Name: David Xie

ID: 40065595

Date: October 20, 2020

COMP 352 – Assignment 2

Question 1

a)

```
Algorithm MyMagic(A, n)
    Input: Array of integer containing n elements
    Output: Possibly modified Array A

    done ← true                  //worst case of constant (1) execution
    j ← 0                        //worst case of constant (1) execution
    while j ≤ n - 2 do           //worst case of n executions
        if A[j] > A[j + 1] then          //worst case of n-1 executions
            swap(A[j], A[j + 1])         //worst case of n-1 executions
            done:= false                 //worst case of n-1 executions
        j ← j + 1                //worst case of n-1 executions
    end while
    j ← n – 1                    //worst case of constant (1) execution
    while j ≥ 1 do               //worst case of n executions
        if A[j] < A[j - 1] then          //worst case of n-1 executions
            swap(A[j – 1], A[j])         //worst case of n-1 executions
            done:= false                 //worst case of n-1 executions
        j ← j – 1                //worst case of n-1 executions
    end while
    if ¬ done            //worst case of constant (1) execution
        MyMagic (A, n)           //worst case of n/2+1 calls
    else
        return A
```

$$f(n) = (10n - 4) \times \left(\frac{n}{2} + 1\right) = 5n^2 + 8n - 4$$

$f(n)$ is $O(g(n))$ if there exists a $c > 0$ & $n_0 \geq 1$ such that $f(n) \leq c \times g(n)$ for $n \geq n_0$

$5n^2 + 8n - 4 \leq 100n^2$ for $c = 100$ & $n_0 = 1$, therefore, time complexity $O(n^2)$

$f(n)$ is $\Omega(g(n))$ if there exists a $c > 0$ & $n_0 \geq 1$ such that $f(n) \geq c \times g(n)$ for $n \geq n_0$

$5n^2 + 8n - 4 \geq n^2$ for $c = 1$ & $n_0 = 1$, therefore, time complexity $\Omega(n^2)$

b)

A = (9, 3, 11, 5, 2)

Call MyMagic(A, 5)

Loop 1:

- (3, 9, 11, 5, 2)
- (3, 9, 5, 11, 2)
- (3, 9, 5, 2, 11)

Loop 2:

- (3, 9, 2, 5, 11)
- (3, 2, 9, 5, 11)
- (2, 3, 9, 5, 11)

Recursion

- Loop 1
  - (2, 3, 5, 9, 11)
  - done is false
- Loop 2
  - (2, 3, 5, 9, 11)
- Recursion because done is false
  - Loop 1: same thing
  - Loop 2: same thing

Resulting A = (2, 3, 5, 9, 11)

c)

MyMagic sorts an array of n elements and returns the sorted array. It does so by pushing larger elements towards the end of the array in the first loop, Then, it pushes smaller elements towards the start of the array in the second loop. Next, it does recursion until the array is sorted.

d)

Yes, since MyMagic is tail recursive, it can be reimplemented nonrecursively, which can save on resources.

e)

MyMagic is tail recursive since it makes its recursive call as the final step.

Question 2

$f(n)$ is $O(g(n))$ if there is $c > 0$ & $n_0 \geq 1$ such that $f(n) \leq c * g(n)$ for $n \geq n_0$

$f(n)$ is $\Omega(g(n))$ if there is $c > 0$ & $n_0 \geq 1$ such that $f(n) \geq c * g(n)$ for $n \geq n_0$

$f(n)$ is $\Theta(g(n))$ if there is $c' > 0$, $c'' > 0$ & $n_0 \geq 1$ such that $c' * g(n) \leq f(n) \leq c'' * g(n)$ for $n \geq n_0$

i) $f(n) = 10^5 n * \log(n) + n^3$, $g(n) = \log(n)$

$10^5 n * \log(n) + n^3 \nleq c * \log(n)$ because $f(n)$ grows much faster than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) > c * g(n)$

$10^5 n * \log(n) + n^3 \geq \log(n)$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows much faster than $g(n)$ and it is valid at base, this is valid for $n \geq n_0$. This shows that $f(n)$ is $\Omega(g(n))$

ii) $f(n) = 2\log(n^2)$, $g(n) = (\log(n))^2$

$2\log(n^2) \leq 4 * (\log(n))^2$ with $c = 4$ and $n_0 = 1$. Since $f(n)$ grows slower than $g(n)$ and it is valid at base, this is valid for $n \geq n_0$. This shows that $f(n)$ is $O(g(n))$

$2\log(n^2) \ngeq c * (\log(n))^2$ because $f(n)$ grows slower than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) < g(n)$.

iii) $f(n) = \log(n^2) + n^3$, $g(n) = \log(n) + 5$

$\log(n^2) + n^3 \nleq c * \log(n) + 5$ because $f(n)$ grows much faster than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) > c * g(n)$

$\log(n^2) + n^3 \geq \log(n) + 5$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows much faster than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $\Omega(g(n))$

iv) $f(n) = n\sqrt{n} + \log(n)$, $g(n) = \log(n^2)$

$n\sqrt{n} + \log(n) \nleq c * \log(n^2)$ because $f(n)$ grows much faster than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) > c * g(n)$

$n\sqrt{n} + \log(n) \geq \log(n^2)$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows much faster than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $\Omega(g(n))$

v) $f(n) = 2^n + 10^n$, $g(n) = 10n^2$

$2^n + 10^n \nleq c * 10n^2$ because $f(n)$ grows much faster than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) > c * g(n)$

$2^n + 10^n \geq 10n^2$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows much faster than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $\Omega(g(n))$

vi) $f(n) = n!, g(n) = n^n$

$n! \leq n^n$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows slower than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $O(g(n))$

$n! \ngeq c * n^n$ because $f(n)$ grows slower than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) < g(n)$.

vii) $f(n) = \log^2 n, g(n) = \log(n)$

$\log^2 n \nleq c * \log(n)$ because $f(n)$ grows much faster than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) > c * g(n)$

$\log^2 n \geq \log(n)$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows much faster than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $\Omega(g(n))$

viii) $f(n) = n, g(n) = \log^2 n$

$n \nleq c * \log^2 n$ because $f(n)$ grows much faster than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) > c * g(n)$

$n \geq \log^2 n$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows much faster than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $\Omega(g(n))$

ix) $f(n) = \sqrt{n}, g(n) = \log(n)$

$\sqrt{n} \nleq c * \log(n)$ because $f(n)$ grows much faster than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) > c * g(n)$

$\sqrt{n} \geq \log(n)$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows much faster than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $\Omega(g(n))$

x) $f(n) = 2^n, g(n) = 3^n$

$2^n \leq 3^n$ with $c = 1$ and $n_0 = 1$. Since $f(n)$ grows slower than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $O(g(n))$

$2^n \ngeq c * 3^n$ because $f(n)$ grows slower than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) < g(n)$.

xi) $f(n) = 2^n, g(n) = n^n$

$2^n \leq 2 * n^n$ with $c = 2$ and $n_0 = 1$. Since $f(n)$ grows slower than $g(n)$, this is valid for $n \geq n_0$. This shows that $f(n)$ is $O(g(n))$

$2^n \ngeq c * n^n$ because $f(n)$ grows slower than $g(n)$. Regardless of c, there will eventually be an n for which $f(n) < g(n)$.