

## Mysteries and Other Array Algorithms

### 1. A Mystery Procedure

Consider the following procedure that takes as input an array and two indices.

```
1: procedure MYSTERY(int[]  $A$ , int  $l$ , int  $r$ )
2:    $range := r - l + 1$ 
3:    $subrange := \lceil 2 \cdot range / 3 \rceil$ 
4:   if  $range = 2$  and  $A[l] > A[r]$  then
5:     SWAP( $A, l, r$ )
6:   else if  $range \geq 3$  then
7:     MYSTERY( $A, l, l + subrange - 1$ )
8:     MYSTERY( $A, r - (subrange - 1), r$ )
9:     MYSTERY( $A, l, l + subrange - 1$ )
10:  end if
11: end procedure
```

Note that division in line 3 is division of real numbers and recall that the ceiling function  $\lceil x \rceil$  returns the least integer that is greater or equal to  $x$ .

1. What effect does the call MYSTERY( $A, 1, A.length$ ) have on an array  $A$ ? Explain your answer.

**Hint:** Since MYSTERY is a recursive procedure, you need to provide an inductive argument, which goes as follows. Suppose you want to show that every array segment  $A[l..r]$  has the property  $P$  when MYSTERY( $A, l, r$ ) returns. Then you have to provide two arguments. First, you show that a segment having length 1 or 2 has property  $P$  when MYSTERY returns. Second, you assume that the segments  $A[l..l + subrange - 1]$ ,  $A[r - (subrange - 1)..r]$ , and  $A[l..l + subrange - 1]$  have property  $P$  if MYSTERY returns in lines 7, 8, and 9, respectively, and then, based on this assumption you show that  $A[l..r]$  has property  $P$ .

2. What is the asymptotic running time of MYSTERY?  
**Hint:** Use an appropriate technique from the lecture.

(8 Points)

## 2. Duplicate Elimination without Order Preservation

Develop an algorithm that eliminates duplicate occurrences from an array. More precisely, the algorithm should take as input an array of integers  $A$  and return an array  $B$  that contains the same values as  $A$ , possibly in a different order, such that no value in  $B$  occurs more than once.

1. Write pseudocode for a straightforward solution, which need not be efficient. Explain the idea of that algorithm.
2. Write pseudocode for an efficient algorithm and explain the idea of that algorithm, too.  
**Hint:** Again, an algorithm that you know from the lecture may be useful.
3. What is the asymptotic worst-case complexity of each of the two algorithms? Explain your answer.

(10 Points)

## 3. Duplicate Elimination with Order Preservation

Develop now an algorithm that eliminates duplicate occurrences from an array of integers in such a way that the first occurrence of each value remains while all the later ones are dropped. For example, if the input array is

$$A = [3, 5, 9, 4, 5, 11, 9, 3, 1, 1, 3, 4, 5, 7],$$

then the output array is

$$B = [3, 5, 9, 4, 11, 1, 7].$$

Your goal is to develop an *efficient* algorithm. Implement your algorithm in the class `DuplicateEliminationWithOrderPreservation`, which you find in the zip file `DSA_A5.zip`.

1. Develop JUnit tests for the algorithm, covering both special and general cases. Include one test for the example above.
2. Write pseudocode for an efficient algorithm and explain the idea of that algorithm.  
**Hint:** The algorithm for Question 2 may come in handy.

3. What is the asymptotic worst-case complexity of the algorithm? Explain your answer.
4. Implement your algorithm as a method `eliminate`.

(12 Points)

**Deliverables.**

1. For question 3, hand in the Java file that you wrote.
2. Write one report for all tasks.

Combine all deliverables into one zip file, which you submit via the OLE website of the course. Please, follow the “Instructions for Submitting Course Work” on the Web page with the assignments, when preparing your coursework.

Submission: Until Tue, 11 April 2017, 23:55 hrs, to the OLE submission page of

Lab A / Lab B / Lab C