EXERCISE REPORT

TEAM 15

MATHEMATICAL ANALYSIS OF NON-RECURSIVE ALGORITHM

Lecturer: Son Nguyen Thanh MCS

Class: CS112.N21.KHTN

Member 1: Quan Vo Minh – 21520093

Member 2: Nhat Nguyen Viet – 21520378

Excercise: Team 13's Excercise

Date: 20/03/2023

10. The ***range*** of a finite nonempty set of n real numbers S is defined as the difference between the largest and smallest elements of S. For each representation of S given below, describe in English an algorithm to compute the range. Indicate the time effciency classes of these algorithms using the most appropriate notation (O, Θ or Ω).

a. An unsorted array

- Pseudo-code:

```
$//Input: set\ n\ real\ numbers\ S[0..n-1]$
$smallest = highest = S[0]$
$\textbf{for}\ i \leftarrow 0\ \textbf{to}\ n-1\ \textbf{do}$
    $\textbf{if}\ smallest > S[i]\ \textbf{then}\ smallest \leftarrow S[i]$
    $\textbf{if}\ highest > S[i]\ \textbf{then}\ highest \leftarrow S[i]$
$result = highest - smallest$
```

- Time complexity: $\Theta(n)$

b. A sorted array

- Pseudo-code:

```
$//$Input: $set\ n\ real\ numbers\ sorted\ (increasing)\ S[0..n-1]$
$result = S[n-1] - S[0]$
```

- Time complexity: $\Theta(1)$

c. A sorted singly linked list

Assume that our linked list has a pointer to tail (highest).

- Pseudo-code:

```
$//$Input: $a\ sorted\ singly\ linked\ list\ (increasing)$
$result = tail.value - head.value$
```

- Time complexity: $\Theta(1)$

d. A binary search tree

In this binary search tree, we assume that each node has a value and left child of a node is smaller than that node, right child of a node is larger than that node.

- Pseudo-code:

```
$//Input$: BST with $n$ elements start by $root$

$//is\_right = 0 \to go\ down\ through\ left\ child$
$\boldsymbol{find}(node, is\_right)$
    $direction = is\_right$
    $\boldsymbol{if}\ node.child[direction]\ is\ NULL\ \boldsymbol{then}\ return\ node.value$
    $return\ \boldsymbol{find}(node.child[direction]$

$smallest = \boldsymbol{find}(root, 0)$
$highest = \boldsymbol{find}(root, 1)$
$result = highest - smallest$
```

- Time complexity: $O(n), \Theta(\log_2 n), \Omega(1)$

11. *Lighter or heavier?* You have $n > 2$ identical-looking coins and a two-pan balance scale with no weights. One of the coins is a fake, but you do not know whether it is lighter or heavier than the genuine coins, which all weigh the same. Design a $\Theta(1)$ algorithm to determine whether the fake coin is lighter or heavier than the others.

We split n coins into 3 equal parts A, B, C and the remainder could be 0,1 or 2.

Case 1: Weights of A, B, and C are equal, which means the fake coin is in the remainder. Knowing that all the coins ins A, B, and C are real. We compare the weight of coins in A, B, and C to coins in the remainder.

Case 2: One part is heavier/lighter than the other two, we know that the fake coin is heavier/lighter than the real coin.

In case 1, the algorithm takes two comparisons to determine weight of the fake coins.

In case 2, after two comparisons, we need one more comparison for sth

So complexity is O(n)

12.

```
**ALGORITHM**   $GE(A[0..n-1, 0..n])$
//Input: An $n \times (n+1)$ matrix A $[0..n-1, 0..n]$ of real numbers
$\boldsymbol{for}\ i \leftarrow 0\ \boldsymbol{to}\ n-2\ \boldsymbol{do}$
    $\boldsymbol{for}\ j \leftarrow i+1\ \boldsymbol{to}\ n-1\ \boldsymbol{do}$
        $\boldsymbol{for}\ k \leftarrow i\ \boldsymbol{to}\ n\ \boldsymbol{do}$
            $A[j,k] \leftarrow A[j,k] - A[i,k] \times A[j,i]/A[i,i]$
```

a. Find the time effieciency class of this algorithm. $O(n^3)$

b. What glaring inefficiency does this pseudo-code contain and how it can be eliminated to speed the algorithm up?

The glaring inefficiency this pseudo-code contains is:

In the k-*loop*, expression $A[j, i]/A[i, i]$ is calculated k times. But if we pay attention to given pseudo-code, the expression $A[j, i]/A[i, i]$ remains the same whether k is changed or not.

In order to speed the algorithm up, we can eliminate the repetitive calculation of this expression by pre-calculate it outside the *k-loop,* and multiplying $A[j,k]$ by $A[j,i]/A[i,i]$ inside the *k-loop.*

Pseudo-code:

```
**ALGORITHM**   $GE(A[0..n-1, 0..n])$
//Input: An $n \times (n+1)$ matrix A $[0..n-1, 0..n]$ of real numbers
$\boldsymbol{for}\ i \leftarrow 0\ \boldsymbol{to}\ n-2\ \boldsymbol{do}$
    $\boldsymbol{for}\ j \leftarrow i+1\ \boldsymbol{to}\ n-1\ \boldsymbol{do}$
        $temp = A[j,i]/A[i,i]$
        $\boldsymbol{for}\ k \leftarrow i\ \boldsymbol{to}\ n\ \boldsymbol{do}$
            $A[j,k] \leftarrow A[j,k] - A[i,k] \times temp$
```