Name:_____

NetID:_____ **Lecture: A B**

Discussion: Thursday Friday 10 11 12 1 2 3 4 5 6

```
01 Munch(a_1, ..., a_n: an array of n positive integers)
02   if (n = 1) return a_1
03   else if (n = 2) return a_1 + a_2
04   else if (n = 3) return a_1 + a_2 + a_3
05   else
06       p = ⌊n/3⌋
07       q = ⌊2n/3⌋
08       rv = Munch(a_1, ..., a_p) + Munch(a_{q+1}, ..., a_n)
09       rv = rv + Munch(a_{p+1}, ..., a_q)
10       return rv
```

Dividing an array takes constant time.

1. (5 points) Let $T(n)$ be the running time of Munch. Give a recursive definition of $T(n)$.

2. (3 points) What is the height of the recursion tree for $T(n)$, assuming n is a power of 3?

3. (3 points) What is amount of work (aka sum of the values in the nodes) at level k of this tree?

4. (4 points) What is the big-Theta running time of Munch? Briefly justify your answer.

Name:_____

NetID:_____ **Lecture: A B**

Discussion: Thursday Friday 10 11 12 1 2 3 4 5 6

```
01 Crunch(a_0, ..., a_{n-1}))  \\ input is an array of n integers
02       if (n = 2 and a_0 > a_1)
03             swap(a_0, a_1)   \\ interchange the values at positions 0 and 1 in the array
04       else if (n > 2)
05             p = ⌊n/4⌋
06             q = ⌊n/2⌋
07             r = p + q
08             Crunch(a_0, ..., a_q)   \\ constant time to make smaller array
09             Crunch(a_{q+1}, ..., a_{n-1})   \\ constant time to make smaller array
10             Crunch(a_p, ..., a_r)   \\ constant time to make smaller array
```

1. (5 points) Suppose that $T(n)$ is the running time of Crunch on an input array of length n. Give a recursive definition of $T(n)$.

2. (4 points) What is the height of the recursion tree for $T(n)$, assuming n is a power of 2?

3. (3 points) What is the amount of work (aka sum of the values in the nodes) at level k of this tree?

4. (3 points) How many leaves are in the recursion tree for $T(n)$? (Simplify your answer.)

Name:_____

NetID:_____ **Lecture: A B**

Discussion: Thursday Friday 10 11 12 1 2 3 4 5 6

```
01 Crumple(a_1, ..., a_n: a list of n positive integers)
02   if (n = 1) return a_1
03   else if (n = 2) return a_1 + a_2
04   else if (n = 3) return a_1 + a_2 + a_3
05   else
06       p = ⌊n/3⌋
07       q = ⌊2n/3⌋
08       rv = Crumple(a_1, ..., a_p) + Crumple(a_{q+1}, ..., a_n)
09       rv = rv + Crumple(a_{p+1}, ..., a_q)
10       return rv
```

Dividing a list takes $O(n)$ time.

1. (5 points) Let $T(n)$ be the running time of Crumple. Give a recursive definition of $T(n)$.

2. (3 points) What is the height of the recursion tree for $T(n)$, assuming n is a power of 3?

3. (3 points) What is amount of work (aka sum of the values in the nodes) at level k of this tree?

4. (4 points) What is the big-Theta running time of Crumple?

Name:_____

NetID:_____ **Lecture: A B**

Discussion: Thursday Friday 10 11 12 1 2 3 4 5 6

```
01 Slide(a_1, ..., a_n)  \\ input is a linked list of n integers
02        if (n = 1) return a_1
03        else
04              m = ⌊n/2⌋
05              p = Slide(a_1, ..., a_m)    \\ O(n) time to split list
06              q = Slide(a_{m+1}, ..., a_n)    \\ O(n) time to split list
06              return max(p,q)
```

1. (5 points) Suppose that $T(n)$ is the running time of Slide on an input array of length n and assume that n is a power of 2. Give a recursive definition of $T(n)$.

2. (4 points) What is the height of the recursion tree for $T(n)$?

3. (3 points) What is the amount of work (aka sum of the values in the nodes) at non-leaf level k of this tree?

4. (3 points) What is the big-Theta running time of Slide?

Name:_____

NetID:_____ **Lecture: A B**

Discussion: Thursday Friday 10 11 12 1 2 3 4 5 6

```
01 Swing(a_1, ..., a_n; b_1, ..., b_n)  \\ input is 2 arrays of n integers, n is a power of 2
02       if (n = 1)
03             return a_1 b_1
04       else
05             p = n/2
06             rv = Swing(a_1, ..., a_p, b_1, ..., b_p)
07             rv = rv + Swing(a_1, ..., a_p, b_{p+1}, ..., b_n)
08             rv = rv + Swing(a_{p+1}, ..., a_n, b_{p+1}, ..., b_n)
09             rv = rv + Swing(a_{p+1}, ..., a_n, b_1, ..., b_p)
10             return rv
```

1. (5 points) Suppose that $T(n)$ is the running time of Swing on an input array of length n. Give a recursive definition of $T(n)$. Assume that dividing an array in half takes constant time.

2. (3 points) What is the height of the recursion tree for $T(n)$, assuming n is a power of 2?

3. (3 points) What is the amount of work (aka sum of the values in the nodes) at level k of this tree?

4. (4 points) What is the big-Theta running time of Swing. Briefly justify your answer. Recall that $\sum_{k=0}^{n} a^k = \frac{a^{n+1}-1}{a-1}$.

Name:_____

NetID:_____ **Lecture: A B**

Discussion: Thursday Friday 10 11 12 1 2 3 4 5 6

```
01 Wave(a_1,..., a_n)  \\ input is an array of n positive integers
02      m := 0
03      for i := 1 to n − 1
04          for j := i + 1 to n
05              if |a_i − a_j| > m then m := |a_i − a_j|
06      return m
```

1. (3 points) What value does the algorithm return if the input list is $4, 13, 20, 5, 8, 10$

2. (3 points) Let $T(n)$ be the number of times that line 5 is executed. Express $T(n)$ using summation notation, directly following the structure of the code.

3. (3 points) Find an (exact) closed form for $T(n)$. Show your work.

4. (3 points) What is the big-theta running time of Wave?

5. (3 points) Check the (single) box that best characterizes each item.

The running time of mergesort is recursively defined by $T(1) = d$ and $T(n) =$

$2T(n-1)+c$	☐	$2T(n-1)+cn$	☐
$2T(n/2)+c$	☐	$2T(n/2)+cn$	☐