1026B: Assignment 2: How to Invest Your Money?

Due: February 28 (9pm)
Weight: 8%
Goal:

- **Use Python to solve interesting real-world problem, with an emphasis on problem-solving and algorithm design skills**
- **By completing this assignment, you will gain skills relating to**
  • using loops,
  • defining & using functions,
  • using lists in Python.

There are two "investment machines". If you put $X in one machine, there is 60% of chance it will return $2*X (and 40% returning $0); if you put $X in the other, there is 40% of chance it will return $2*X (and 60% returning $0). ***You just don't know which machine is which!*** You have a total of $100 to start with, and you can only invest a maximum of 30 times. The minimum amount to invest is $1.

Use Python to implement the following investment strategy:

1. Put $1 in each investment machine for 10 times to see which one returns more money (thus, you have invested a total of 20 times). Break tie whatever way you prefer.
2. For the remaining money, always invest half (of the current total money) in the more profitable machine discovered each time for the last 10 times.

Simulate this strategy for 100 times. Print out the average amount of money you have with this strategy.

[Bonus Question] Can you design and implement better investment strategies? Students with the top 3 average amount of money will receive 2% as bonus; the next 3 students will receive 1% as bonus.

Note: The first step above is often called "exploration" which uses trial-and-error to see which option is best. The second step is often called "exploitation" which utilizes the newly learned best option to make a maximum "profit". Exploration and exploitation can be interleaved and integrated in sophisticated ways.

We are making many simplifying assumptions here; in real-world, things are often much more complicated. For example, there are usually more investments to choose from, each with an unknown distribution of returns which may vary over time, and so on. These types of problems do have widespread important applications. The "investments" can be the new restaurants you are exploring and enjoying, different subjects you try to choose in schools, or different jobs/career you want to work for in your life. The assignment may also be helpful to improve rational decision-making under uncertainty.

The Python code for the two investment machines will be provided to you as an independent python file called investment_machine.py in which a python class ‘InvestmentMachine’ was defined with a function called ‘invest”. **Do not** modify investment_machine.py, just put it with your own python file in the same directory.

The following sample Python program invest $3 in each machine 15 times, and print out the final total. It may help you start with your assignment. Save the following codes in any_name.py, in the same directory as investment_machine.py. Load and run any_name.py in PyCharm. You would get a total return value of around $100.

```
##################
#import Investment Machine class
from investment_machine import InvestmentMachine
#initialize investment machine instance
my_investment_machines = InvestmentMachine()

total_reward_machine_1 = 0
total_reward_machine_2 = 0

#invest to machine 1 with $3 for 15 times
for times_machine_1 in range(15):
  # v1 += v2 is shorthand for v1 = v1 + v2
  total_reward_machine_1 += my_investment_machines.invest(1, 3)
  # The first argument is for machine 1 or machine 2, randomly assigned to have 40% and 60% return
  # The second argument is for the amount of money put in.

#invest to machine 2 with $3 for 15 times
for times_machine_2 in range(15):
  total_reward_machine_2 += my_investment_machines.invest(2, 3)

print("The total return of my investment strategy is $", total_reward_machine_1 + total_reward_machine_2)
```

You can implement the investment strategy for this assignment without using functions (and you could start without using functions), but we ask you to use the following functions in your final submission:

a. Define and use the main function
b. Define and use a function for repeated investment of a fixed amount in one machine. The function should take three arguments: the machine number, the money you invest each time, and total times you invest with the fixed amount.
c. Define and use a function for repeated investment of a fixed percentage of the current total money you have in one machine. The function should take three arguments: the machine number, the percentage you invest each time, and total times you invest with the fixed percentage.

You could combine b and c above into one function (not required to get the full mark).

In addition, we want you to practice the list functions. Keep the total return for each of the 100 simulations in a list, and use the list method to get the average of the 100 simulations. The list would also be useful when further analysis is needed to design better strategies.

Note: The assignment is to be done individually and must be your own work. Software may be used to detect cheating.

What You Will Submit and Be Marked On:

1. A 1-3 page written part (with Word, text, PDF file) about:
   - A brief problem-solving process you use in designing and implementing your Python program
   - The output of your program. You can copy/paste from the Output window.
   - If you work on the bonus question, explain your strategies in more detail.

2. Your Python source file(s). Make sure that your code meets the requirements (such as using functions and list described above). The name of the Python program you submit should be your UWO userid_Assign2.py. Make sure you attach your Python file to your assignment; DO NOT put the code inline in the textbox or the written part above. Make sure that you develop your code with Python 3.6 as the interpreter. TAs will not endeavor to fix code that uses earlier versions of Python.

You can submit multiple files if you work on the bonus question.

3. Non-functional specifications: as described below

Non-functional Specifications:

1. Include brief comments in your code identifying yourself, describing the program, and describing key portions of the code.
2. Assignments are to be done individually and must be your own work. Software may be used to detect cheating.
3. Use Python coding conventions and good programming techniques, for example:
   i. Meaningful variable names
   ii. Conventions for naming variables and constants
   iii. Use of constants where appropriate
   iv. Readability: indentation, white space, consistency