CS 173, Spring 19 Examlet 11, colored sheet

Name:____

NetID:____________________ **Lecture:** **A** **B**

Discussion: **Thursday** **Friday** **9** **10** **11** **12** **1** **2** **3** **4** **5** **6**

Jump($a_1, \ldots, a_n$: an array of n positive integers)
  if (n = 1) return a_1
  else if (n = 2) return $a_1 + a_2$
  else if (n = 3) return $a_1 + a_2 + a_3$
  else
      p = $\lfloor n/3 \rfloor$
      q = $\lfloor 2n/3 \rfloor$
      rv = Jump($a_1, \ldots, a_p$) + Jump($a_{q+1}, \ldots, a_n$)
      rv = rv + Jump($a_{p+1}, \ldots, a_q$)
      return rv

Dividing an array takes constant time.

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

   Solution:

   $T(1) = a$

   $T(2) = b$

   $T(3) = c$

   $T(n) = 3T(n/3) + d$

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

   Solution: $\log_3(n) - 1$

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

   Solution: $d3^k$

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

   Solution: The number of leaves is $3^{\log_3 n - 1} = \frac{n}{3}$, which is $\Theta(n)$. The total number of nodes is proportional to the number of leaves. Since each node contains a constant amount of work, the running time is proportional to the number of nodes. So the running time is $\Theta(n)$.

Name:____

NetID:____________________ **Lecture: A B**

Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6

```
Swing(k,n)   \\ inputs are positive integers
    if (n = 1) return k
    else if (n = 2) return k^2
    else
        half = ⌊n/2⌋
        answer = Swing(k,half)
        answer = answer*answer
        if (n is odd)
            answer = answer*k
        return answer
```

1. (5 points) Suppose $T(n)$ is the running time of Swing. Give a recursive definition of $T(n)$.

   Solution:

   $T(1) = c$, $T(2) = d$

   $T(n) = T(n/2) + f$

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

   Solution: $\log_2 n - 1$

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

   Solution: One.

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

   Solution: $\Theta(\log n)$

Name:____

NetID:______________________ **Lecture: A B**

Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6

```
Waltz(a_1, a_2, ... a_n: list of real numbers)
  if (n = 1) then return 0
  else if (n = 2) then return |a_1 − a_2|
  else
    L = Waltz(a_2,a_3,...,a_n)
    R = Waltz(a_1,a_2,...,a_{n−1})
    Q = |a_1 − a_n|
    return max(L,R,Q)
```

Removing the first element of a list takes constant time; removing the last element takes $O(n)$ time.

1. (3 points) Give a succinct English description of what Waltz computes.

   Solution: Waltz computes the largest difference between two values in its input list.

2. (4 points) Suppose $T(n)$ is the running time of Waltz. Give a recursive definition of $T(n)$.

   Solution: $T(1) = d_1$ $T(2) = d_2$

   $T(n) = 2T(n-1) + cn$

3. (4 points) What is the height of the recursion tree for $T(n)$?

   Solution: We hit the base case when $n-k = 2$, where k is the level. So the tree has height $n-2$.

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

   Solution: 2^{n-2}

Name:____

NetID:____________________ **Lecture: A B**

Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6

```
Grind(a_1, ..., a_n)  \\ input is a sorted array of n integers
    if (n = 1) return a_1
    else
        m = ⌊n/2⌋
        if a_m > 0
            return Grind(a_1, ..., a_m)   \\ constant time to extract part of array
        else
            return Grind(a_{m+1}, ..., a_n)   \\ constant time to extract part of array
```

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

   Solution:

   $T(1) = c$

   $T(n) = T(n/2) + d$

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

   Solution: $\log_2 n$

3. (3 points) How many leaves does this tree have?

   Solution: One.

