CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST:

LAST:

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

```
01 Shake(p_1, ..., p_n : list of n 2D points, n ≥ 3)
02       if (n = 3)
03             return the largest of d(p_1, p_2), d(p_1, p_3), and d(p_2, p_3)
04       else
05             x = Shake(p_2, p_3, p_4, ..., p_n)
06             y = Shake(p_1, p_3, p_4, ..., p_n)
07             z = Shake(p_1, p_2, ..., p_{n-1})
08             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 $T(n)$.

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?

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

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

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST:

LAST:

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

```
01 Rattle(k,n)   \\ inputs are natural numbers
02         if (n = 0) return 1
03         else if (n = 1) return k
04         else if (n is odd)
05                 temp = Rattle(k,floor(n/2))
06                 return k*temp*temp
07         else
08                 temp = Rattle(k,floor(n/2))
09                 return temp*temp
```

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

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

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

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

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST: | **LAST:**

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

```
01 Roll(a_1, ..., a_n: a list of n positive integers)
02     if (n = 1) return a_1
03     else if (n = 2) return max(a_1, a_2)
04     else if (a_1 < a_n)
05         return Roll(a_2, ..., a_n)
06     else
07         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 $T(n)$.

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

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 Roll?

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST:

LAST:

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

```
01 Bounce(a_1, ..., a_n)  \\ input is a sorted list of n integers
02       if (n = 1) return a_1
03       else
04            m = ⌊n/2⌋
05            if a_m > 0
06                 return Bounce(a_1, ..., a_m)   \\ O(n) time to extract half of list
07            else
08                 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 $T(n)$.

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

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

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

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST:

LAST:

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

```
01 Skip(a_1, ..., a_n; b_1, ..., b_n)  \\ input is 2 lists 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 = Skip(a_1, ..., a_p, b_1, ..., b_p)
07             rv = rv + Skip(a_1, ..., a_p, b_{p+1}, ..., b_n)
08             rv = rv + Skip(a_{p+1}, ..., a_n, b_{p+1}, ..., b_n)
09             rv = rv + Skip(a_{p+1}, ..., a_n, b_1, ..., b_p)
10             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.

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.)

CS 173, Fall 2016
Examlet 11, Part A

NETID:

FIRST:

LAST:

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

```
01 Swing (a_1, ..., a_n: list of integers)
02     if (n = 1)
03         if (a_1 is even) return true
04         else return false
05     else if (Swing(a_1, ..., a_{n-1}) is true or Swing(a_2, ..., a_n) is true)
05         return true
06     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?

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

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

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