**Homework 1 - Debugging/Time Complexity/Reading from a file. Due 8:00 AM Wed 5/16/2018**

**Part 1: Debugging (33 pts)**

**Q1: When running a debugger in an IDE, what is the difference between step over and step into?**

Step into - When a method is about to be invoked and you would like to debug the code within the method as well. The debugger will enter the method and continue step by step. If the line does not have a function it will behave just like step over.

Step over - When a method is about to be invoked, but you care about the overall process of your code, and not this specific function, so the debugger executes the method as one step, and not the individual steps of the method. The result will still be returned, but each line of the function will not be debugged.

\*\*\* In relation to Question 3, when stepping over a function with a bug, the debugger may not show you the bug within.

**Q2: Describe how to run the debugger in your favorite IDE.**

Starting a debugger session begins with placing a breakpoint at the statement of interest. This is done by clicking the left gutter of that line, in which a red dot will appear as well as the entire line changing to pink. By right-clicking the dot, many properties become available for customization. Selecting the green bug in the top right will start the debugger session, and within the stepping toolbar there are options to step over, step into, force step into, and step out.

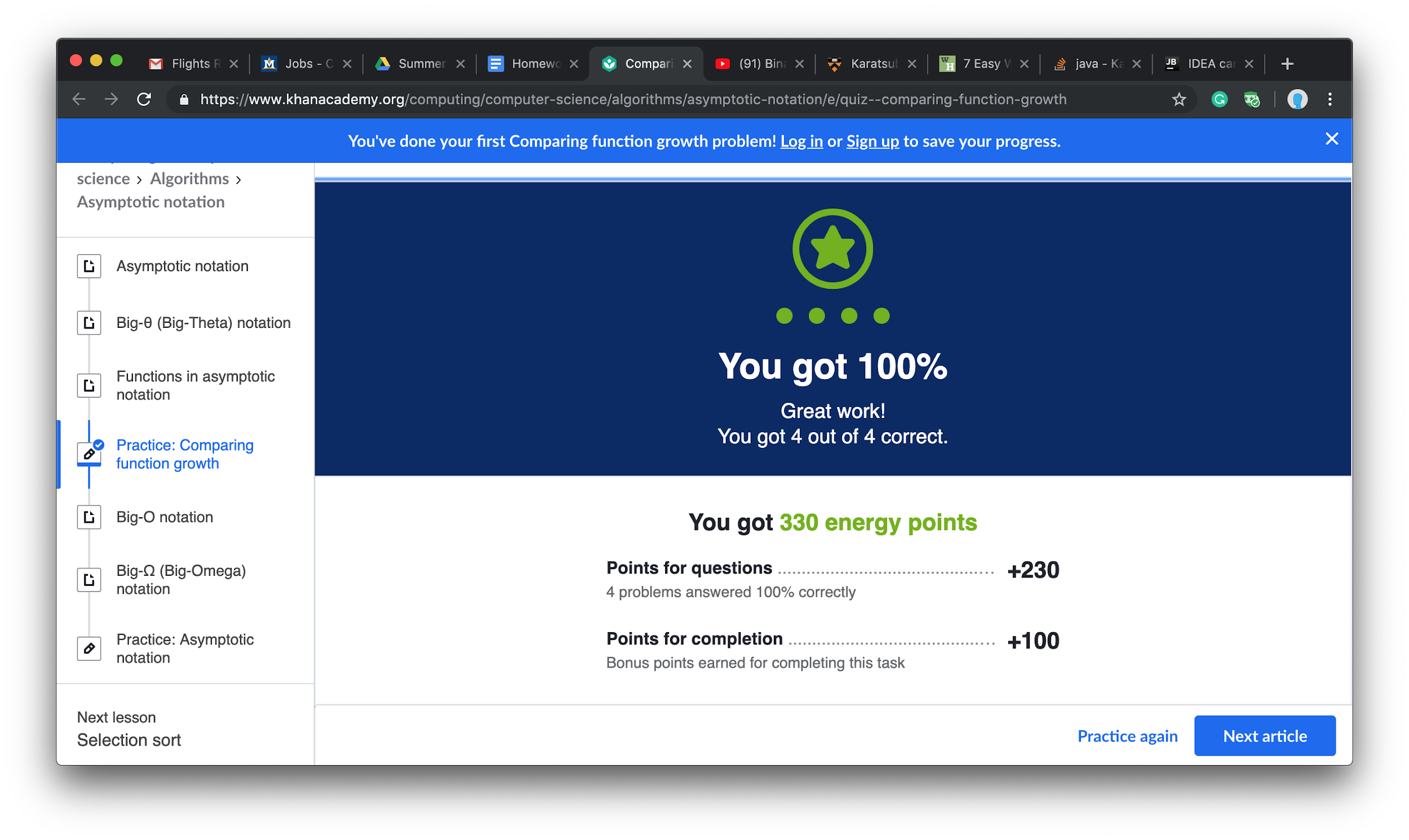
**Q3: Describe when you would use a debugger and when the debugger may not show you a bug in your code.**

It is practical to use the step into tool when attempting to find an error within a method, or simply to understand the different steps of the function. By using the step into tool, each element of the method, and it’s corresponding parts, are shown independently. The step over tool should be used to run the entire method until it returns as one step. This can be efficiently used when attempting to observe a more broad view of the program's functions and abilities while ignoring one possibly inoperative portion.

The debugger may not show bugs if the code has not been previously compiled before debugging begins, or when the execution flow is changed using the debugger.

**Part 2: Time Complexity (33 pts)**

**Q1: Complete the following quiz. Take the quiz until you get 100% on all of the questions. Please provide a screenshot showing that you received a 100% on the quiz.**

****

**Part 3: Reading in from a file (34 pts)**

Write a program that reads the contents of an input.txt file and print the contents of each word in its own line. You MAY NOT USE ABSOLUTE FILE PATHS. Absolute file paths will cause your code to FAIL to run on any machine other than yours. Only use relative file paths. Submit your program as either a git repo or zip up the entire project and provide the zipped file.