Computer Science 111
Lab Report Guidelines

With each project, each *individual* will hand in a short report (2–3 pages) that explains your program and your results. The report should contain the following sections:

1. **Title**

   Include the title of the project, your name, your partner's name, the course number, and the date.

2. **Introduction**

   Explain the purpose of the project (or the question(s) it is trying to answer). Provide some background on the problem. Why is this problem interesting? Write 1-2 paragraphs.

3. **Methods**

   This section has two parts.

   (a) Explain, in words, the algorithm(s) you followed to solve the problem. Describe what the algorithm does, not necessarily how it does it. Take some time to do this carefully. Read what you have written! Would someone with no background on this problem understand how the algorithm works? Would you be able to recreate your program based on this description? This should be about 2 paragraphs.

   For example, suppose you wrote the following program to play a guessing game (but, yours would need to have a docstring for each function, like in Section 3.4):

```
import random

def guessingGame(numGuesses):
   secretNumber = random.randrange(1, 101)  # computer picks a number 1-100

   for guess in range(numGuesses):
      myGuess = input("Please guess my number: ")  # get a guess from user

      if myGuess == secretNumber:                  # if guessed correctly,
         print("You got it!")                      #   print message and
         return                                    #   break out of the loop
      elif myGuess < secretNumber:                 # otherwise, give a hint
         print("Too low.  Try again.")
      else:
         print("Too high.  Try again.")

   print("Too bad.  You lose.")                    # user loses :(

def main():
   guessingGame(10)

main()
```

   For this simple example, a description in words might be:

   > In this game, the player is given 10 chances to guess a secret number. For each guess, it prints a prompt and waits for a response. If the guess is too high or too low, it prints out a message saying this. On the other hand, if the guess is correct, it prints `"You got it!"` and ends. If the players uses up all of her guesses, it prints `"Too bad.  You lose."`

Note that this description does not replace comments in your program. Program comments and docstrings are still an important component of your documentation, and part of your grade.

(b) Explain how you implemented your algorithm in Python. Indicate what functions your program contains, what the parameters are for those functions, what modules are used, and generally *how* each function works. Focus on the new techniques or data types that you have learned with that project. You do not need to explain every detail, especially later in the semester when concepts like `for` loops and `if` statements have become more commonplace. Explain the overall flow of the program and how functions interact (i.e. when one function calls another). Please use **bold font** to visually highlight the function names and Python constructs (e.g. **for loops, if-statements, matplotlib**, etc.

For example, for the guessing game program, you might write:

> My program contains two functions: `guessingGame` and `main`.
>
> The **guessingGame** function takes a single integer parameter, **numGuesses**. Then it selects a random number between 1 and 100. Next, it uses a **for loop** to give the player up to **numGuesses** guesses. For each guess, we print a prompt and wait for a response with the **input** function. We then use an **if/else** statement to determine if the guess was correct. If it is, we print a message and return from the function. Otherwise, we print a message indicating whether the guess was too high or too low. If we arrive at the end of the loop, the player must have used all of her chances and not guessed the correct number, we print a message indicating that she lost.
>
> The **main** function is where the program begins. It simply calls the **guessingGame** function with the parameter 10.

4. **Results and Conclusions**

   Explain your results. Be sure to answer all reflection questions in detail (each will be graded). Be sure to include screenshots of all graphs produced by your code (on several inputs), and explain what they mean in the context of the lab (e.g. explain what the graph tells you about how the populations change over time). If you went beyond the basic requirements of the project, explain here what you did. Talk about what you learned from the project.

5. **Group Notes**

   In a sentence or two, indicate how you and your partner worked together on the project. Did you split up the work or did you always work together? Was your partner an equal participant or did one person do more work than the other? This information will be kept confidential from your partner, so please be frank.

Writing your individual report

Although you will complete each project with a partner, you will hand in your own *individual project report*. It is natural (and encouraged) that you will talk to your partner about all aspects of the project while you are working on it together, including all the code. Please hand in the same code as your partner, as only one may be graded. If your code is different, please explain why in the Group Notes section. Comments and docstrings should be the same for both partners. Please be sure to include the names of both partners in the program docstring and in each lab report. While it is tempting to break up the work (one partner writes the first function and the other writes the second), to get the full value of the lab, you should work with your partner to write all the code together. This is the best way to understand how the code works, which you will need to know when you write your lab report. The code should be completed 24 hours before the lab is due, to give you a full day to work on your individual lab reports.

When it comes time to write your lab report, you must do so alone. For example, in a project where the goal is to empirically derive some quantitative value(s), I would expect that you write the code to derive these values together with your partner, and I would expect to see the same value(s) from both partners in their reports. However, it would not be appropriate for both partners to have an identical (or even too similar) sentences in their reports describing the project or results. Collaborating on the lab report is a violation of the honor code analogous to plagiarizing an essay.

In summary, work closely together on everything up to the report. When it comes time to write the report, do this individually and separately (no longer sitting next to or communicating with your partner). Please turn in the same

code as your partner, or include a remark in your lab report explaining why you have different code (e.g. if you disagreed about how to implement something). Please upload your code (as a .py file with your last name in the filename, e.g. `white_lab1.py`) and your lab report (as a .pdf file with your last name, e.g. `white_lab1report.pdf`) to Note Bowl. If you typed your lab report as a Word file, you can Save As (or Export As) a pdf.

Project grading

Each project will generally be evaluated roughly as follows, though if projects contain less programming then the lab report will be worth more. Normally, partners will receive the same grade on the first two items below, but if one partner has not contributed, has been skipping meetings, etc. then I reserve the right to take point off from the partner who is slacking.

Program satisfies project guidelines	40%
Program style and comments	10%
Report	50%

The Results section should include screenshots of all graphs produced, sentences interpreting these graphs, any numerical results or outputs requested, and answers to all Reflection Questions in the Project.