CS2710 : PDS Lab Tutorial Questions

Tutorial/Prep 3

Aug. 14, 2024

Information

- This document comprises tutorial questions for CS2700, which includes both conceptual/theory questions (relevant for CS2700) and programming questions (relevant for both CS2700 and CS2710; these questions can also be thought of as preparatory/practice programming questions for CS2710 Lab 3, and so also referred to as Prep 3).
- Try to solve programming problems using array/vector and strings only. (Don't use any other data structure). You can use built-in sort() if asked/required.

Conceptual questions (for CS2700)

1. [PROGRAM FRAGMENT ANALYSIS] Give an analysis of the running time (in terms of Theta (Θ)) for each of the following six program fragments:

1.
```
sum = 0;
for( i = 0; i < n; ++i )
    ++sum;
```

2.
```
sum = 0;
for( i = 0; i < n; ++i )
    for( j = 0; j < n; ++j )
        ++sum;
```

3.
```
sum = 0;
for( i = 0; i < n; ++i )
    for( j = 0; j < n * n; ++j )
        ++sum;
```

4.
```
sum = 0;
for( i = 0; i < n; ++i )
    for( j = 0; j < i; ++j )
        ++sum;
```

5.
```
sum = 0;
for( i = 0; i < n; ++i )
    for( j = 0; j < i; ++j )
        for( k = 0; k < j; ++k )
            ++sum;
```

6.
```
sum = 0;
for( i = 1; i < n; ++i )
    for( j = 1; j < i * i; ++j )
        if( j % i == 0 )
            for( k = 0; k < j; ++k )
                ++sum;
```

2. [ORDERING BY ASYMPTOTIC GROWTH RATES]

a. Rank the following functions by order of growth. That is, find an arrangement $g_1, g_2, \ldots, g_{30}$ of the functions satisfying $g_1 = \Omega(g_2), g_2 = \Omega(g_3), \ldots, g_{29} = \Omega(g_{30})$. Partition your list into equivalence classes such that functions $f(n)$ and $g(n)$ belong to the same class if and only if $f(n) = \Theta(g(n))$.

$\lg(\lg^* n)$	$2^{\lg^* n}$	$(\sqrt{2})^{\lg n}$	n^2	$n!$	$(\lg n)!$
$(3/2)^n$	n^3	$\lg^2 n$	$\lg(n!)$	2^{2^n}	$n^{1/\lg n}$
$\ln \ln n$	$\lg^* n$	$n \cdot 2^n$	$n^{\lg \lg n}$	$\ln n$	$\frac{1}{\sqrt{\lg n}}$
$2^{\lg n}$	$(\lg n)^{\lg n}$	e^n	$4^{\lg n}$	$(n+1)!$	$\sqrt{\lg n}$
$\lg^*(\lg n)$	$2^{\sqrt{2 \lg n}}$	n	2^n	$n \lg n$	$2^{2^{n+1}}$

b. Give an example of a single nonnegative function $f(n)$ such that for all functions $g_i(n)$ in part (a), $f(n)$ is neither $O(g_i(n))$ nor $\Omega(g_i(n))$.

3. [Runtime Analysis] Suppose an algorithm has a runtime of $T(n)$ defined as follows. Simplify $T(n)$ using Big-Oh notation and explain the simplification.

   1. $T(n) = 3T(n/2) + n$

   2. $T(n) = T(n-1) + n$

   3. $T(n) = 2T(n/2) + n$

   4. $T(n) = T(n-1) + \log n$

   5. $T(n) = 4T(n/2) + n^2$

Programming questions (for CS2710 Lab 3 preparation/practice, and for CS2700)

4. [Binary Search vs. Linear Search] Write a program to read a list of sorted integers (of size N) , and perform Binary Search and linear search on it and find running time of each.
   Instruction: Read about measuring running time before Lab 3

