COMP 4030/6030 - Spring 2017
Exam 1

Name: Mazen Istanbouli

Note: Do not write on the backs; only front pages are digitized and graded.

1. (*20 points*) Answer with True or False.

0

- $3n^2 + n^3 \in \Omega(n^3)$ False
- $3n^2 + n^3 \in O(n^4)$ False

2. (*10 points*) Use the definition of O to show that $5n^2 + 10n \in O(n^2)$.

8

$5n^2 + 10n^2 = 15n^2$ when $n = 1$, $c = 15$

3. (*20 points*) Use the definition of Θ to show that $3n^3 + 5n - 5 \in \Theta(n^3)$.

5

$3n^3 + 5n^3 - 5n^3 = 3n^3$ when $n = 1$, $c = 3$

4. (*10 points*) Use Θ to specify the running time of the following procedure in terms of n.

10

```
sum = 0
for i in range(n):        Θ(n)
    for j in range(5):        Θ(1)
        sum = sum + i + j
```

Running time is $\Theta(n)$

5. (*10 points*) Use Θ to specify the running time of the following procedure in terms of n.

3

```
sum = 0
for i in range(n):        Θ(n)
    j = n
    while j > 1:
        sum = sum + i*j
        j = j / 2         Θ(log2 n)
```

Running time is ~~$\Theta(\log_2 n)$~~ $\Theta(n \log_2 n)$ ✗

6. (*10 points*) Is the running time of the procedure in Question 4 in $\Omega(n)$? Explain briefly.

2

No, because the lower bound would be $\Omega(1)$ since the best case is where the ~~loop~~ function is $n = 1$.

7. (*10 points*) What is the space complexity of the procedure in Question 4? $S(n)$

1

8. (*10 points*) Explain what the following function does. The input is a list of numbers.

```
def foo(L):
    if len(L) == 0:
        return 1
    return L[0] * foo(L[1:])
```

3

The function foo defines a list of numbers that when the length of the list is "0", the function returns 1. If the length of the list is greater than 0, then it multiplies the first number in the list to the first number removed from the list. ✗

9. (*10 points*) Use mathematical induction to explain that the following function correctly adds up all numbers that are divisible by 5 in the input list.

```
def bar(L):
    if len(L) == 0:
        return 0
    if L[0] % 5 == 0:
        return L[0] + bar(L[1:])
    return bar(L[1:])
```

Base: Given L_k, with k as the length of list L, if $k = 0$, then $len(L_k) = 0$

Induction Hypothesis:

Given L_{k+1}, then

$bar(L_{k+1}) = L_{k+1}[0] + bar(L_{k+1}[1:])$

10. (*10 points*) The input of the following function is a binary tree T. Assume that such a tree T has 3 attributes: T.left, T.right (both of which are also binary trees), and T.color (which is a string of either "red", "green", or "blue"). If T is empty, the function *is_empty*(T) returns True (if not, it returns False). Complete defining the following Python function so that it correctly counts the number of red nodes in the input binary tree T.

```
def count_red_nodes(T):
    # fill in your codes below to count the number of red nodes in the binary tree T
    if T == 0:
        is_empty(T) = True
    return 0
    if T.color = "red":
```