Trying Again: Counting how long recursive functions run

```
int sum(int[] arr){
  return help(arr,0);
}

int help(int[]arr,int i) {
  if(i==arr.length)
    return 0;
  return arr[i] + help(arr,i+1);
}
```

$$T(n) = \begin{cases} \end{cases}$$

7/01/2022 7

7

You Try

pollev.com/artliu

```
Mystery(int n){
     if(n <= 4)
          return 1;
     for(int i=0; i < n; i++){
          if(i % 3 == 2)
               break;
     }
     return 4 * Mystery(n - 5) + Mystery(n / 2)
}
```

$$T(n) = \begin{cases} \end{cases}$$

7/01/2022 12

12

Unrolling: Solving Recurrence Relations

$$T(n) = \begin{cases} T(n-1) + c_1 & \text{if } n > 0 \\ c_0 & \text{otherwise} \end{cases}$$

Write it out to get a general form, then solve for base-case:

7/01/2022 17

17

Tree Method

$$T(n) = \begin{cases} 2 * T(n/2) + c_1 & \text{if } n > 1 \\ c_0 & \text{otherwise} \end{cases}$$

7/01/2022 22

22

Another Example

$$T(n) = \begin{cases} 2 * T(n/2) + c_1 n & \text{if } n > 1 \\ c_0 & \text{otherwise} \end{cases}$$

27

Really common recurrences

Should know how to solve recurrences but also recognize some really common ones:

$T(n) = O(1) + T(n/2)$	logarithmic	$O(\log n)$	binary search
$T(n) = O(1) + 2T(n/2)$	linear	$O(n)$	recursive "binary" sum
$T(n) = O(1) + T(n\text{-}1)$	linear	$O(n)$	recursive sum
$T(n) = O(n) + T(n\text{-}1)$	quadratic	$O(n^2)$	
$T(n) = O(1) + 2T(n\text{-}1)$	exponential	$O(2^n)$	
$T(n) = O(n) + T(n/2)$	linear	$O(n)$	
$T(n) = O(n) + 2T(n/2)$	loglinear	$O(n \log n)$	merge sort

31