1. Give simplified Big-O

$\cancel{10}n\log n + \cancel{3}n \to O(n\log n) + O(n) \to \boxed{O(n\log n)}$

$\cancel{5}n^2\log n + \cancel{13}n^3 \to O(n^2\log n) + O(n^3) \to \boxed{O(n^2\log n)}$

$\cancel{20}n\log\log n + \cancel{2}\,n\log n \to O(n\log\log n) + O(n\log n) \to \boxed{O(n\log\log n)}$

$2^{3n} \to \cancel{2^3}\cdot 2^n \to \boxed{O(2^n)}$

$f(n) = O(g(n))$

2. Using definition of Big O, show

$$10n^2 + 15n \text{ is } O(n^2)$$

$f(n) \le c\cdot g(n)$

$\forall n \ge n_0$

$10n^2 + 15n \le 10n^2 + n^2$

$\underbrace{10n^2 + 15n}_{f(n)} \le \underbrace{11}_{c}\underbrace{n^2}_{g(n)}$

$\therefore f(n) \le c\cdot g(n)$

$\forall n \ge 11$

3. True or False, would you give this as a solution in class?

True or $\boxed{\text{False:}}$ $10n^2 + 15n$ is $O(n^3)$

4. Write the recurrence relation

$T(n)$

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

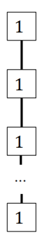
$\boxed{1 + T(n-5)}$

5. Write the recurrence relation

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

6. Solve the recurrence relation for binary search using a Tree (come back to after Binary Search Tree)

$$T(n) = \begin{cases} 1 \; when \; n \le 1 \\ T\left(\frac{n}{2}\right) + 1 \; otherwise \end{cases}$$



0. Draw the tree.
1. What is the input size at level i?
2. What is the number of nodes at level i?
3. What is the work done at recursive level i?
4. What is the last level of the tree?
5. What is the work done at the base case?
6. Sum over all levels (using 3,5).
7. Simplify

7. Worst case tight bound runtime

```
int x = 0
for (int i = n; i >= 0; i--) {
   if ((i % 3) == 0) {
      break
   }
   else {
      x += n
   }
}
```

$\boxed{O(n)}$

8. Worst case tight bound runtime

```
int x = 0
for (int i = 0; i < n; i++) {
   for (int j = 0; j < (n * n / 3); j++) {
      x += j
   }
}
```

$$\frac{n \cdot n \cdot n}{3} = \frac{n^3}{3} = \frac{1}{3} n^3$$

$$\boxed{O(n^3)}$$

9. Worst case tight bound runtime

```
int x = 0
for (int i = 0; i < n; i++) {
   for (int j = 0; j < i; j++) {
      x += j
   }
}
```

$$\boxed{O(n^2)}$$