CP2410 Practical 02 - Analysis of Algorithms

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

$4n \log n + 2n$	2^{10}	$2 \log n$
$3n + 100 \log n$	$4n$	2^n
$n^2 + 10n$	n^3	$n \log n$

2. (R-3.2) The number of operations executed by algorithms A and B is 8 n log n and $2n^2$, respectively. Determine n_0 such that A is better than B for $n \geq n_0$.

3. (R-3.9) Show that if d(n) is O(f(n)), then a*d(n) is O(f(n)), for any constant a > 0.

4. See the following functions from **ch03/exercises.py** in the sample code. For each of example1, to example5, determine the running time, in big Oh notation, of the function in terms of **n**.

```
def example1(S):
    """Return the sum of the elements in sequence S."""
    n = len(S)
    total = 0
    for j in range(n):  # loop from 0 to n-1
        total += S[j]
    return total


def example2(S):
    """Return the sum of the elements with even index in sequence S."""
    n = len(S)
    total = 0
    for j in range(0, n, 2):  # note the increment of 2
        total += S[j]
    return total


def example3(S):
    """Return the sum of the prefix sums of sequence S."""
    n = len(S)
    total = 0
    for j in range(n):  # loop from 0 to n-1
        for k in range(1 + j):  # loop from 0 to j
            total += S[k]
    return total


def example4(S):
    """Return the sum of the prefix sums of sequence S."""
    n = len(S)
    prefix = 0
    total = 0
    for j in range(n):
        prefix += S[j]
        total += prefix
    return total
```

```
def example5(A, B):  # assume that A and B have equal length
    """Return the number of elements in B equal to the sum of prefix sums in A."""
    n = len(A)
    count = 0
    for i in range(n):  # loop from 0 to n-1
        total = 0
        for j in range(n):  # loop from 0 to n-1
            for k in range(1 + j):  # loop from 0 to j
                total += A[k]
        if B[i] == total:
            count += 1
    return count
```