Marinel Tinnirello
11/15/2019
CMPT 435 - Assignment #9

1)

a) $T(n) = 2T(n/2) + n^3$ a = 2 b = 2 d = 3
$T(n) = (4/3)n^3$
$T(n) = n^3$

b) $T(n) = T(9n/10) + n$ a = 1 b = 10 d = 1
$T(n) = 1 + 10n$
$T(n) = 10n$

c) $T(n) = 16T(n/4) + n^2$ a = 16 b = 4 d = 2
$T(n) = n^2\log(n)$
$T(n) = n^2$

d) $T(n) = 7T(n/3) + n^2$ a = 7 b = 3 d = 2
$T(n) = n^{\log3(7)}$
$T(n) = n^2$

e) $T(n) = 2T(n/4) + \text{sqrt}(n)$ a = 2 b = 4 d = 1
$T(n) = \text{sqrt}(n)\log(n)$
T(n) = sqrt(n)

2)

a) Set the maxes up as a pair
Base Case:
If there's only 1 element, set the maxes equal to that element (each other)
If there's only 2 elements, compare which is the higher max
General case:
Create a mid, left array, right array
Run the left and right arrays through recursion to find the maxes
Run the result through method max1max2 to find the max and 2nd max

b)
```
if (i == j)                                              // 1
        result.max = A[i]
        result.max2nd = A[j]
if (i == j + 1)                                          // 1
        if (A[i] > A[j])                                 // 2
                result.max = A[i]
                result.max2nd = A[j]
        else                                             // 2
                result.max = A[j]
                result.max2nd = A[i]
else
        left = dcfindmax2ndmax(A, i, mid)                // n
        right = dcfindmax2ndmax(A, mid + 1, j)           // n
        result = max1max2(left, right)                   // n
```

c) C(n): 3/2 n - 2