CS 61B Spring 2020 Exam Prep 8: More Asymptotics

1 Flip Flop

For each problem, give the best and worst-case runtimes in $\Theta(.)$ notation as a function of `n`. Your answer should be simple with no unnecessary leading constants or summations.

```
public static void flip(int n) {
    if (n <= 100) {
        return;
    }
    for (int i = 1; i < n; i++) {
        // Assume g(i,n) will be equal to i for at least one i
        if (g(i, n) == i) {
            flop(i, n);
            return;
        }
    }
}
```

Given the method `flip` defined above, we will determine the best and worst case runtimes when `flop` is defined as:

(a)
```
public static void flop(int a, int b) {
    flip(b - a);
}
```

Best Case: ___Θ( )___ Worst Case: ___Θ( )___

(b)
```
public static void flop(int a, int b) {
    int low = Math.min(a, b - a);
    flip(low);
    flip(low);
}
```

Best Case: ___Θ( )___ Worst Case: ___Θ( )___

(c)
```
public static void flop(int a, int b) {
    flip(a);
    flip(b - a);
}
```

Best Case: ___Θ( )___ Worst Case: ___Θ( )___

2 Some More Analysis

For each of the pieces of code below, give the **worst case** runtime in $\Theta(.)$ notation as a function of N. Your answer should be as simple as possible (i.e. avoid unnecessary constants, lower order terms, etc.). If the worst case is an infinite loop, write an infinity symbol in the blank. Assume there is no limit to the size of an int (otherwise technically they're all constant). (Sping 2015)

(a) Runtime: ___Θ( )___

```
public static void f1(int N) {
    int sum = 0;
    for (int i = N; i > 0; i -= 1) {
        for (int j = 0; j < i; j += 1) {
            sum += 1;
        }
    }
}
```

(b) Runtime: ___Θ( )___

```
public static void f2(int N) {
    int sum = 0;
    for (int i = 1; i <= N; i = i*2) {
        for (int j = 0; j < i; j += 1) {
            sum += 1;
        }
    }
}
```

(c) Runtime: ___Θ( )___

```
public static void f3(int[] a) {
    if (a.length == 0) { return; }
    int N = a.length;
    int[] newA = new int[N-1];
    for (int i = 0; i < newA.length; i += 1) {
        newA[i] = a[i];
    }
    f3(newA);
}
```

(d) Runtime: ___Θ( )___

```
public static void f4(int N) {
    int x = N;
    while (x > 0) {
        x = x >>> 1;
    }
}
```

(e) Runtime: ___Θ( )___

```
public static void f5(int N) {
    f1(N);
    f2(N);
    f3(new int[N]);
    f4(N);
}
```