Homework 1: K-armed Bandit Algorithms
Due: Monday, September 14th 11:59 pm

The purpose of this project is to study different properties of multi-armed bandit algorithms.

Part I
Build a testbed by generating 500 randomly selected k-armed bandit problems with k = 7. For each bandit problem, select the true action values from a Gaussian distribution with mean 0 and variance 1. For each action a, select an actual reward value from a normal distribution with mean $Q^*(a)$ and variance 1. For an algorithm, one *run* includes playing a single bandit problem for 1000 time steps. The algorithm's behavior will be evaluated by averaging its performance over 500 bandit problems.

Part II
Implement the sample-average algorithm and run it on the testbed you developed in previous part according to the following settings:

1. Using greedy action selection,
2. Using ε–greedy action selection with $\varepsilon = 0.01$ and $\varepsilon = 0.1$,
3. Using upper-confidence bound action selection with $c = 1$ and $c = 2$.**(*)**

For all three settings, plot the *average reward* and *%optimal action* graphs. Then answer the following questions:

1. Which action selection method performs worse than others? Why?
2. Which ε value improves faster? What is the best average reward value?
3. Which ε value will perform best in the long run in terms of cumulative reward and probability of selecting the best action? How much better will it be?
4. What is the difference between results from $c = 1$ and $c = 2$? Why? **(*)**
5. Why is there a performance spike on the 8th step for the UCB method? **(*)**

Part III (for extra credit)
Implement the Gradient Bandit algorithm and plot the *average reward* and *%optimal action* graphs for the testbed you developed according to the following settings:

1. Using $\alpha = 0.01, \alpha = 0.1, \alpha = 0.5$.
2. Using no reward baseline, reward baseline of +5, reward baseline of +10

Answer the following questions:

1. How do you compare the effect of reward baseline (discuss all scenarios)?
2. How do you compare the effect of step size (discuss all scenarios)?

Evaluation: we will grade your submission according to the following table:

Item	COMP4600	COMP5300
Implementation of the testbed	30	20
Implementation of sample-average algorithm with given settings	40	40
Plotting the results and answering questions in Part II	30	40
Implementation of gradient algorithm with two settings (for extra credits)	10	10
Plotting the results and answering questions in Part III (for extra credits)	10	10

Note 1: The parts marked with **(*)** are optional for COMP4600 (undergraduates) but mandatory for COMP5300 (graduates). In this homework, the Part III questions are optional for all students and are given for extra credits.

Note 2: We do not accept code in any programming language other than Python 3 (do not use Python 2).

Note 3: For the implementation of the testbed and the algorithms, you are not allowed to rely on any python library other than `numpy` and `matplotlib` (for plotting).

Note 4: All the code, plots, explanations, and answers should be included in a single Jupyter Notebook (`.ipynb`) file. Include your name as part of the filename and submit through Blackboard.

Submission: By 11:59pm on Monday, September 14th 2020, submit both your `student_name.ipynb` files on Blackboard. Make sure everything is entirely contained within this file and it runs without any error.

Late Policy: Up to two late days are allowed, but a grade penalty of 50% and 75% will be applied at the first and second day, respectively.