6.00.1x Week 2 FAQ

- **When using a "guess and check" algorithm, how big or small does the epsilon and step have to be to make the program efficient as well as accurate enough?**

  The smaller is epsilon, the more accurate the approximation.

  There isn't exactly an "optimal" value for step. The rule is that the smaller the step, the more accurate the answer. But the smaller the step, the longer it will take to run. So the optimal is a balance between these two facts. So it's basically a question of how long can you wait for the answer.

- **In the Approximate Solutions video at 3:49 the result is 2.999700000001906. Why is that? This also comes up in Exercise 2 where the answer isn't 5.**

  By checking when the solution is within .01 of the cube trying each guess to the nearest .0001, the answer should be 2.9997 rather than 3. Also kiwi has provided a link to the Python docs that have a nice tutorial on floating point numbers as to why it isn't exactly 2.9997.

- **How does the guess my number exercise work?**

  The grader will "think" of a number. **You don't want to use `input` to get that number**.

  You want to use the binary search technique to determine the correct number. You get the input from the grader and if the grader says the number you have calculated is "l" for low, then you want to restrict the possible numbers to the upper half of the remaining numbers. It is says "h" for high, then you want to restrict the possible numbers to the lower half of the numbers. Each guess will cut the possible numbers in half. So the grader will give you a set number of "l"s "h"s and finally a "c". *Each guess* and your *final calculation* of the guess should come up with what the grader is looking for.

- **What is the purpose of parentheses after a function name?**

  The parentheses tells Python to execute the function. If you have the code `print(function_name)` then it will only print the memory address where the function is stored at. If you have `print( function_name() )` with the parentheses then the function will execute and whatever is returned from the function is printed out.

- **What does "scope" mean?**

  Scope stands for where a variable is visible. Variables defined in a function are said to be in that function's scope. The best way to see how scope works is to use the Python Tutor here. On the "Write code in" drop-down box, select Python 3.6. Paste some code and press the [Visualize Execution] button. Then press the [Forward] button to see how the different scopes work.

- **What does global and local scope mean, and can a function change something defined in the global scope?**

  If a variable is defined outside of any function then it is in the global scope. Variables defined inside a function are in that function's local scope.

  There are two ways a function can change something that is in the global scope. One is to put `global` in front of the variable name, and then define `global variable_name` inside the function where you want to change that variable. (The other way to change something is if the variable is "mutable", for example if the variable is a list or a dictionary. Lists and dictionaries are covered in the Week 3 Lectures.)

- **What causes an UnboundLocalError,?**
  This is caused when a reference is made to a local variable that hasn't been defined yet.

- **What does it mean that functions are "first-class objects" in Python?**
  Functions can be passed into functions and returned from functions, just like a variable.

- **What is the difference between `.find` and `.index`?**
  If the value you are looking for isn't found in the string, the two methods work differently. `.index` returns an error, but `.find` returns -1.

- **What is recursion, and when is it best to use recursion?**
  Recursion is when a function calls itself within it's code. The function would call itself an infinite number of times, except that you define a "base case" where you return a value *instead of* doing the recursive call. There are some problems that can't be solved without recursion. Recursion is best used when you don't know exactly how many times a function will need to be executed in order to solve the problem. When there's a definite number of steps that need to be executed then iteration is the better approach, because it is more efficient due to how the Python language was designed.

- **How do you import a module, and what is the purpose of doing it?**
  There are two ways to import a module: The code to use is `import module_name` or `from module_name import *`. (If you only want a selected function to be imported you would code `from module_name import function_name`). A module contains a set of functions that are commonly used together. Usually it's written and tested by someone else. Because modules are used by many people, it is proven to be correct and very efficient code.

- **I've been struggling with PSET 2 for hours, and just can't seem to get it. What should I do?**
  See the following Walkthroughs by Tom Ballatore:

  Problem Set 2 Walkthrough-Problem 1
  Problem Set 2 Walkthrough-Problem 2
  Problem Set 2 Walkthrough-Problem 3