Data Structure in Mathematics

Assignment 2

R-3.8 Order the following functions by asymptotic growth rate.

$$\begin{array}{ccc} 4n\log n + 2n & 2^{10} & 2^{\log n} \\ 3n + 100\log n & 4n & 2^n \\ n^2 + 10n & n^3 & n\log n \end{array}$$

Answer :

$$\text{constant}: \ 2^{10}$$

$$\text{n}: \ 2^{\log n} = n$$

$$\text{n}: \ 3n + 100\log n \quad (3 > 1)$$

$$\text{n}: \ 4n \quad (4 > 3)$$

$$n\log n: \ n\log n$$

$$n\log n: \ 4n\log n + 2n$$

$$n^2: \ n^2 + 10n$$

$$n^3: \ n^3$$

$$2^n: \ 2^n$$

R-3.18 Show that 2^{n+1} is $O(2^n)$.

Answer : From the power rules, $2^{n+1} = 2 \cdot 2^n$. Notice that, there is a constant $C = 2$ such that $2^{n+1} \leq C \cdot f(n)$, where $f(n) = 2^n$. Thus, 2^{n+1} is $O(2^n)$

R-3.20 Show that n^2 is $\Omega(n\log n)$.

Answer : We know that $f(n)$ is $O(g(n))$ if and only if $g(n)$ is $\Omega(f(n))$. Therefore, if $n\log n$ is $O(n^2)$. Then, n^2 is in fact $\Omega(n\log n)$. $n\log n$ is $O(n^2)$.

R-3.24 Give a big-Oh characterization, in terms of n, of the running time of the example2 function shown in Code Fragment 3.10.

definition :

```
def example2(S):
        n = len(S)
        total = 0
        for j in range(0, n, 2):
                total += S[j]
        return total
```

Answer : To access the length of S, we need $S \geq 1$. The loop runs $n/2$ times which is still $O(n)$. Each loop requires at least $O(1)$ times. Therefore, the return takes $O(1)$. Thus, $O(1 + n + 1)$ is $O(n)$.

R-3.26 Give a big-Oh characterization, in terms of n, of the running time of the example4 function shown in Code Fragment 3.10.

definition :

```
def example4(S):
        n = len(S)
        prefix = 0
        total = 0
        for j in range(n):
                prefix += S[j]
                total += prefix
        return total
```

Answer : Initial state of $O(n)$ is $O(1)$. The loop is called n times. Each time, the loop uses $O(1)$. Therefore, the total is $O(n)$ and the return is $O(1)$. The total is $O(1 + n + 1) = O(n)$