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

   Solution: $\Theta(\log n)$

Name:____

NetID:________________________ **Lecture: A B**

Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6

```
Weave(a_0, ..., a_{n-1}))  \\ input is an array of n integers (n ≥ 2)
    if (n = 2 and a_0 > a_1)
        swap(a_0, a_1)  \\ interchange the values at positions 0 and 1 in the array
    else if (n > 2)
        p = ⌊n/4⌋
        q = ⌊n/2⌋
        r = p + q
        Weave(a_0, ..., a_q)  \\ constant time to make smaller array
        Weave(a_{q+1}, ..., a_{n-1})  \\ constant time to make smaller array
        Weave(a_p, ..., a_r)  \\ constant time to make smaller array
```

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

   Solution:

   $T(2) = d$

   $T(n) = 3T(n/2) + f$

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

   Solution: $\log_2 n - 1$

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

   Solution: $f \cdot 3^k$

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

   Solution: $3^{\log_2 n - 1} = 1/3(3^{\log_2 n}) = 1/3(3^{\log_3 n \log_2 3}) = 1/3 \cdot n^{\log_2 3}$

Name:____

NetID:________________________ **Lecture:** **A** **B**

Discussion: **Thursday** **Friday** **9** **10** **11** **12** **1** **2** **3** **4** **5** **6**

```
Act(a_1, ..., a_n; b_1, ..., b_n)  \\ input is 2 arrays of n integers, n is a power of 2
        if (n = 1)
                return a_1 b_1
        else
                p = n/2
                rv = Act(a_1, ..., a_p, b_1, ..., b_p)
                rv = rv + Act(a_1, ..., a_p, b_{p+1}, ..., b_n)
                rv = rv + Act(a_{p+1}, ..., a_n, b_{p+1}, ..., b_n)
                rv = rv + Act(a_{p+1}, ..., a_n, b_1, ..., b_p)
                return rv
```

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

   Solution:

   $T(1) = c$

   $T(n) = 4T(n/2) + d$

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

   Solution: $\log_2 n$

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

   Solution: There are 4^k nodes, each containing n. So the total work is $4^k d$

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

   Solution: The number of leaves is $4^{\log_2 n} = 4^{\log_4 n \log_2 4} = n^{\log_2 4} = n^2$ which is $\Theta(n^2)$. The total number of nodes is proportional to the number of leaves (because $\sum_{k=0}^{n} 4^k = \frac{4^{n+1}-1}{3}$). Since each node contains a constant amount of work, the running time is proportional to the number of nodes. So the running time is $\Theta(n^2)$.

Name:____

NetID:________________________ **Lecture: A B**

Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6

```
01 Dig (a_1, ..., a_n: array of integers)
02     if (n = 1)
03         if (a_1 > 8) return true
04         else return false
05     else if (Dig(a_1, ..., a_{n-1}) is true and Dig(a_2, ..., a_n) is true)
06         return true
07     else return false
```

1. (3 points) If Dig returns true, what must be true of the values in the input array?

   Solution: The values in the input array must all be greater than 8.

2. (5 points) Give a recursive definition for $T(n)$, the running time of Dig on an input of length n, assuming it takes constant time to set up the recursive calls in line 05.

   Solution:

   $T(1) = c$

   $T(n) = 2T(n-1) + d$

3. (3 points) What is the height of the recursion tree for $T(n)$?

   Solution: $n - 1$

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

   Solution: $\Theta(2^n)$

Name:____

NetID:______________________ **Lecture:** **A** **B**

Discussion: **Thursday** **Friday** **9** **10** **11** **12** **1** **2** **3** **4** **5** **6**

```
Swim(a_1, ..., a_n)  \\ input is a sorted list of n integers
        if (n = 1) return a_1
        else
                m = ⌊n/2⌋
                if a_m > 0
                        return Swim(a_1, ..., a_m)   \\ O(n) time to extract half of list
                else
                        return Swim(a_{m+1}, ..., a_n)   \\ O(n) time to extract half of list
```

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

   Solution:

   $T(1) = c$

   $T(n) = T(n/2) + dn$

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

   Solution: $\log_2 n$

3. (3 points) What value is in each node at level k of this tree?

   Solution: $dn/2^k$

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

   Solution: $\Theta(n)$

   [more detail than you need to supply] There is only one node at each level. So the total work is $c+d(n+n/2+\ldots+2)$. The dominant term of this is proportional to $n\sum_{k=0}^{\log n} 1/2^k = n(2-1/2^{\log n}) = n(2-1/n) = 2n-1$.