Consulted Stack Overflow about swapping variables
https://stackoverflow.com/questions/16749669/how-to-swap-more-t
han-two-variables-using-temporary-variables
'''
```

You do not need to include a similar notation if you consulted with a classmate; we expect that. Just don't share code.