Nathan Hutton

1.

- $m(n)$

  $m(n-1)+3$

  $(m(n-2)+3)+3$

  $m(n-(n-1))+3(n-1)$

  $m(1)+3(n-1)$

  $3n-3+4$

  $3n+1$

- $m(n)$

  $(m(n-1)\cdot 2)+3$

  $(((m(n-2)\cdot 2)+3)\cdot 2)+3$

  $(((((m(n-3)\cdot 2)+3)\cdot 2)+3)\cdot 2)+3)$ or $m(n-2)2^3+3(2^3)-3$

  $m(n-(n-1))\cdot 2^{(n-1)}+3(2^{(n-1)})-3$

  $m(1)\cdot 2^{(n-1)}+3(2^{(n-1)})-3$

  $9\cdot 2^{(n-1)}+3(2^{(n-1)})-3$

- $m(n)$

  $(m(n-1)\cdot 4)$

  $(m(n-2)\cdot 4)\cdot 4$

  $((m(n-3)\cdot 4)\cdot 4)\cdot 4$

  $m(n-3)\cdot 4^3$

  $m(n-(n-1))\cdot 4^{(n-1)}$

  $m(1)\cdot 4^{(n-1)}$

  $2\cdot 4^{(n-1)}$

2.

- $m(n) = 2m(n/2) + n$ $\quad a=2,\ b=2,\ f(n)=n,\ d=1$

  $m(n)$ is $O(n \log n)$ since $a = b^d$ and $d=1$

- $m(n) = 4m(n/4) + n^2$ $\quad a=4,\ b=4,\ f(n)=n,\ d=2$

  $m(n)$ is $O(n^2)$ since $a < b^d$ and $d=2$

3. - $T(n) = 2T(n-1) + 1$
     - Exponential

4.

~~count_leaves function w/ root~~

- def count_leaves(root):

```
def count_leaves(root):
    if root is null:
        return 0
    elif left and right child of root are null:
        return 1
    else:
        return count_leaves(root.left) + count_leaves(root.right)
```

- $T(n-1) + 5$, linear

  I don't feel like this question was very fair. I may submit a comment w/ a different answer later since I don't think this one was correct after I get more information about the question from you on Slack.