PHYS 162 Fall 2019 Midterm 2

Rules, Guidelines, and Suggestions:

- **Do not begin until you are instructed to do so.**
- **Please indicate somewhere in your source code what operating system you're using (Windows, Mac, Linux), what text editor you use (Xcode, notepad++, Emacs, etc.), and by what means you compile your code (from within some fancy software, from the cygwin terminal, from a Linux terminal, from the Mac Terminal, etc.). Providing this information helps me to figure out if some error is system-dependent.**
- No communicating with anyone except for me.
- The only internet resource you can use is the Google Drive folder for this course. You can also refer to *your own* homework files.
- For your C++ source files, make sure they compile!
- Compress all files to be submitted into a single zip file before submitting on Canvas.
- NO CELL PHONES. They cannot be out or visible. You CANNOT use them to keep time and you CANNOT use them as a calculator.
- The test is out of 100 points. Point values for each problem are indicated.
- Be sure to attempt every problem. If you get stuck on a problem, skip it and move on to the next one. Work all the problems you consider easy first. An easy problem and a difficult problem of the same length have the same point value.
- Partial credit will be given assuming you SHOW YOUR WORK and USE COMMENTS to tell me what you're thinking.
- Don't hesitate to raise your hand or come to the front of the room to ask me a question if something is unclear. I can't guarantee I'll be able to give you an answer, but it never hurts to ask.
- You will have until 2:50PM to complete the exam.
- You may keep this copy of the exam when you are done.

Python

1. (30 points) Use Python's numerical solving tools (within the scipy module) to:

   (a) (12 points) Find all positive solutions for x given the equation

   $$1.5\,x\cos(0.6x) = 2.2 + e^{x/5}.$$

   (b) (18 points) Find all solutions for x and y to the given system of equations

   $$\begin{cases} 0.59x^4 + y^2/2.4 = 1.5(x+0.8) \\ e^{-0.3xy} = \sin(y^2) \end{cases} .$$

   Note: I will be looking for plots that you used to roughly locate the solutions; these plots needn't be fancy. Also, I recommend one Python *script file for each part.*

2. (35 points) Suppose some biology researcher went out into the field and measured the lengths of 1,000 individual snakes of a certain species. Those measured lengths, in cm, are contained in the file "snakeData.dat", which can be found on Google Drive in the Exams folder. Write a Python script to import that data and complete the following tasks:

   (a) (4 points) Compute directly from the data the average snake length and the sample's standard deviation of snake lengths.

   (b) (6 points) Make a histogram of the snake length data.

   (c) (5 points) Add points at the center and top of each rectangle plotted in the histogram.

   (d) (9 points) It can reasonably be assumed that the snake lengths found in nature are normally distributed about some mean and standard deviation. Use scipy's curve fitting capabilities to fit a Gaussian curve (see Lecture Notes 07) to your histogram and *add this curve* to your plot. Make sure it looks pretty good; if not, consider adjusting the number of bins you are using for the histogram or providing initial guesses to the curve fitting function to help it find the correct fit values.

   (e) (3 points) State how well your fit values for the mean and standard deviation compare to the values you found in part (a).

   (f) (3 points) For values pulled from a normal distribution, it is known that roughly 68% of them are within one standard deviation of the mean. *Using your fit values*, state what you can conclude from the gathered data about the range of snake lengths that make up 68% of the population of that snake species.

   Note: this problem only requires a single Python *script file.*

C++

3. (20 points) C++ doesn't have a built-in factorial (!) function, but we can write our own easily enough:

```
int factorial(int n)
{
     int retval = 1;
     for(int i = 1; i <= n; i++)
          retval *= i;
     return retval;
}
```

Don't worry about figuring out how this function works. Just type it into a C++ program after your #include statements and before your main function. From within your main function, feel free to test that the function works for some known values of n, say, $n = 0, 1, 2, 3$. Now, within your main function, use a for loop to evaluate and print out to the screen the factorials of **even integers**, starting with zero, and going up to some maximum value of your choosing. Have your program print out to the screen answers to the following questions:

- How many iterations does your loop have to make before you observe an overflow error?
- What can be done to fix this problem? Feel free to propose more than one solution.

Note: this problem only requires a single C++ source code file.

4. (15 points) Write a C++ program that:

- asks the user to input the name of their favorite movie, stores it to a variable, and then prints it out to the screen (you can add your own phrasing however you see fit as long as it includes the user's input);
- asks the user how many movies they've seen in their lifetime and stores it to a variable;
- uses a conditional statement and the user's answer to the previous question to print out a message depending on how many movies they've seen; if they've seen fewer than 10 movies, say something like "Wow, you've hardly seen any movies"; if they've seen more than 1,000 movies, say something like "Wow, you've seen a lot of movies"; and, for any number of movies between 10 and 1,000, say something like "You've seen a typical number of movies." Again, you can be creative with the phrasing so long as the logic is correct.

Note: this problem only requires a single C++ source code file.