Exam 2: Study guide

Exam 2 for CE 311K is a one-hour closed-book exam held on 19th November 2019 in JGB 2.218 during class time. You may bring two sheet of 8.5 x 11 inch of your own handwritten notes to the examination (One of those must be your Exam 1 cheat sheet). You may use your calculator. You may not access the internet during the exam. The exam questions will be determined such that they satisfy a subset of the objectives listed here.

Exam 1 will cover:

- Errors, functions, data structures, Taylor series and Newton Raphson
- Lecture handouts # 3 and #5
- Homeworks 02 through 04 (Errors, functions, Taylor series and Newton Raphson)
- Labs 02 to 05 (Errors, functions, Taylor series and Newton Raphson)

To perform successfully on Exam I, you should be able to:

1. Determine data types and outputs during casting operations. Please note only Python standard data types (`str`, `int`, `float`, `bool`) will be covered. Numpy data types are not included (for e.g., `np.float16` and others)
2. Evaluate relative and absolute error(s) for a given program and develop a suitable termination criteria (`break`) or (`return` in case of a function).
3. Define function with arguments (including default) and multiple return types for a given problem and call (use) the functions in a Python code.
4. Rewrite a given program using functions to make re-use of code as much as possible.
5. Identify and fix errors in passing function arguments and return types.
6. Evaluate the output of a given function(s).
7. Evaluate the value of different variables within and outside the function (scoping)
8. Recursions are **not** part of the exam.
9. Develop Python code that use, index, manipulate and search (`in` and `not in`) lists.
10. Iterating through a list using indexing and `in` operations.
11. List comprehensions are *not* part of the exam.
12. Deduce the value of a variable after trying to modify a list item, a tuple and a dictionary using an index or a key.

13. Other use cases of tuples and dictionary are **not** part of the exam.

14. Develop Taylor series approximation for non-polynomial functions. Write a Python code to solve for the Taylor approximation with relative errors.

15. Develop Newton-Raphson code to find the root of a function. Compute the tolerance error at each iteration.

You won't be required to write lengthy code (more than 30 lines). I will not penalise for obvious typos and syntax errors in your code (for e.g., missing `:` at the end of function definitions), unless that is what is tested.