Name: John Spicer Date: 11/6/15

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

1. Complexity: θ(n2)
2. Complexity: θ(n2 \* 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) {  
    **int** sum = 0;  
     
    **for** (**int** j = 0; j < n; ++j) {  
    sum += arr[j];  
    }  
     
    **return** f(n / 2) + sum + f(n / 2);  
   }  
     
   Recurrence: T(n) = 2T(n / 2) + n

Complexity: θ(n \* lg(n))

1. **void** g(**int** n, **int** arrA[], **int** arrB[]) {  
    **for** (**int** i = 0; i < n; ++i) {  
    **for** (**int** j = 0; j < n; ++j) {  
    arrB[j] += arrA[i];

}

}  
  
 g(n / 2);  
}  
  
Recurrence: T(n) = T(n / 2) + n2

Complexity: θ(n2)