111A Introduction to Computer and Computer Science

Homework Assignment #5

Due: 11/7 12:00:00

In this homework assignment, you will practice how to create your own functions by solving one of the most difficult issues in your entire life – managing your money for something. As you have learned all the related knowledge during the lectures (if, while, for, def, and so on…).

Problem #1: Set the goal and find ways to achieve them

Now you have explored how the percentage of your salary you saved in each month and your annual raise affect your saving plan. However, in practical, you might want to set a particular goal, e.g., to be able to afford the down payment in four years. Therefore, how much money you should save in each month becomes the next critical issue. In this problem, you are going to write a program to solve this issue.

Before you start to write the code, here are some basic assumptions for this problem:

1. Call the cost of your target "**target_price**".
2. Assuming the down payment is 25% of your target price, which leads you this result $\text{down payment} = 0.25 \times \text{target price}$.
3. Assuming your monthly salary is "**monthly_salary**".
4. Assuming you are going to dedicate a certain percentage of your monthly salary for the down payment. Call it "**saving_rate**". Be careful, this number should be in decimal form, i.e. $\text{saving rate} = 0.87$ for 87%.
5. Call the amount that you have saved "**current_savings**". You should start with $\text{current savings} = 0$ unless you have wonderful parents or you are "Fa Dà Cái".
6. Assuming you also invest your current savings wisely, with an annual return of "**r**". In other words, your account of current savings will receive an additional money $\left(\text{current savings} \times \frac{r}{12}\right)$ at the end of each month. For simplicity, let assume $r = 0.04$.
7. Call your semi-annual salary raise "**semi-annual_raise**". Again, it should be a decimal number, i.e. $\text{semi-annual raise} = 0.05$ for 5%.
8. Raise your salary by this rate in **every six months**.
9. Call your objective as "**target_month**". For simplicity, let us assume the value of this variable is **an integer**.

Please write function called "**findSavingRate**" that aims to find the best saving rate for your down payment in certain months. Besides, your function should return a message "**NA**" if it is impossible to achieve the "**target_price**" within the "**target_month**".

Sample code:

```
def findSavingRate(target_price, monthly_salary, annual_raise,
                   target_month):
    """
    Params:
        target_price:     int, target_price >= 0
        monthly_salary:   int, monthly_salary >= 0
        annual_raise:     float, annuam_raise >= 0, raise monthly_salary
                          after every 12 months

    Returns:
        saving_rate:      float or string, return "NA" if impossible

    """
    downpayment = 0.25 * target_price
    current_saving = 0
    annual_return_rate = 0.04

    return saving_rate
```

Problem #2: Population translation

There are two towns, town A and town B. The ratio of the population of town A to the total population is ***r***, town B have the ratio ***1-r***. The population flows to each other year by year under influence of various factor. For simplicity, we denote $\boldsymbol{R_0}$ as initial population rate matrix and $\boldsymbol{P_{tran}}$ as population translation matrix.

$$\boldsymbol{R_0} = \begin{bmatrix} r \\ 1-r \end{bmatrix}, 0 \le r \le 1$$

$$\boldsymbol{P_{tran}} = \begin{bmatrix} p_A & p_B \\ 1-p_A & 1-p_B \end{bmatrix}, p_A \& \, p_B \in [0,1]$$

$$\boldsymbol{R_{n+1}} = \boldsymbol{P_{tran}} \boldsymbol{R_n}$$

Please write a function called "**PopulationTranslation**" that calculates the population rates after ***n*** year, given initial population rates R_0 and P_{tran}. You can **only use recursive algorithm**.

```
def PopulationTranslation(Ptran, R0, n):
    """
    Params:
        Ptran: list[list[float, float], list[float, float]],
       ex. [[0.8, 0.5],[0.8, 0.5]]
        R0   : list[float, float], ex. [0.8, 0.2]
        n    : int, n >= 1
```

Hand in procedure:

As we had mentioned in the lecture, you should list all your collaborators in your programs. Here is the template:

```
"""
Created on Sun Aug 7 01:23:45 2022

@author: Xi Winnie, student ID

@collaborators: Jane Doe, her student ID
                John Doe, his student ID
"""
```

Please save your code as a ".py" file, where the file name should follow this format:

111A_hw#5_ID.py

For example,

111A_hw#5_0123456789.py

Please be aware. **We are not going to accept any homework file with wrong file name or without signature**. Please double check the content of your file.

Once you have accomplished your works, you can upload your homework to the "E3@NYCU" system. There will be a section for uploading your homework.