Onur Yaman
2007961

Ceng 310 - Homework 3

Task-1

R-3.2

Firstly, we need to find an intersection point.
(For example in base 2 in logaritmic function)

$$8\cancel{n}\log_2 n = 2n^{\cancel{2}}$$

$$4\log_2 n = n$$

For all $n \geq n_0 = 16$, $8n\log_2 n \leq 2n^2$

So, $A = O(B)$ implying that A is better in time consuming.

R-3.9

$d(n) = O(f(n)) \Rightarrow$ for all $n \geq n_0$ $d(n) \leq f(n)$

By multiplying both sides by a constant $a > 0$,

we get $a \cdot d(n) \leq a \cdot f(n)$

By ignoring a, we have $O(a \cdot f(n)) = O(f(n))$. Then,

$a \cdot d(n) = O(a \cdot f(n)) = O(f(n)) \Rightarrow a \cdot d(n) = O(f(n))$

R-3.17

Take $n_0 = 1$ and $k = 33$. Then, for all $n \geq n_0 = 1$
$(n+1)^5 \leq 33 \cdot n^5$. This means $(n+1)^5 = O(n^5)$

Also note that we can ignore all terms except n^5 in
$(n+1)^5 = n^5 + 5n^4 + 10n^3 + 10n^2 + 5n + 1$. This means $O(n^5) = O((n+1)^5)$

R-3.18

Take $k = 3$. For all $n \geq n_0$

since $2^{n+1} = 2 \cdot 2^n \leq 3 \cdot 2^n$, we have $2^{n+1} = O(2^n)$

R-3.20

To find sufficiently large $n \geq n_0$ and constant k

$$n^2 = n \cdot \log n$$
$$n = \log n$$

For example in base 2 :

$$n = k \cdot \log_2 n$$

Taking $n_0 = 4$ and $k = 2$, we have $4 = 2 \cdot \log_2 4$

For all $n \geq n_0 = 2$, $n \geq 2 \log_2 n$ $\left(n = 8 \Rightarrow 8 > 2 \log_2 8 \right)$

$n^2 = \Omega(n \log n)$

Task-2

R-3.15

($\Rightarrow$) Assume that $f(n) = O(g(n))$. This means that

For all $n \geq n_0$ $f(n) \leq k \cdot g(n)$ for some $k > 0$

For all $n \geq n_0$ $g(n) \geq \frac{1}{k} \cdot f(n)$ note that $\frac{1}{k} > 0$

This means $g(n) = \Omega(f(n))$

($\Leftarrow$) Assume that $g(n) = \Omega(f(n))$. This means that

for all $n \geq n_0$ $g(n) \geq k \cdot f(n)$. for some $k > 0$

For all $n \geq n_0$ $f(n) \leq \frac{1}{k} \cdot g(n)$ note that $\frac{1}{k} > 0$

This means $f(n) = O(g(n))$

Task-3

R-3.25

```
def example3(S):                    cost:
    n = len(S)                      c1
    total = 0                       c2
    for j in range(n):              c3
        for k in range(1+j)         c4 · n
            total += S[k]           c5 · n(n+1)/2
    return total                    c6
```

Let the time consuming function in terms of n be f(n)

$f(n) = c_5 \frac{n^2}{2} + (2c_4 + c_5)\frac{n}{2} + c_1 + c_2 + c_3$

By ignoring all terms except n^2

$O(f(n)) = O(\frac{c_5}{2} n^2) = O(n^2)$ note that $\frac{c_5}{2}$ is also ignored

$O(f(n)) = O(n^2)$

R-3.27

```
def example5(A, B):                 cost:
    n = len(A)                      c1
    count = 0                       c2
    for i in range(n):              c3
        total = 0                   c4 n
        for j in range(n):          c5 n
            for k in range(1+j)     c6 n^2
                total += k          c7 n (n(n+1)/2)
        if B[i] == total:           c8 n
            count += 1
    return count
```

Time consuming in terms of n:

$$f(n) = \frac{c_7}{2}(n^3+n^2) + c_6 n^2 + (c_4+c_5+c_8)n + c_1+c_2+c_3$$

Ignoring all terms except n^3 and ignoring $\frac{c_7}{2}$,

we have $O(f(n)) = O(n^3)$

Time consuming is associated with n^3. $\Rightarrow \boxed{O(n^3)}$

Task-4

R-3.33

Claim: $O(n\log n)$ is always faster than $O(n^2)$ for sufficently large $n \geq n_0 = 100$.

To check this.

When $n < 100$, claim is false:

Assume that $O(n^2) = k n^2$ for some $k > 0$

Take $n = 10^2$

$$10^2 \log 10^2 = k\, 10^4 \Rightarrow k = \frac{1}{50}$$

When Bob's time method is $\frac{1}{50}n^2$

$$10 \log 10 > \frac{1}{50} 10 \quad (n = 10)$$

Bob's method runs faster, so the claim is false.

When $n > 100$, claim is true:

Take $n = 10^3$

$$10^3 \log 10^3 < \frac{1}{50} 10^6$$

Al's method runs faster, so the claim is true.