CS 173, Spring 2015 Examlet 11, Part A

NETID:

FIRST: **LAST:**

Discussion: Monday 9 10 11 12 1 2 3 4 5

maxpair($a_1, \ldots, a_n$: an array of n positive integers, $n \geq 2$)
  if (n = 1) return 0
  else if (n = 2) return $a_1 + a_2$
  else
    p = $\lfloor n/3 \rfloor$
    q = $\lfloor 2n/3 \rfloor$
    rv = max(maxpair($a_1, \ldots, a_p$), maxpair($a_{q+1}, \ldots, a_n$))
    for i=p to q
      rv = max(rv, $a_i + a_{i+1}$)
    return rv

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

   Solution:

   $T(2) = c$

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

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

   Solution:

   $\log_3(n)$

   [If n is a power of 2, it will hit the $n = 1$ base case and not the $n = 2$ base case.]

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

   Solution:

   $\frac{dn}{3^k}2^k + f2^k$

4. (4 points) How many leaves does this recursion tree have? Simplify so that your answer is easy to compare to standard running times. Recall that $\log_b x = \log_a x \log_b a$.

   Solution: $2^{\log_3 n} = n^{log_3 2}$

CS 173, Spring 2015
Examlet 11, Part A

NETID:

FIRST: **LAST:**

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

```
1 procedure ComputeIt(a_1,..., a_n)  \\ input is an array of n positive integers
2     m := 0
3     for i := 1 to n − 1
4         for j := i + 1 to n
5             if |a_i − a_j| > m then m := |a_i − a_j|
6     return m
```

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

   Solution: $20 - 4 = 16$

2. (4 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.

   Solution:

   $$T(n) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} 1$$

3. (4 points) Find an (exact) closed form for $T(n)$. Show your work.

   Solution: $T(n) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} 1 = \sum_{i=1}^{n-1} (n-i) = (\sum_{i=1}^{n-1} n) - (\sum_{i=1}^{n-1} i) = n(n-1) - \frac{n(n-1)}{2} = \frac{n(n-1)}{2}$

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

   Solution: $\Theta(n^2)$

CS 173, Spring 2015
Examlet 11, Part A

NETID:

FIRST: | **LAST:**

Discussion: Monday 9 10 11 12 1 2 3 4 5

```
Magic(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 = Magic(a_2,a_3,...,a_n)
    R = Magic(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 Magic computes.

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

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

   Solution:

   $T(1) = d_1 \qquad T(2) = d_2$

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

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}

CS 173, Spring 2015
Examlet 11, Part A

NETID:

FIRST: | **LAST:**

Discussion: Monday 9 10 11 12 1 2 3 4 5

```
01 Frog(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 = Frog(p_2, p_3, p_4, ..., p_n)    \\ removing p_1 from list takes constant time
06         y = Frog(p_1, p_3, p_4, ..., p_n)    \\ removing p_2 from list takes constant time
07         z = Frog(p_1, p_2, p_4, ..., p_n)    \\ removing p_3 from list takes constant time
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.

1. (5 points) Suppose $T(n)$ is the running time of Frog on an input array of length n. Give a recursive definition of $T(n)$. Assume that setting up the recursive calls in lines 5-7 takes constant time.

   Solution:

   $T(3) = c$

   $T(n) = 3T(n-1) + d$

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

   Solution: We hit the base case when $n - k = 3$, where k is the level. So the tree has height $n - 3$.

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

   Solution: 3^{n-3}

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

   Solution: $\Theta(3^n)$

CS 173, Spring 2015
Examlet 11, Part A

NETID:

FIRST: **LAST:**

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

```
diff(a_1, ..., a_n) : list of n positive integers, n ≥ 2)
    if (n = 2) return |a_1 − a_2|
    else
        bestval = 0
        for k = 1 to n
            newval = diff(a_1, a_2, ..., a_{k−1}, a_{k+1}, ... a_n)   \\ constant time to remove a_k
            if (newval > bestval) bestval = newval
        return bestval
```

1. (3 points) Describe (in English) what diff computes.

   Solution: Diff computes the largest difference between two values in the list. Or, equivalently, the largest value minus the smallest value.

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

   Solution:

   $T(2) = c$

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

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

   Solution: $n - 2$

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

   Solution: $\frac{n!}{2}$

CS 173, Spring 2015
Examlet 11, Part A

NETID:

FIRST: **LAST:**

Discussion: Monday 9 10 11 12 1 2 3 4 5

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

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

   Solution:

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

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

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 Getpower?

   Solution: $\Theta(\log n)$