Written Homework 05

Assigned: Wed 4 Nov 2015
Due: Wed 18 Nov 2015

Instructions:

• The assignment has to be uploaded to blackboard by the due date. NO assignment will be accepted after 11:59pm on that day.

• We expect that you will study with friends and often work out problem solutions together, but you must write up your own solutions, in your own words. Cheating will not be tolerated. Professors, TAs, and peer tutors will be available to answer questions but will not do your homework for you. One of our course goals is to teach you how to think on your own.

• We require that all homework submissions be neat, organized, and typeset. You may use plain text or a word processor like Microsoft Word or LaTeX for your submissions. If you need to draw any diagrams, however, you may draw them with your hand.

• To get full credit, show intermediate steps leading to your answers, throughout.

Problem 1 [48 pts (16 each)]: Sums

i. Evaluate the following sum. You must show your work, and your final answer should be a single integer.

\[ \sum_{k=4}^{84} 9k = \]

Solution: This is an arithmetic sum. The first term is 9 \cdot 4 = 36, the last term is 9 \cdot 84 = 756, and there are 84 - 4 + 1 = 81 terms in all. Using the formula from class and the text, the sum evaluates to \( \frac{81}{2} \cdot (36 + 756) = 32076 \).

ii. Derive a formula in terms of \( n \) for the following sum. You must show your work, and your final formula should only contain \( n \) and integers (but not \( k \)).

\[ \sum_{k=4}^{n} 8k = \]

Solution: This is another arithmetic sum. The first term is 32, the last term is 8\( n \), and there are \( n - 4 + 1 = n - 3 \) terms in all. Using the formula from class and the text, the sum is:

\[ \frac{(n - 3)}{2} \cdot (8n + 32) = (n - 3)(4n + 16) = 4n^2 + 4n - 48 \]
iii. Derive a formula in terms of $a$ and $b$ ($b \neq 0$) for the following sum. You must show your work, and your final formula should only contain $a$, $b$, and integers (but not $k$).

\[
\sum_{k=4}^{84} a \cdot b^{-k} =
\]

**Solution:** This is a geometric sum. We cannot simply plug the values here into the formula from section 10.2.2 of the text, because that formula was for a sum starting with $k = 0$, and this one starts with $k = 5$. However, we can derive a formula for our sum with the same technique used in that section:

Let $S = \sum_{k=4}^{84} ab^{-k} = ab^{-4} + ab^{-5} + \cdots + ab^{-84}$ be the sum we want to compute. Then,

\[
(1/b)S = ab^{-5} + ab^{-6} + \cdots + ab^{-85}.
\]

Taking the difference of these two, we get $S - (1/b)S = ab^{-4} - ab^{-85}$, because all other terms cancel out. Then, we have:

\[
S - (1/b)S = \frac{(b-1)}{b}S = ab^{-4} - ab^{-85}
\]

And the answer is:

\[
S = \frac{b \cdot (ab^{-4} - ab^{-85})}{b-1} = \frac{ab^{-3} - ab^{-84}}{b-1}
\]

**Problem 2 [32 pts (8,12,12)]: Growth of Arithmetic and Geometric Sequences**

You have just graduated from college with a BS in computer science or software engineering. Two companies have offered you a job.

1. Company A offers you $50,000 plus a yearly raise of 5% of your salary.
2. Company X offers you $60,000 plus a yearly raise of $2,000.

i. Assuming that all other benefits being equal and assuming long term employment, which company’s offer would you accept?

**Solution:** You know geometric sequences grow faster than arithmetic sequences (Provided the geometric ratio $r$ is greater than 1), so you will go with company A.

ii. Does your salary from company A ever exceed your salary from company X? (Hint: you could use wolframalpha to solve the inequality or a graph system to find the crossover value). Explain your answer.

**Solution:** To answer the question you must solve the inequality for $n$

\[
a + bn \leq p(1 + r)^n
\]

where $a = 60,000$, $b = 2,000$, $p = 50,000$ and $r = 0.05$, or dividing by 50,000, reducing fractions, and writing in decimal notation

\[
1.2 + 0.04n \leq 1.05^n
\]
I’ve used a graphing system to see that the crossover value is when $x$ is little bigger than 9.

And you can check

\[ 50,000 \times 1.05^9 \approx 77566.41 < 60,000 + 2000 \times 9 = 78,000 \]

\[ 50,000 \times 1.05^{10} \approx 81444.73 < 60,000 + 2000 \times 10 = 80,000 \]

Wolfram Alpha gives the result $x = 9.23999$ as the crossover value.

You would need to work at company A for 10 years before your salary exceeds company X’s salary.

iii. How long would you need to stay at company A until your total earnings equaled your total earnings from company X?

**Solution:** Now we want

\[ \sum_{k=0}^{n-1} 1.05^k \geq \sum_{k=0}^{n-1} (1.2 + 0.04k) \]

\[ \sum_{k=0}^{n-1} 1.05^k = \frac{1.05^n - 1}{0.05} \]

and

\[ \sum_{k=0}^{n-1} (1.2 + 0.04k) = 1.2n + 0.02n(n - 1) \]

\[ 1.05^n = 0.06n + 0.001n(n - 1) + 1 \]

Wolfram Alpha gives $x = 16.99$.

You would need to work at company A for 17 years before your total earned salaries would be equal.

**Problem 3** [20pts:10,10 ]: Search Algorithms

A company database has 10,000 customers sorted by last name, 30% of whom are known to be good customers. Under typical usage of this database, 60% of lookups are for the good customers. Two design options are considered to store the data in the database:
1. Put all the names in a single array and use binary search.

2. Put the good customers in one array and the rest of them in a second array. Only if we do not find the query name on a binary search of the first array do we do a binary search of the second array.

Given these options, answer the following.

i. Calculate the expected worst-case performance for each of the two structures above, given typical usage. Which of the two structures is the best option?

ii. Suppose that over time the usage of the database changes, and so a greater and greater fraction of lookups are for good customers. At what point does the answer to part i change?

iii. Under typical usage again, suppose that instead of binary search we had used linear search. What is the expected worst-case performance of each of the two structures and which is the better option? Where is the cross-over this time?

Solution:

Binary Search

Using one array:

\[ \lceil \lg(10000 + 1) \rceil = 14. \]

Using two arrays:

before you lookup the array of bad customer, you look up the array of good customers first

Two arrays, binary search:

\[ 0.6 \times \lceil \lg(3000 + 1) \rceil + 0.4 \times (\lceil \lg(3000 + 1) \rceil + \lceil \lg(7000 + 1) \rceil) = 17.2 \]

In this scenario, option 1 is the best option

Crossover point:

When about 84.6 percent of lookups are performed for good customers, option 2 has a better worst case performance.

\[ x \times \lceil \lg(3000 + 1) \rceil + (1 - x) \times (\lceil \lg(3000 + 1) \rceil + \lceil \lg(7000 + 1) \rceil) = 14 \text{.} \]

Then: \( x = 0.846 \).

Ordered Linear Search

Using one array:

10,000

Using Two arrays:

\[ 0.6 \times 3000 + 0.4 \times (3000 + 7000) = 5800 \]

In this scenario, option 2 is better.

Crossover point:

There is no crossover in this scenario because option 2 cannot perform worse than option 1 under any circumstances.