Name: ALFREDO PEREZ____________________________________
COSC 40403 - Analysis of Algorithms: Fall 2020: Homework 2
Due: 23:30 on 8/31

Question	Points	Score
1	5	
2	10	
3	10	
4	10	
5	10	
6	5	
7	10	
Total:	60	

1. (5 points) Write out the definitions of $O(f(n))$, $\Omega(f(n))$, and $\Theta(f(n))$ in equation format using LaTeX.

Solution: $T(n)$ is $O(f(n)) = \exists$ constants > 0 and $n_0 \geq 0 \mid \forall\, n \geq n_0$ we have $T(n) \leq c$ $f(n)$
$T(n)$ is $\Omega(f(n)) = \exists$ constants > 0 and $n_0 \geq 0 \mid \forall\, n \geq n_0$ $T(n) \geq c\ f(n)$.

$T(n)$ is $\Theta(f(n))\ if$ T(n) = O(f(n))= $\Omega(f(n))$

2. (10 points) Show directly that $f(n) = n^2 + 3n^3 \in \Theta(n^3)$. That is, use the definitions of O and Ω to show that $f(n)$ is both $O(n^3)$ and $\Omega(n^3)$.

Solution:
Big O is defined as $O(g(n)) = f(n)$: There exist positive constants c and n0 such that $0 < f(n) <$ c $g(n)$ for n > n0. $g(n)$ is tight upper bound of $f(n)$.
Omega-Ω is defined as $\Omega(g(n)) = f(n)$: There exist positive constants c and n0 such that $0 <= cg(n) <= f(n)$ for $n >= n0$).

$g(n)$ is asymptotic tight lower bound for $f(n)$

Given $f(n) = n^2 + 3n^3$
Therefore $n^2 <= n^3$
$n^2 + 3n^3 <= 4n^3, n > 1$
Therefore $n^2 + 3n^3 = O(n^3)$ with c = 4, n0 = 1
Big-O of $f(n) = O(n^3)$

Given $f(n) = n^2 + 3n^3$
$0 <= cn^2 <= 3n^3$
$cn^2 <= 3n^3 => c = 1 and n0 = 1.$
Theta , Ω of $f(n) = \Omega(n^3)$

3. (10 points) Group the following functions by complexity category.

$n \lg n$	$(\lg n)^2$	$5n^2 + 7n$	$n^{5/2}$
$n!$	$2^{n!}$	4^n	n^n
$n^n + \ln n$	$5^{\lg n}$	$\lg(n!)$	$\lg(n)!$
$\sqrt{n}$	e^n	$8n + 12$	$10^n + n^{20}$

Solution:
n^n and $(n^n + lnn)$
$n!$
$10^n + n^20$
4^n
e^n
$(\lg n)^2$
$n^{5/2}$
$5^{\lg n}$
$5n^2 + 7n$
$nlogn$ and $lg(n!)$
$8n + 12$
$\sqrt{n}$
$(\lg n)^2$

4. (10 points total) Consider the following algorithm:

```
n = int(input("Enter a value for n: "))
i = 2
while i <= n:
    j = 0
    while j <= n:
        print(i, j)
        j = j + n // 4
   i = i + 1
```

(a) (5 points) What is the output when $n = 4$, $n = 16$, $n = 32$? Do not show all of the output. Just a small subset.

(b) (5 points) What is the time complexity $T(n)$. You may assume that n is divisible by 4.

Solution: When n = 4:
2 0
2 1
2 2
2 3
2 4
When n = 16:
2 0
2 4
2 8
2 12
2 16
When n = 32:
2 0
2 8
2 16
2 24
2 32

The time complexity of this algorithm is O(n).

5. (10 points total) Consider the following algorithm (written in a C-like language).

```
int add_them(int n, int A[]){  // Assume this array is 1-based
    index i, j, k

    j = 0
    for(i=1; i<=n; i++)
        j = j + A[i]
     k = 1
    for(i=1; i<=n; i++)
        k = k + k
    return(j+k)
}
```

(a) (3 points) If $n = 5$ and the array A contains $\langle 2, 5, 3, 7, 8 \rangle$, what is the output?

(b) (4 points) What is the time complexity $T(n)$ of the algorithm?

(c) (3 points) Try to improve the efficiency of the algorithm.

Solution:
The function returns garbage values since we don't know A[5] it returns previously stored value in that address.
The time complexity is O(n).

```
int add_them(int n, int A[]){  // Assume this array is 1-based
     index i, j, k

     j = 0
     k=1
     for(i=1; i<=n; i++){
          k=k+k
          j = j + A[i]
     }
     return(j+k)
}
```

6. (5 points) Exercise 2 on page 67.

Solution: A) n^2 = 6,000,000 operations
B) n^3 = 33,019 Operations
C) $100n^2$ = 600,000 Operations
D) $nlogn$ = 1.29 * $10^($12) operations
E) 2^n = 45 operations
F) $2^(2^n)$ = 5 operations

7. (10 points total) Justify the correctness of the following statements assuming that $f(n)$ and $g(n)$ are asymptotically positive functions.

   (a) (5 points) $f(n) + g(n) \in O(\max(f(n), g(n)))$

   (b) (5 points) $f^2(n) \in \Omega(f(n))$

Solution: Based on the definitions of Big-Oh notation we know that any function g is big-oh of another function f if g is asymptotically greater than f.

With these two functions we can see that f(n) and g(n) will be greater than each other. if f(n) is greater than f(n) is required right hand side function else g(n).

Since f(n) or g(n) will have to be our right side function, we know that: $f(n)+g(n)\epsilon O(f(n), g(n))$

Based on the definition of big-omega, if any function f is omega of any function g then f must be asymptotically greater than g.

if $(f(n))2\epsilon\Omega(f(n))$ is true then left side must be greater than right side.

now, suppose $f(n) = 1/n$ then clearly $1/n^2 <= 1/n$ so,right side function is greater than left side function.
Hence, this is false.