Exercise: Using Functions to Compute NPVs

Summary

An introduction to defining and using functions.

Input Data

Three input files are provided that give cash flows for different projects: **camry.txt**, **volt.txt**, and **solar.txt**. The first two give the total cost of ownership over five years of a Toyota Camry and a GM Volt using data on purchase prices, fuel costs, and resale value (as of a couple of years ago). The last file gives a rough estimate of the costs and electricity savings of a solar power system proposed for use in teaching and research at SU.

Deliverables

Your finished repository should include one new file: **npv.py**, a script that computes the NPVs of the cash flows in the input files.

Instructions

The overall approach will be to create two functions, one that reads in a cash flow from a file and returns it as a list of payments with dates, and one that computes the NPV of a list of payments. Following that, the functions will be applied to the three input files.

1. Define a function called `read_cashflow` that accepts one string parameter, `filename`, and returns a list. Use type hinting in the definition. Within the function, do the following:

    1. Create an empty list called `payments`.

    2. Open the file whose name is given by `filename`.

    3. Loop through the file line by line doing the following:

        1. Split the line on whitespace using `.split()`.

        2. Create a new dictionary called `new_pmt` with two attributes: `t`, which should be set to the numerical value of the first item on the line (the year) using the `int()` call, and `amt`, which should be set to the numerical value of the second item on the line using the `float()` call.

        3. Append `new_pmt` to `payments`

    4. After the loop completes, the function should close the file and return `payments`.

2. Define a function called `npv` that accepts two parameters, a float `r` giving an interest rate in decimal form (e.g., 0.02 rather than 2 for two percent) and a list of payments in variable `cashflow`, and returns a float. As before, use type hinting in the definition. Within the function, do the following:

    1. Create a variable called `val` and set it to 0.

    2. Loop through `cashflow` using `payment` as the loop variable. For each `payment`, the loop should do the following:

        1. Use the values of the payment's `t` and `amt` attributes to compute the PV of the payment using the interest rate `r`.

        2. Add the result to `val`

    3. After the loop completes, the function should return `val`.

3. After the functions are defined, `npv.py` should compute and print the NPV of the cash flow in each of the three files using an interest rate of 5 percent. The print statement should give the name of the input file and then the NPV rounded to the nearest integer.

Submitting

Once you're happy with everything and have committed all of the changes to your local repository, please push the changes to GitHub. At that point, you're done: you have submitted your answer.

Tips

- Don't be surprised if the results are all negative. The car calculations are total costs, so they're naturally negative, and the solar panels don't generate enough electricity to cover their costs. Real solar projects often involve tax credits, which are not included here and would make the calculation more favorable.
- It would be a best-practice to do the calculations at the end with a loop over a list consisting of the names of the input files. It's not required but you might want to try setting it up that way.