CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST: **LAST:**

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

```
Shake(p_1, ..., p_n : list of n 2D points, n ≥ 3)
    if (n = 3)
        return the largest of d(p_1, p_2), d(p_1, p_3), and d(p_2, p_3)
    else
        x = Shake(p_2, p_3, p_4, ..., p_n)
        y = Shake(p_1, p_3, p_4, ..., p_n)
        z = Shake(p_1, p_2, ..., p_{n-1})
        return max(x, y,z)
```

The function $d(p, q)$ returns (in constant time) the straight-line distance between two points p and q. Removing the first element of a list takes constant time; removing the last element takes $O(n)$ time.

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

   Solution: $T(3) = c$

   $T(n) = 3T(n-1) + dn + f$

2. (4 points) What is the amount of work (aka sum of the values in the nodes) at non-leaf level k of this tree?

   Solution: At level k, there are 3^k nodes and each node contains $d(n-k) + f$. So the total work is $3^k(dn - dk + f)$.

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

   Solution: 3^{n-3}

4. (3 points) Is the running time of Shake $O(2^n)$?

   Solution: No, the running time can't be $O(2^n)$. The work in the leaves is $\Theta(3^n)$ and 3^n grows faster than 2^n.

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST: **LAST:**

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

```
Rattle(k,n)   \\ inputs are natural numbers
        if (n = 0) return 1
        else if (n = 1) return k
        else if (n is odd)
               temp = Rattle(k,floor(n/2))
               return k*temp*temp
        else
               temp = Rattle(k,floor(n/2))
               return temp*temp
```

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

   Solution: $T(0) = T(1) = c$

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

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

   Solution: $\log_2 n$

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

   Solution: One.

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

   Solution: $\Theta(\log n)$

CS 173, Fall 2016 Examlet 11, Part A

NETID:

FIRST: **LAST:**

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

```
Roll(a_1, ..., a_n: a list of n positive integers)
    if (n = 1) return a_1
    else if (n = 2) return max(a_1, a_2)
    else if (a_1 < a_n)
        return Roll(a_2, ..., a_n)
    else
        return Roll(a_1, ..., a_{n-1})
```

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

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

   Solution: $T(1) = c$

   $T(2) = d$

   $T(n) = T(n-1) + pn$

2. (3 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 $n - 2$.

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

   Solution: Notice that the tree doesn't branch, so there is only one node at each level. So the total amount of work at level k is $p(n-k)$.

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

   Solution:

   $\Theta(n^2)$

   [Much more detail than you needed to give:] Notice that the sum of all the non-leaf nodes is $\sum_{k=1}^{n-3} p(n-k)$. If we move the constant p out of the summation and substitute in the new index value $j = n - k$, we get

   $$p\sum_{j=3}^{n-1} j = p\sum_{j=1}^{n-1} j) - 3 = p\frac{(n-1)n}{2} - 3 = \frac{p}{2}n^2 - \frac{p}{2}n - 3$$

   The dominant term of this is proportional to n^2.

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST:	**LAST:**

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

```
Bounce(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 Bounce(a_1, ..., a_m)   \\ O(n) time to extract half of list
                else
                        return Bounce(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 Bounce on an input list of length n and assume that n is a power of 2. Give a recursive definition of $T(n)$.

   Solution:

   $T(1) = c$

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

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: $n/2^k$

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

   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 $n\sum_{k=0}^{\log n} 1/2^k = n(2-1/2^{\log n}) = n(2-1/n) = 2n-1$.

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST: | **LAST:**

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

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

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

   Solution:

   $T(1) = c$

   $T(n) = 4T(n/2) + dn + f$

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

   Solution: $\log_2 n$

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 $f + dn/2^k$. So the total work is $4^k f + 2^k dn$

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

   Solution: $4^{\log_2 n} = 4^{\log_4 n \log 24} = n^{\log 24} = n^2$

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST: **LAST:**

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

```
Swing (a_1, ..., a_n: list of integers)
    if (n = 1)
        if (a_1 is even) return true
        else return false
    else if (Swing(a_1, ..., a_{n-1}) is true or Swing(a_2, ..., a_n) is true)
        return true
    else return false
```

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

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

   Solution: The input list contains at least one even value.

2. (5 points) Give a recursive definition for $T(n)$, the running time of Swing on an input of length n.

   Solution:

   $T(1) = c$

   $T(n) = 2T(n-1) + dn + f$

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

   Solution: $n - 1$

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

   Solution: 2^{n-1}