COMP 4030/6030 - Spring 2017
Exam 1

Name: Rick DeSaussure

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

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

20

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

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

10

$5n^2+10 \leq c \cdot n^2$ / $5n^2+10 \leq (5n^2+10n^2) = 12n^2$ when $c=12$ while $n>1$

15

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

20

$3n^3+5n-5 \leq c \cdot n^3$ / $3n^3+5n-5 \leq 3n^3+5n^3-5n^3 = 3n^3$

$c=3$ while $n>1$

$3n^3+5n-5 \geq c \cdot n^3$ / $3n^3+5n-5 \geq 1 \cdot n^3$

$c=1$ while $n>1$

since it is both in $O(n^3)$ and $\Omega(n^3)$, it is also in $\Theta(n^3)$

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

4

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

$T(n) = n \cdot n \in \Theta(n^2)$

X

5. (*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): n
    j = n
    while j > 1:
        sum = sum + i*j
        j = j / 2  2log(n)
```

$T(n) = n \cdot 2\log n \in \Theta(n \log n)$

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

5

If the running time is specified in terms of $\Theta(n^2)$, it is, by extension, also in $O(n^2)$ and $\Omega(n^2)$. Thus, it is not in $\Omega(n)$

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

8

$S(n)=1$ $\Theta(1)$

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:])
```

10

foo(L) returns 1 if len(L)==0, but recursively returns L[0] · L[1:].

foo(L) returns the product of all values in L if len(L)>0, else returns 1

- smallest input - "solve" for k
- assume (k-1)

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[0:])
    return bar(L[0:])
```

If L is empty [len(L)==0], the list returns 0 since no numbers can be added up

If len(L)==(k-1), L[0]%5==0 (is evenly divisible by 5), bar(L) returns L[0] and recursively checks and adds L[0] and any other L member that fulfills the condition of "L[0]%5==0"

else, bar(L) recursively checks L[1:] for c)

Assuming it works for (k-1), if len(L)==k, it will add all values that are divisible by 5, as explained above.

7

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
    totRed=0
    if is_empty(T) True:
        return 0
    if is_empty(T.left)==false:
        if T.color =="Red":
            totRed=totRed+1
        return totRed+count_red_nodes(T.left)
    else if is_empty(T.right)==False:
        if T.color =="red":
            totRed=totRed+1
        return totRed+count_red_nodes(T.right)
    return totRed
```

recursion

2