

---

## DSC 40B - Discussion 01

---

### Problem 1. (Lab Problems)

What is the time complexity of the following functions? State your answer using  $\Theta$  notation.

a) 

```
def foo(n):
    for i in range(n**2 - 2*n + 100):
        j = 0
        while j < n:
            j += 1
```

**Solution:**  $\Theta(n^3)$

b) 

```
def foo(n):
    while n > 1:
        n /= 10
        print(n)
```

**Solution:**  $\Theta(\log n)$

c) 

```
def foo(n):
    for i in range(n):
        for j in range(i**2): # <-- notice the bound!
            print(i + j)
```

**Solution:**  $\Theta(n^3)$

### Problem 2.

Consider the code below:

```
def foo(n):
    i = 1
    while i * i < n:
        i += 1
    return i
```

a) What does `foo(n)` compute, roughly speaking?

**Solution:** It computes, approximately,  $\sqrt{n}$ . Of course, `foo` always returns an integer, so the result of the function is usually not exactly correct.

More precisely, `foo` returns the largest integer greater than or equal to  $\sqrt{n}$ . For example,  $\sqrt{5}$  is between 2 and 3; `foo(5)` returns 3.

b) What is the asymptotic time complexity of `foo`?

**Solution:**  $\Theta(\sqrt{n})$

**Problem 3.**

Let  $f(n) = \sum_{p=0}^n 3^p$ . What is  $f$  in  $\Theta$  notation?

**Solution:**

General form of a geometric sum  $\sum_{p=0}^n x^p = \frac{1 - x^{n+1}}{1 - x}$ .

Substituting our equation yields  $\sum_{p=0}^n 3^p = \frac{1 - 3^{n+1}}{1 - 3}$ .

Therefore,  $f(n) = \Theta(3^n)$  after throwing out the constants.