Q One Subproblem

Consider the following recurrence.

$$T(1) = d$$
$$T(N) = T(N/2) + cN$$

1. Visualize the recursion tree.
2. Give a closed-form solution for T(N) by unrolling the recurrence. Simplify.

3

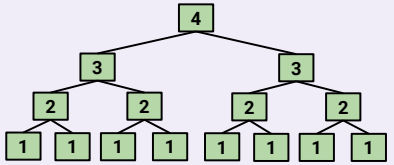
Q1: Visualize the recursion tree.

Q2: Give a closed-form solution for T(N) by unrolling the recurrence. Simplify.

Q Constant Work Per Call

Give a recurrence relation for f3.

```
static int f3(int n) {
    if (n <= 1)
        return 1;
    return f3(n-1) + f3(n-1);
}
```



6

?: What is the recursive work? What is the non-recursive work?

Q1: Give a recurrence relation for f3.

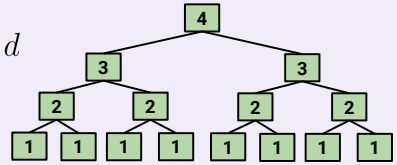
Q Summing Over Levels

Approach: Identify a pattern and then sum over all recursive levels.

```
static int f3(int n) {
    if (n <= 1)
        return 1;
    return f3(n-1) + f3(n-1);
}
```

What is the value of the last term in the sum for T(N)?

$$T(N) = c + 2c + 4c + \cdots + \_\_\_ d$$



Give a simple asymptotic runtime bound.

10

Q1: What is the value of the last term in the sum for T(N)?

Q2: Give a simple asymptotic runtime bound.

Q Runtime: g0

Give best and worst case runtime. Assume k(N) runs in constant time and returns a boolean.

```
static void g0(int N) {
    if (N == 0)
        return;
    g0(N / 2);
    if (k(N))
        g0(N / 2);
}
```

13

Q1: Give best and worst case runtime. Assume k(N) runs in constant time and returns a boolean.