1 Asymptotic Order Notation

Let $f(x)$ and $g(x)$ be functions of a positive x.

1.1 Big O Notation

Big O notation describes an upper bound on the growth rate of a function.

$$f(x) = O(g(x))$$

when there is a positive constant c such that,

$$f(x) \leq cg(x)$$

for all $x \geq n$.

1.2 Big Omega Notation

Big Omega notation describes a lower bound on the growth rate of a function.

$$f(x) = \Omega(g(x))$$

when there exist positive constants c, n such that for all $x \geq n$ we have,

$$f(x) \geq cg(x)$$

1.3 Big Theta Notation

Big Theta notation describes a tight bound on the growth rate of a function.

$$\Omega(g(x)) = f(g(x)) = O(g(x))$$

Then such a tight bound is stated as f is big-Θ of g.

$$f(x) = \Theta(g(x))$$

2 Question 1: Conditional Work Subroutine

Consider the following Java subroutine:

```
public static void conditionalWork(int n) {
    for (int i = 0; i < n; i++) {
        if (Math.random() < 0.5) {
            taskA()
        } else {
            taskB()
        }
    }
}
```

2.1 Part 1

If on a certain machine the function **taskA()** takes 2 seconds on each call in the loop and the function **taskB()** takes 4 seconds then for $n = 10$, what is the total number of expected seconds taken by the subroutine public static void conditionalWork(int n)?

The total number of expected seconds is 30 seconds.

$$a(x) = \frac{x}{2}$$

$$b(x) = 4(\frac{x}{2})$$

2.2 Part 2

The subroutine conditionalWork(int n) is ran on a much faster machine reducing the time taken by **taskA()** on each call down to $\frac{1}{2}$ of a second and **taskB()** now takes $\frac{1}{5}$ of a second. For $n = 10$, what is the new total number of seconds expected by the subroutine conditionalWork(int n)?

$$A = \frac{1}{2}$$

$$B = \frac{1}{5}$$

$$result(10) = 25 \cdot \frac{1}{2} + 5 \cdot \frac{1}{5} = 5$$