Name: \_\_Grace Miguel\_\_\_ Date: \_\_\_\_\_\_\_3/19/21\_\_\_\_\_\_\_

Pledge: \_\_\_\_I pledge my honor that I’ve abided by the Stevens Honor System.\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Complexity: \_\_Theta(n^2)\_\_\_\_\_\_\_\_

1. Complexity: \_\_\_Theta(n^2lgn)\_\_\_\_\_\_\_\_\_
2. Complexity: \_\_\_Theta(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);  
}

**a= 2**

**b=2**

**f(n) = Theta(n)**

**d=1**   
Recurrence: T(n)=2T(n/2) + Theta(n)\_\_\_\_

Complexity: \_\_Theta(nlgn)\_\_\_\_\_\_\_\_

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);  
}

A=1 b=2 f(n)=n^2 d=2

1 2^2

Recurrence: \_\_T(n)=T(n/2)+Theta(n^2)\_\_\_\_

Complexity: \_\_\_\_Theta(n^2)\_\_\_