Project 2

Due Friday, September 30 by 4:00pm

For this second project, you will use a random simulation to approximate the average maximum hourly customer volume for a company. Through the power of programming, we can do this without any specialized statistical knowledge. Suppose, for this rough example, a company is expecting `C` customers per day, and is open for `H` hours a day. For simplicity, there is no "rush hour" for this company, customers are equally likely to appear at any hour (we call this a "uniform distribution"). But even without a "rush hour", customers will still randomly show up in large amounts, which stresses the company resources. With a desire to plan ahead, we'd like to know how many customers to expect during whichever turns out to be the busiest hour. Is there a way to figure this out precisely using advanced statistical knowledge? Probably, but let's try doing a "Monte Carlo" simulation using Python instead. What you must do is create a random time for each customer's visit in a day, and keep track of how many customers are visiting during each of the `H` hours the business is open. Afterwards, we find the maximum number of customers in one hour. Then we redo this for `D` days and take the average of the maximum numbers of costomers in one hour. Note that `D` controls how many days will be simulated and averaged over. The idea is that increasing `D`, and hence increasing the number of simulations, will result in a more accurate approximation of the "true" average maximum hourly customer volume that we should expect to see in the long run. Of course, you can't make `D` too high, or else the computer won't finish computing the simulations in a reasonable amount of time. This is what your program should look like this when it is run (however, your output may differ slightly since random numbers were used):

```
Hello, welcome to my average maximum hourly customer volume calculator!
Give the number of customers per day:  100
Give the number of business hours per day:  10
Give the number of days to simulate:  750
The average maximum hourly customer volume is:  15.156
```

For full credit, as well as for improved code organization, you must wrap up the main part of your program in a **function** named `ave_max_calc(C, H, D)`, with **inputs** being the three variables described above, and which **returns** the average maximum hourly customer volume. Your code should start with the definition of the function, and outside the function you can print the above statements prompting user input. The last line of your code should look something like `print(ave_max_calc(<inputs>))`, which calls your function (with some inputs provided by the user) and prints the output.
Now for a few comments:

1) You should use the `random` module for this project. Feel free to use, for instance, `random.random()` to get a random number from 0 to 1.

2) You should also make use of lists and loops. Consider using "list comprehensions".

3) In real life, clock hours are notated as, for instance, 9:00am - 10:00am. For our programming purposes, however, you don't need label your "hours", just think of them as the first hour, the second hour, etc.

4) When you're done, you can play around a bit with probability. When `C` is really large, what seems to be the relationship between it and the average maximum hourly customer volume? When a business with 100 customers spread out over 10 business hours is thinking of increasing the number of business hours to 15, how many more customers must the business get to justify the cost, assuming the daily revenue increases proportionally to the number of customers and hourly expenses increases proportionally to the average maximum hourly customer volume?

Lastly, a few rules that apply to all projects:

1) First and foremost, the Python interpreter must be able to execute your script. It is better to have a running but partial solution than a program that attempts to do all computations, but that fails to execute. Handing in an incomplete but working program is better than handing in a program that crashes or does not run at all.

2) The first line of your Python program must be

   ```
   # MCS 260 Project One by Author
   ```

   where you replace `Author` by your name.

2) Add comments to clarify your choice of variables and to indicate what is happening at different steps of the program.

3) Programs will be graded partly on the basis of good hygiene. Programs should be readable with a logical, clear structure and they should avoid unnecessary or overly convoluted calculations. The "PEP 8" style guide for Python helps to improve your grade in this regard.

4) Your .py file must be sent to me via email as an attachment and **in addition** must be printed out and submitted in class (so I can comment on your work easily). My email address is jbergen@uic.edu.