CS471 Complexity and Computability
Department of Computer Science
School of Science and Technology

Quiz 1

1. (15 *pts*) What is the asymptotic running time of Counting sort? Does the running time depend on other factors other than input size?

```
COUNTING-SORT(A, B, k)
let C[0..k] be a new array
for i = 0 to k                                   } Θ(k)
    C[i] = 0
for j = 1 to A.length                            } Θ(n)
    C[A[j]] = C[A[j]] + 1
// C[i] now contains the number of elements equal to i.
for i = 1 to k                                   } Θ(k)
    C[i] = C[i] + C[i - 1]
// C[i] now contains the number of elements less than or equal
for j = A.length downto 1                        } Θ(n)
    B[C[A[j]]] = A[j]
    C[A[j]] = C[A[j]] - 1
```

$T(n) = \Theta(n+k)$

if $k = O(k) \Rightarrow T(n) = \Theta(n)$

Yes, it also depends on k which is the range (max) of the input elements.

2. (25 *pts*) Prove or disprove each of the following.

(a) (15 *pts*) $f(n) + o(f(n)) = \theta(f(n))$

1. $g(n) = o(f(n)) \Rightarrow g(n) < c \cdot f(n)$ for all $c>0$, there exists n_0

2. $c_1 \cdot f(n) \leq f(n) + g(n) \leq c_2 \cdot f(n)$ // divide by $f(n)$

$c_1 \leq 1 + \frac{g(n)}{f(n)} \leq c_2$ // as $\frac{g(n)}{f(n)} < c$ for all $c>0$

$c_1 = 1$ $c_2 = 2$. ✓

(b) (10 *pts*) $\frac{n}{100} + 2 = \Omega(n)$

$\frac{n}{100} + 2 \geq c \cdot n \Rightarrow (n + 200 \geq)$ // divide by n

$\frac{1}{100} + \frac{2}{n} \geq c \Rightarrow c = \frac{1}{100}$ $n_0 = 1 \leq n$ ✓

3. (30 *pts*) Solve the recurrence $T(n) = 3T(\sqrt{n}) + 1$ by making a change of variables.

$m = \lg n \Rightarrow n = 2^m$

$T(n) = 3T(\sqrt{n}) + 1$

$T(2^m) = 3T(2^{m/2}) + 1$

Let $S(m) = T(2^m)$

$S(m) = 3 \cdot S(m/2) + 1$

by master method $a = 3$, $b = 2$

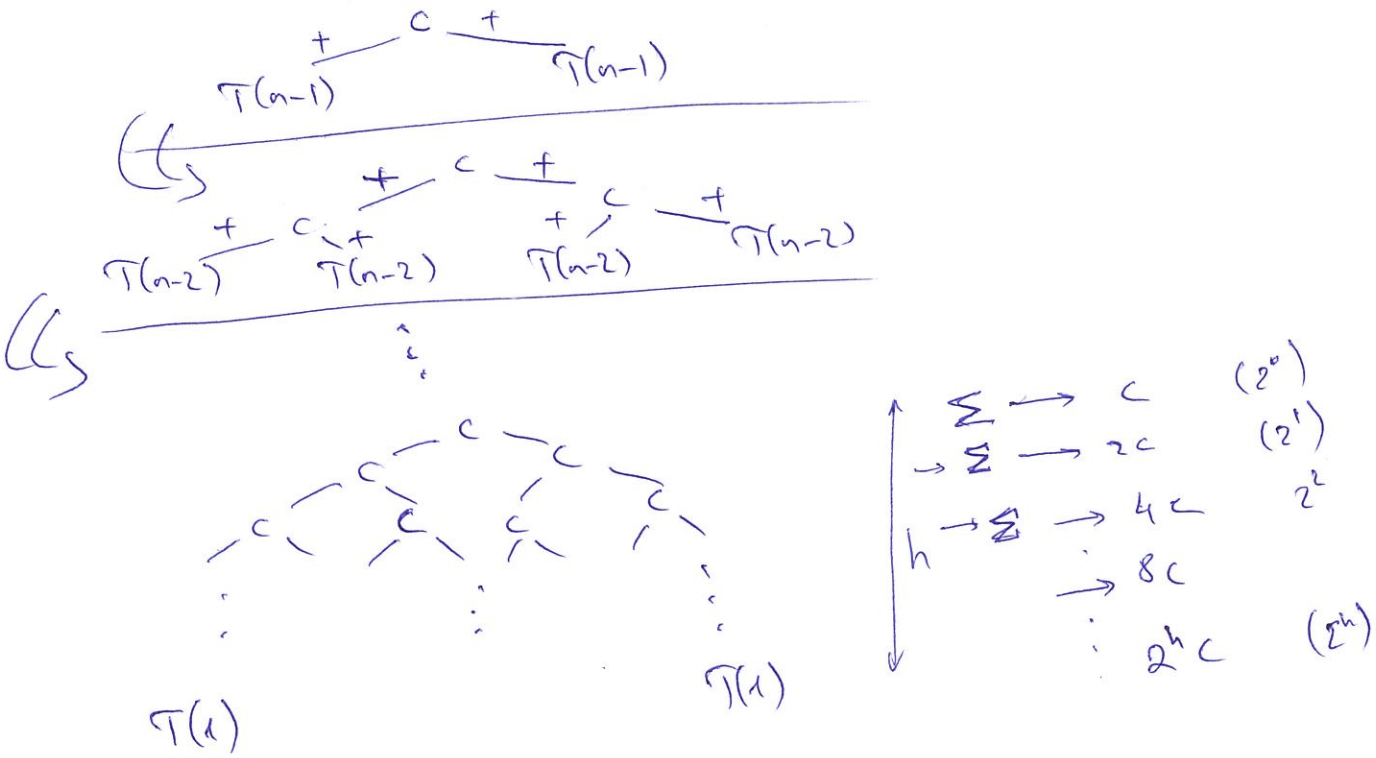
$m^{\log_b a} = m^{\lg_2 3} > 1 \Rightarrow 1^{st}$ case $T(n) = \Theta(m^{\lg_2 3}) = \Theta(\lg n^{\lg_2 3})$

4. (30 *pts*) Write a recurrence for the running time T(n) of `f(n)`, and solve that recurrence.

```
f(n)
    if n == 1
        return 1
    else
        return f(n-1) + f(n-1)
```

i) $T(n) = T(n-1) + T(n-1) + \Theta(1)$

ii) $T(n) = T(n-1) + T(n-1) + c$



$T(1) = T(n-h) \Rightarrow 1 = n-h \Rightarrow h = n-1$

$$T(n) = \sum_{i=0}^{h} 2^i \cdot c = \sum_{i=0}^{n-1} 2^i \cdot c = \frac{2^{n-1+1}}{2-1} \cdot c = c \cdot 2^n = O(2^n)$$