Name: **Zachary Talarick** Date: **2/13/19**

Pledge: **I pledge my honor that I have abided by the Stevens Honor System.**

Use the Master Theorem to find the complexity of each recurrence relation listed below.

1. Complexity: **Θ(n^2)**
2. Complexity: **Θ(n^2 lg n)**
3. Complexity: **Θ(n)**

For each function below, write the recurrence relation for its running time and then use the Master Theorem to find its complexity.

1. **int** f(**int** arr[], **int** n) {

**if** (n == 0) {

**return** 0;

}

**int** sum = 0;  
 **for** (**int** j = 0; j < n; ++j) {  
 sum += arr[j];  
 }  
 **return** f(arr, n / 2) + sum + f(arr, n / 2);  
}  
Recurrence: **T(n) = 2T(n / 2) + n**

Complexity: **Θ(n lg n)**

1. **void** g(**int** n, **int** arrA[], **int** arrB[]) {

**if** (n == 0) {

**return**;

}

**for** (**int** i = 0; i < n; ++i) {  
 **for** (**int** j = 0; j < n; ++j) {  
 arrB[j] += arrA[i];

}

}  
 g(n / 2, arrA, arrB);  
}  
Recurrence: **T(n) = T(n / 2) + n^2**

Complexity: **Θ(n^2)**