

Hack 4.0

Computer Science I

Department of Computer Science & Engineering
University of Nebraska–Lincoln

Introduction

Hack session activities are small weekly programming assignments intended to get you started on full programming assignments. You may complete the hack on your own, but you are *highly encouraged* to work with another student and form a hack pair. Groups larger than 2 are not allowed. However, you may discuss the problems *at a high level* with other students or groups. You may not share code directly.

If you choose to form a Hack Pair, you *must*:

1. Both join a hack pair on Canvas (go to People then Hack Pairs)
2. You must both work on the hack equally; it must be an equal effort by both partners. Do not undermine your partner's learning opportunity and do not undermine your own by allowing one partner to do all the work.
3. Turn in only one copy of the code under the individual whose last name comes first (with respect to Canvas).

You are graded based on style, documentation, design and correctness. For detail, see the general course rubric.

Category	Point Value
Style	2
Documentation	2
Design	5
Correctness	16
Total	25

Table 1: Rubric

Correctness: Formatting variations are perfectly fine. You need to report *just as much* information as the expected output. Points are awarded proportionally per test case. Values must be rounded correctly.

Problem Statement

A 401(k) plan is a type of tax-qualified account for people to save for retirement. Employees typically have part of their pre-tax pay check deposited into such an account and employers may match a certain amount of those contributions. As of 2018, there is a maximum annual contribution limit of \$18,500.

The money contributed to a 401(k) account may be invested in many different ways. While the account may grow tax free, it is not immune to inflation. To account for inflation, you can use the following formula which is the inflation-adjusted rate of return.

$$\frac{1 + \text{rate of return}}{1 + \text{inflation rate}} - 1$$

Write a program that produces an *amortization table* for a 401(k) account. Your program will read the following inputs as command line arguments.

- An initial starting balance
- A monthly contribution amount (we'll assume its the same over the life of the savings plan)
- An (average) annual rate of return (on the scale [0, 1])
- An (average) annual rate of inflation (on the scale [0, 1])
- A number of years until retirement

Your program will then compute a monthly savings table detailing the (inflation-adjusted) interest earned each month, contribution, and new balance. To get the monthly rate, simply divide by 12. Each month, interest is applied to the balance at this rate (prior to the monthly deposit) and the monthly contribution is added. Thus, the earnings compound month to month. Be sure to round appropriately, to the nearest cent so that rounding errors do not compound.

Your program will need to validate the input and quit with an error message on any potential bad input(s). For inputs

```
10000 500 0.09 0.012 10
```

your output should look something like the following.

Month	Interest	Balance
1 \$	64.23 \$	10564.23
2 \$	67.85 \$	11132.08
3 \$	71.50 \$	11703.58
4 \$	75.17 \$	12278.75
5 \$	78.87 \$	12857.62
6 \$	82.58 \$	13440.20
7 \$	86.33 \$	14026.53
8 \$	90.09 \$	14616.62
9 \$	93.88 \$	15210.50
...		
116 \$	678.19 \$	106767.24
117 \$	685.76 \$	107953.00
118 \$	693.37 \$	109146.37
119 \$	701.04 \$	110347.41
120 \$	708.75 \$	111556.16
Total Interest Earned: \$		41556.16
Total Nest Egg: \$		111556.16

Take, as an example, the first month's calculation. The inflation-adjusted annual rate would come to:

$$\frac{1 + .09}{1 + .012} - 1 = 0.077075 \dots$$

and so monthly would be $0.0064229 \dots$. With an initial balance of 10,000, that gives us 64.229249 dollars and when rounded gives us the \$64.23 in the table.

Instructions

- You are encouraged to collaborate any number of students before, during, and after your scheduled hack session.
- Design at least 3 test cases *before* you begin designing or implementing your program. Test cases are input-output pairs that are known to be correct using means other than your program.
- Include the name(s) of everyone who worked together on this activity in your source file's header.
- Name your program `retire.c`, and turn it in via webhandin, making sure that it runs and executes correctly in the webgrader. Each individual student will need to hand in their own copy and will receive their own individual grade.