CSE 332 - Section 3 Worksheet

0. Recurrence Relations

a) Find a recurrence T(n) modeling the worst-case runtime complexity of f(n)

```
1  f(stack) {  // stack.size = n
2    if (stack.size == 0) {
3      return 0
4    }
5    stack.pop  // stack.size --
6    return 2*f(stack)+1
7  }
```

b) Find a recurrence T(n) modeling the worst-case runtime complexity of bar(n)

```
1  bar(A, start, n) {
2    if (n ≤ 10000) {
3      return 1000
4    }
5    if (bar(A, start, n/3) > 5) {
6      for (int i=0; i<n; i++) {
7        println("Yay")
8      }
9      return 5 * bar(A, start, n/3)
10   } else {
11     for (int i=0; i<n*n; i++) {
12       println("Yay")
13     }
14     return 4 * bar(A, start, n/3)
15   }
```

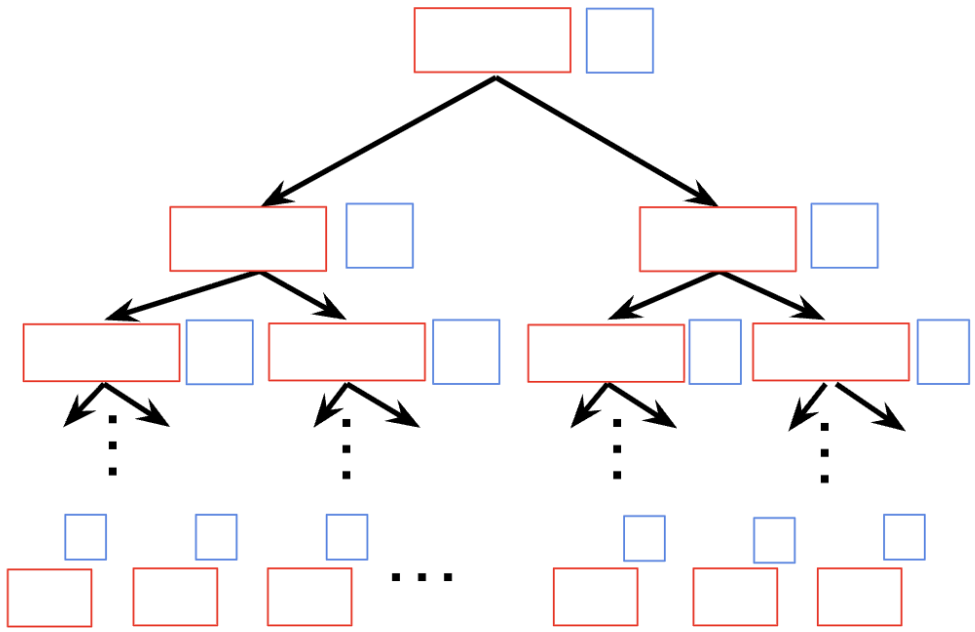
CSE 332 - Section 3 Worksheet

1. Tree Method

For each of the following recurrence relations, use the tree method to convert it to closed form:

a)

$$T(n) = \begin{cases} 1 & \text{if } n \leq 1 \\ 2T\left(\frac{n}{2}\right) + n & otherwise \end{cases}$$



Work per level:

levels:

Closed form running time:

CSE 332 - Section 3 Worksheet

b)

$$T(n) = \begin{cases} 4 & \text{if } n \leq 4 \\ 2T\left(\frac{n}{2}\right) + n & otherwise \end{cases}$$

Work per level:
levels:
Closed form running time:

c)

$$T(n) = \begin{cases} 100 & \text{if } n \leq 1 \\ 2T\left(\frac{n}{2}\right) + 100n & otherwise \end{cases}$$

Work per level:
levels:
Closed form running time:

CSE 332 - Section 3 Worksheet

d)

$$T(n) = \begin{cases} 1 & \text{if } n \leq 1 \\ 3T\left(\frac{n}{2}\right) + n & otherwise \end{cases}$$

Work per level:
levels:
Closed form running time:

e)

$$T(n) = \begin{cases} 1 & \text{if } n \leq 1 \\ 2T\left(\frac{n}{3}\right) + n & otherwise \end{cases}$$

Work per level:
levels:
Closed form running time:

CSE 332 - Section 3 Worksheet

f)

$$T(n) = \begin{cases} 1 & n \leq 1 \\ 2T(n-2) + 1 & otherwise \end{cases}$$

Work per level:
levels:
Closed form running time:

CSE 332 - Section 3 Worksheet

2. Putting It All Together

Consider the function $f(L, start, j)$. Find a recurrence modeling the worst-case runtime of this function and then find a Big-Oh bound for this recurrence.

```
1  f(L, start, j) { // n = j - start
2      if (j - start <= 1) {
3          return 0
4      }
5      int result = f(L, start, j/2)
6      for (int i = start; i < j; i++) {
7          result *= 4
8      }
9      return result + f(L, start, j/2)
10 }
```

a) Find a recurrence $T(n)$ modeling the *worst-case runtime complexity* of $f(L, 0, L.size())$ where $n = j - start$

b) Use your answer in part (a) to find a closed form for $T(n)$