   Input Format:
   Take N and target as input in the first line
   Now from Iterate and store values from 1 to N in an array in ascending order

   Output Format:
   Display running time of each Search in a different line

```
Sample Input 0:
1000 675
Sample Output 0:
Target at index: 675
Time taken by Linear Search is : 0.001000 sec
Target at index: 675
Time taken by Binary Search is : 0.000000000 sec
```

```
Sample Input 1:
10000 124
Sample Output 1:
Target at index: 124
Time taken by Linear Search is : 0.002000 sec
Target at index: 124
Time taken by Binary Search is : 0.0010000000 sec
```

   Hint: Read about how to measure running time in C++ **using chrono library here**: `https://www.geeksforgeeks.org/measure-execution-time-function-cpp/` **before coming to Lab3**.

5. [ROWS WITH MAX 1S] You are given a 2D array A of size $N \times N$ consisting of only 1's and 0's, where each row is sorted in non-decreasing order. You need to find and return the index of the first row that has the most number of 1s. If no such row exists, return -1. Note: 0-based indexing is followed.
   Try solving the problem with $O(N^2)$ and $O(N \log N)$ Time complexity.

   Input Format:
   The first line will contain integer N (number of rows/columns in an array)
   Then take array as input

   Output Format:
   Print row containing most number of 1's.

   Sample Input 1
   ```
   4
    0  0  0  1
    0  1  1  1
    1  1  1  1
    0  0  1  1
   ```
   Sample Output 1
   ```
   2
   ```

   Sample Input 2
   ```
   3
   0 0 1 0 0 0 0 0 0
   ```

   Sample Output 2
   ```
   0
   ```

6. [BOOK MANAGEMENT SYSTEM] You are developing a management system for a library to help organize and manage a collection of books. Each book is represented by a set of attributes, and the library staff needs to sort the collection based on different criteria to facilitate efficient searching and organization. You are given a collection of books, each represented by a C++ class. Your task is to implement sorting functionality to organize the books based on different criteria. You will need to sort the books by publication year and by page count using custom comparator functions.

   You should define a **class Book** with the following attributes:

   - title (string): The title of the book.
   - author (string): The author of the book.
   - yearPublished (int): The year the book was published.
   - pageCount (int): The number of pages in the book.

   Input Format:
   You will receive:

   - An integer n $(1 \leq n \leq 10^6)$ representing the number of books.
   - A list of n Books. Each book is provided with its title, author, year of publication, and page count.

   Output Format:
   Your output should consist of two sorted lists of books (with an appropriate message, check sample outputs):

Books Sorted by Publication Year: Print the books sorted in ascending order of yearPublished. If two books have the same publication year, sort them alphabetically by title.
Books Sorted by Page Count: Print the books sorted in ascending order of pageCount. If two books have the same page count, sort them alphabetically by title.

Constraints:

- All book titles and author names are unique.
- The publication year and page count are positive integers.
- Try to solve the problem in $O(n \log n)$.

Sample Input 1:
4
1984 George Orwell 1949 328
To Kill a Mockingbird Harper Lee 1960 281
The Great Gatsby F. Scott Fitzgerald 1925 180
The Catcher in the Rye J.D. Salinger 1951 214

Sample Output 1:
Books sorted by yearPublished:
The Great Gatsby (1925)
1984 (1949)
The Catcher in the Rye (1951)
To Kill a Mockingbird (1960)

Books sorted by pageCount:
The Great Gatsby (180 pages)
The Catcher in the Rye (214 pages)
To Kill a Mockingbird (281 pages)
1984 (328 pages)

Hint: Declare a class Book, make a vector of Book objects, and write custom comparator functions with C++ std::sort like so:

```
vector<Book> books(N);
sort(books.begin(), books.end(), compareByYear);
sort(books.begin(), books.end(), compareByPageCount);
```

To write the compareBy* functions above, do try to **read more about comparator functions** from resources on the internet **before coming to Lab3.**