The University of the South Pacific

School of Computing, Information & Mathematical Sciences

CS214 Design & Analysis of Algorithms

Lab Week 3

This tutorial covers use of appropriate data structure and complexity analysis of a given algorithm.

Question 1. (Use of data structures) – 1 hour

A. SCIMS has allocated $1000 prize money for the first year students who has GPA of more than 4.0. The maximum recipients cannot be more than five. Create a priority queue for students where the priority for dequeue is based on highest GPA stored in the queue.

   You can either read the data from a file or simply hard code it in your program.

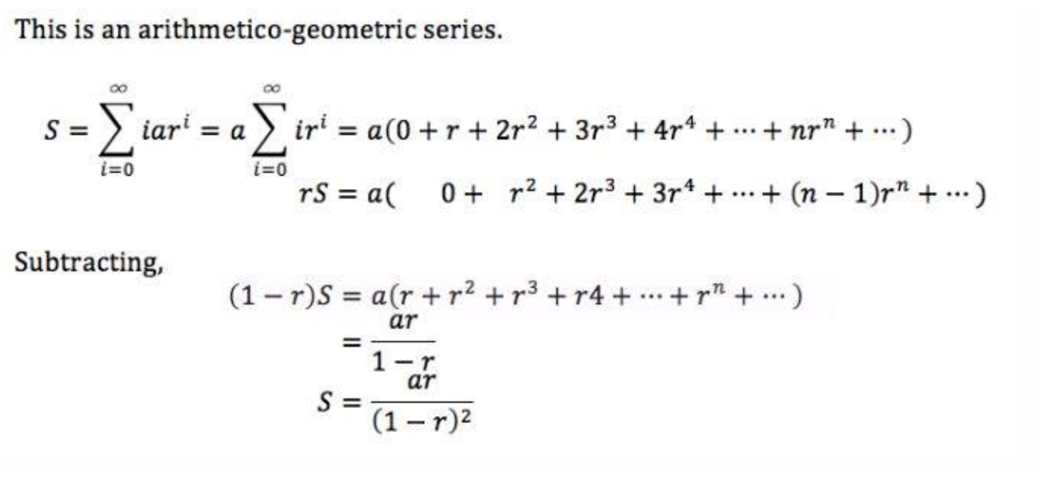
   Test your program to produce the desired output.

B. Is it possible to print student name in alphabetical order without refilling your priority queue? Discuss it with your tutor.

Question 2. (Complexity analysis) – 1 hour

A. Find the best, worst and average time complexity for binary search. Does it have the every-case time complexity? [refer lecture notes for the algorithms]

   Finding average time complexity is a challenging task. You may need the following formula:



This is an arithmetico-geometric series.

$$S = \sum_{i=0}^{\infty} iar^i = a\sum_{i=0}^{\infty} ir^i = a(0 + r + 2r^2 + 3r^3 + 4r^4 + \cdots + nr^n + \cdots)$$

$$rS = a(\quad 0 + \quad r^2 + 2r^3 + 3r^4 + \cdots + (n-1)r^n + \cdots)$$

Subtracting,

$$(1-r)S = a(r + r^2 + r^3 + r4 + \cdots + r^n + \cdots)$$

$$= \frac{ar}{1-r}$$

$$S = \frac{ar}{(1-r)^2}$$

B. Draw the graph of average time complexity for binary search and sequential search in Matlab and discuss your observations with your tutor.

C. If the above two questions have not been covered in the lecture yet, then only find the total passes (operations) required to search an element when it is larger than all the items in an array. Do this experiment for different sizes of array. Draw the total passes (y-axis) against the size of the array (x-axis) for binary search and sequential search in Matlab and discuss your observations with your tutor.

Question 3. (Complexity analysis – Home work) – 1 hour

A. Write the Fibonacci algorithm using iterative and recursive approaches. See the code in Lecture 2.1.
B. Run both the algorithms for *n* = {5, 10, 15, 20, 30, 40, 50, 100, 500, 1000}. You may use even larger values if feasible. You may need to use the function long startTime = System.nanoTime();
C. Now draw the graph of CPU time Vs n for both algorithms in one graph. Do the comparison.

- A. Sharma