Name: Liam Brew Date: 03.06.2020

Pledge: I pledge my honor that I have abided by the Stevens Honor System – Liam Brew

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

1. $T(n) = T\left(\frac{n}{2}\right) + n^2$
   Complexity: $\theta(n^2)$

2. $T(n) = 4T\left(\frac{n}{2}\right) + n^2$
   Complexity: $\theta(n^2 * lgn)$

3. $T(n) = 3T\left(\frac{n}{3}\right) + \sqrt{n}$
   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.

4.
```
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: $T(n) = 2T\left(\frac{n}{2}\right) + n$
   Complexity: $\theta(nlgn)$

5.
```
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: $T(n) = T\left(\frac{n}{2}\right) + n^2$
   Complexity: $\theta(n^2)$