111A Introduction to Computer and Computer Science

Homework Assignment #10

Due: 12/19 12:00:00

You might be unfamiliar with the rule of mortgage in modern financial system. But some of you might have some experience in applying the student loan for the tuitions. In this homework, we are going to develop a program, which aims to calculate the monthly repayment of mortgage (or loan) of total amount l at lending rate r for m months. At the end of this homework, we are also going to plot some figures for visualizing the outcome of your program.

Problem #1: Calculate the monthly repayment for Constant Payment Mortgage (CPM) with fixed lending rate

The most commonly used system for calculating the repayment of loans or mortgages is **Constant Payment Mortgage (CPM)**. As the name implies, the repayments are constant. This means that all of the repayments are the same throughout the entire duration of the loan, unless the interest rate changes. Assuming the amount of loan is l with fixed lending rate r (monthly) for m months, the repayment p can be derived as shown below:

Months	Balance of loan
0	l
1	$l \cdot (1+r) - p$
2	$l \cdot (1+r)^2 - p \cdot (1+r) - p$
3	$l \cdot (1+r)^3 - p \cdot (1+r)^2 - p \cdot (1+r) - p$
…	…
m	$l \cdot (1+r)^m - \sum_{n=0}^{m-1} p \cdot (1+r)^n = 0$

I believe that you should have the ability to derive p.

Please create a superclass called "**Mortgage**" and a child class called "**FixedRate**" to calculate a mortgage. Here is the sample code:

```
class Mortgage:
        def __init__(self, amount, annRate, months):
            """
            Parameters:
              amount, amount of mortgage, a positive integer
              annRate, annual rate, a positive number in the range of
              [0,1]
              months, total months for mortgage, a positive integer
              larger than 12
            """
            self.l = amount
            self.r = annRate/12
            self.m = months
            self.paid = [0]     # to record the monthly progress of
                                # repayment
            self.outstanding = [amount]
                                # to record the progress of repayment
            self.p = self.findPayment(self.l, self.r, self.m)
                                # monthly repayment

        def findPayment(self, l, r, m):
            """Method for calculating the monthly repayment"""
            ???

        def makePayment(self):
            """Method for recording the monthly repayment and
             outstanding"""
            ???

    class FixedRate(Mortgage):
        """Mortgage with fixed rate"""
        def __init__(self, amount, annRate, months):
            ???
            self.label = "Fixed rate: {:.02f}%".format(annRate*100)
```

Problem #2: Calculate the monthly repayment for Constant Payment Mortgage (CPM) with variable rate

In practical, a mortgage with fixed lending rate is very unusual. The lender always wants to extract more profit from the borrower. That makes the lending rate become variable, which is well-known as the variable-rate mortgage. Variable-rate mortgage means the interest rate of a mortgage is fixed for the first m_1 months (e.g., r_1 for the first m_1 months). At the end of $m_1 + 1$ months, the interest rate will be changed to r_2 for the rest of the entire mortgage life. For example, the lender may set an initial rate for the first 2 years (let's say, 1% for the first 2 years), and then it goes up over time (5% for the rest of years, or gradually increasing by 0.5% every year for the rest of years). In this case, the monthly repayment p will be different during this duration, which means you need to calculate a new repayment p at the end of $m_1 + 1$ months according to the outstanding balance.

Please create a child class called "**VarRates**" to calculate a variable-rate mortgage. Here is the sample code:

```
class VarRates(Mortgage):
    """Mortgage with fixed rate annRate1 for the first months1,
    annRate2 for the remaining"""
        def __init__(self, amount, annRate1, months, annRate2, months1):
            """
            Parameters:
              annRate1, the annual rate for the first m1 months,
              a positive number in the range of [0,1]
              annRate2, the annual rate for the rest of mortgage,
              a positive number in the range of [0,1]
              months1, the first m1 months,
              a positive integer larger in the range of [1, months]
            """
            ???
            self.m1 = months1
            self.r2 = annRate2/12
            self.label = "{:.02f}% for the first {} months, then
{:0.2f}%".format(annRate1*100, months1, annRate2*100)

        def makePayment(self):
            """Override the method"""
            ???
```

Please accomplish this homework with an organized code (e.g., with main script and function script). Here is a template for your code structure:

```
111A_hw#10_0123456789
      ├─ func.py          # Functions
      ├─ obj.py           # Objects
      └─ main_hw10.py     # Main scripts of hw10
```

You don't need to follow this structure, just keep your main script clean.

