The Case of the Greedy Boss

Help Center

In life, you are often faced with a fundamental choice: do I enjoy the benefits of the resources that I have earned or invest those resources so as to improve my life in the long-term? For example, if you've graduated from college, you may have been faced with the following choice:

- Find a job and enjoy the immediate benefits of earning a reasonable salary or,
- Go to graduate school (while paying tuition and living on a shoestring) with the prospect of getting a better job in the future.

One of the reasons for the popularity of real-time strategy games like Warcraft 3 and Starcraft is that they allowed the player to experiment with different strategies for resources allocation. One strategy might be to immediately generate lots of weak low cost units and try to overwhelm an opponent who is investing in developing the technological capability to build more powerful (and costly) units.

This week's mini-project involves building a simulator for the web-based game Cookie Clicker. In Cookie Clicker, your goal is to bake as many cookies a possible in a given period of time. (Or alternatively, bake a fixed number of cookies as quickly as possible.) Game play in Cookie Clicker involves choosing strategically among several methods of upgrading your ability to bake cookies with the goal of baking as many cookies as quickly as possible.

The greedy boss scenario

Since the simulation that makes up Cookie Clicker is fairly complex, this practice activity considers a simpler scenario as described in the following forum post from the first session of Principles of Computing.

"Let's say you have a job that pays a salary of $100 a day. You know your boss can be bribed to increase your salary to $200 dollars a day if you pay him $1000. How long would it take you to earn $1000 for the bribe? How much would you paid be after the bribe? Here is a good question to get you thinking in the right direction. If you bribed your boss as soon as you have enough money saved up for the bribe, how much money would you have earned (bribes included) in 20 days?

Now, what happens if your boss is really greedy and will increase your salary by $100 dollar per day every time you give him $1000 dollars? How fast would your salary increase? Finally, let say that your boss is both greedy and smart. He wants a bigger bribe every time he increases your salary. What would happen? That is basically the scenario in Cookie Clicker."

Implementing a greedy boss simulator

Your task in this activity is to write a simulator for the greedy boss scenario described above. This simulator will follow a simple strategy designed to increase your current salary as quickly as possible. In particular, the simulator should bribe the boss at the end of the first day where your current savings is sufficient to afford the cost of the bribe. To get you started, we have created program template that implements some of the non-essential parts of the simulator.

In your simulator, you should assume that your initial salary is $100 per day and that each bribe paid to your boss increases your salary by $100 per day. You can also assume that the cost of the initial bribe is $1000. Your main task is to complete the body of the function `greedy_boss(days_in_simulation, bribe_cost_increment, plot_type)` which takes as input the number of days in the simulation (an integer) and the amount by which the boss increases the cost of a bribe after each bribe (an integer). The final parameter has either the constant value `STANDARD` or `LOGLOG` and specifies whether the

returned list is either in standard scale or log/log scale.

The function `greedy_boss()` should return the list `days_vs_earnings`. To correspond to the start of the simulation, the first entry in the return list `days_vs_earnings` should always be the tuple `(0, 0)`. Earnings from your daily salary should then accumulate starting at day one and so forth. Subsequent tuples added to this list should consist of the day when a bribe takes place and the total salary earned up to and including that day. This total salary earned should include money spent on the current day's bribe as well as all previous bribes.

More details of the implementation

While implementing `greedy_boss`, you should settle on the various quantities that the simulator will need to maintain as part of the simulation and then initialize those quantities appropriately. You should then implement the body of the `while` loop in the template. Note that this loop should increment the variable `current_days` during each execution of its body. However, each iteration of the loop should advance the current directly to the day of the next bribe instead of incrementing the current day by a single day. Advancing the current day directly to the day of the next bribe makes the resulting simulation more efficient since nothing interesting happens while waiting to accumulate enough savings to afford a bribe. We will take this same approach in Cookie Clicker.

One important feature of the greedy boss scenario is that the boss accepts bribes only at the end of the day after your daily salary has been paid. As a result, any daily earnings that are unspent on the bribe should be retained for future bribes. Also, in some situations, your salary may be large enough that you can afford to pay the boss several bribes at the end of the day. Your boss will happily accept them and this behavior should be incorporated into your implementation of the simulator.

The bottom of the template includes two example inputs and outputs designed to help in debugging your code. (Note that the second example includes two bribes being paid at the end of day 34.) We suggest that you make a substantial attempt at completing this activity on your own. While peeking at our sample solution immediately is very tempting, working through this problem on your own is worthwhile. The behavior of the greedy boss simulator is designed to mimic the behavior of the simulator that you will build for Cooke Clicker. Time spent here will be time saved working on Cookie Clicker.

Experimenting with the greedy boss simulator

Once you are satisfied with your implementation of `greedy_boss`, you are welcome to consult our implementation. The function `run_simulation` calls the simulator for a specified number of days with various values for the cost increment of a bribe. If you examine the plots for total salary earned as a function of day passed, you will note (not surprisingly) that, if the boss doesn't ask for any increase in the cost of the bribe, total salary earned (including the cost of the bribes) increases fastest as a function of days past. As the size of the increment to the bribe increases, total salary earned increases at a slower rate. Much of Homework 5 will consider the behavior of the greedy boss simulation in more detail for various increments to the cost of bribes.

Created Thu 12 Mar 2015 1:41 PM PDT

Last Modified Mon 8 Feb 2016 2:31 PM PST