Solutions To Homework D
Grader Natasha Duncan Natasha@udel.edu

Part A (4 points)

We will prove by induction that Stooge sort correctly sorts the input array $A[1n]$. The base cases are for $n = 1$ and $n = 2$. If $n = 1$ then $i = j$, and thus $i+1 > j$; then, line 4 returns $A[i]$ (which is correct). If $n = 2$ then lines 1 and 2 guarantee that a bigger element will be returned in a position following that of a smaller element. Thus, Stooge sort correctly sorts A when $n = 1$ and when $n = 2$. The inductive case happens when $n = 3$. The inductive hypothesis is that Stooge sort correctly sorts arrays of $length < n$. We must show that Stooge sort correctly sorts arrays of length n.

The input array A is essentially split into three pieces: the first third and the last third are both of length k, and the second third is of length $n - 2k$, where $k = \lfloor(n/3)\rfloor$.

In line 6, Stooge sort is applied to the first two-thirds of A (i.e., the subarray of length n - k consisting of the first and middle thirds of A). We are assuming that $n >= 3$, so $k = \lfloor(n/3)\rfloor$ and $n - k$ is less than n. This means that we can invoke the inductive hypothesis on this subarray (i.e., we assume by inductive hypothesis that the first twothirds of A will be sorted in the first twothirds of the new sorted array). After line 6, the largest k of the first $2/3n$ will be in k2(see picture below).

```
___k1__ _ k2__ __k3__
-------|------|-------
Smaller larger
```

In line 7, Stooge sort is applied to the last two-thirds of A' (i.e., the subarray of length $n-k$ consisting $k2$ and $k3$). In this case $n$is $n-1/3n < n$ and by inductive hypothesis the last two-thirds of A' will be sorted correctly. After line 7, the largest k elements of the given array will be in the last third sorted ($k3$).Smallest elements from $k3$ would have moved to $k2$ (see picture below).

```
_k1_ ___k2__ __k3__
----|-------|------
____ Smaller Larger
```

In line 8, Stooge sort is applied to the first two-thirds again (k1 and k2 of length $n - k < n$) of the above sorted array. By inductive hypothesis on this subarray, it will be sorted correctly to the final array). After line 8, the smallest k elements will be in k1.

```
___k1__ __k2__  __k3__
-------|-------|-------
Smaller larger
```

Thus, final array returned by the algorithm, will consist of all the elements of A in sorted order. Most common mistakes: Everyone approach this problem correctly, some with more or less detail in the proof, but correct way if thinking.

Part B (3 points)

The recurrence is $T(n) = 3T(2n/3) + O(1)$ since we split arrays into three smaller parts each of size 2/3 of the original one. The non recursive portion of this algorithm is O(1). Solution is calculated with case 1 of Master Theorem: $A = 3, b = 3/2$ and $T(n) = \Theta(n^{\log_{3/2} 3})$. Computing the log we'll see that it is approximately 2.71.

Recursion tree solution: The depth of the tree is $\log_{3/2} n)$, at this depth there will be $n^{\log_{3/2} 3}$ nodes/leaves each contributing a cost of T(1) for a total of $n^{\log_{3/2} 3}T(1)$. The total cost of the tree will be dominated by leaves cost hence the solution for the recursion is $\Theta(n^{\log_{3/2} 3}$

Most common mistake: Computing the result as $\log_{2/3} 3$ not $\log_{3/2} 3$, thus the result turned out to be negative.

Part C (3 points)

$$n^4 <= O(T(n)), T(n) \leq O(n^e). \text{ So } n^4 <= cT(n)$$

we can substitute $T(n)$ for $n^{2.7095}$ and get the following:

$$n^d \leq c_1 n^{2.7095} \text{ and } n^{2.7095} \leq c_2 n^e$$

where c_1 and c_2 are some constants, we also have to follow the given condition that $e - d < 0.01$. We can pick the following values for d and e: $e = 2.7125, \quad d = 2.7020$

The values don't have to be exactly the ones given, they'll depend on how you rounded the result

Most common mistakes:

- to come up with negative values due to the fact that $\log_{2/3} 3$ was chosen instead of $\log_{3/2} 3$.
- d has to be less than the number and e larger.
- $e - d < 0.1$, not $e - d < 0.01$