If everything goes well, your code might get this result:

```
(nycudopcs)                                  \Homeworks\HW10>python main_hw10.py --amt 500000 --y 6
 --fixRates 0.05 0.07 0.1 --vRates 0.03 0.075 0.02 0.1 --months1 12 12
Fixed rate: 5.00%
Total Paid: 579744
Fixed rate: 7.00%
Total Paid: 613800
Fixed rate: 10.00%
Total Paid: 666936
3.00% for the first 12 months, then 7.50%
Total Paid: 599484
2.00% for the first 12 months, then 10.00%
Total Paid: 624900
```

Total Paid of Fixed Rates: 5%, 7%, and 10%

Total Paid of Variable Rates: (3%, 7.5%) and (2%, 10%)

FRIENDLY REMINDER

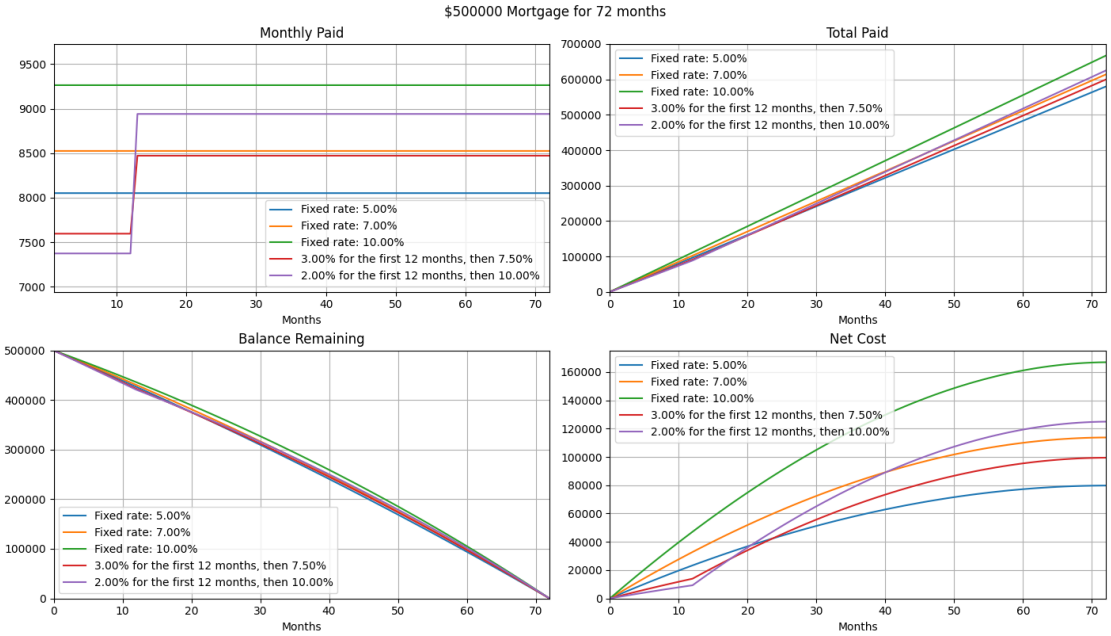
It is **NOT** necessary to calculate multiple mortgages at once if you struggle to put multiple inputs into one parameter (such as multiple fixed rates or variable rates as shown above).

(Bonus) Problem #3: Visualize your result

Please plot the result:

1. Monthly Paid vs. Months
2. Total Paid vs. Months
3. Balance Remaining vs. Months
4. Net Cost vs. Months

For example:



FRIENDLY REMINDER

You don’t need to plot a figure that is exactly same as mine.

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 "**.zip**", "**.7z**", or "**.rar**" file, where the file name should follow this format:

111A_hw#10_ID.zip

For example,

111A_hw#10_0123456789.zip

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.