Assignment 4

Julien Corman, Werner Nutt
Flavio Vella

Duplicate Eliminiation, Loop Invariants and Mysteries

Instructions. The coursework consists of 4 questions. The first, third and fourth require only conceptual answers, the second requires in addition a piece of Java code and JUnit tests. The conceptual answers must be submitted in PDF format via OLE. The code and the tests must be submitted via Codeboard and a copy of it via OLE.
The coursework counts 150%, since you have two weeks time to complete it.

1. 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: 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.

(Weight: 25% of this CW)

2. 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 task is to develop an *efficient* algorithm.

1. Develop JUnit tests for the algorithm. Use as template the test for the example above, available on the Codeboard project for this assignment.

2. Write pseudocode for an efficient algorithm and explain the idea of that algorithm.
   Hint 1: The algorithm for Question 1 may come in handy.
   Hint 2: Think about how you can remember which numbers in A have already been copied into B and therefore must not be copied a second time.

3. What is the asymptotic worst-case complexity of the algorithm? Explain your answer.

4. Implement your algorithm (as method `eliminate` of Class `Assignment3` in the Codeboard project for this assignment).

(Weight: 50% of this CW)

3. Loop Invariants

In this exercise we want to review loop invariants and how they can be used to understand algorithms.
Below is pseudocode for an algorithm that is supposed to check whether an array is sorted.

Input: Nonempty array $A[1..n]$ of integers
Output: TRUE if the array is sorted, FALSE otherwise

```
1: function CHECKSORTEDNESS(int[] A)
2:     n := A.length
3:     j := 1
4:     while j < n and A[j] ≤ A[j + 1] do
5:         j := j + 1
6:     if j = n then
7:         return TRUE
8:     return FALSE
```

Our goal is to show that CHECKSORTEDNESS does in fact check whether an array is sorted.

1. Write down a precise definition of the statement, "array A is sorted."

2. Formulate a loop invariant for the while loop of CHECKSORTEDNESS by which you can show that the algorithm in fact is checking sortedness.

3. Give arguments that your loop invariant holds when the algorithm reaches the while loop for the first time (initialization).

4. Give arguments that your loop invariant is maintained by each execution of the loop (maintenance).

5. Give arguments that the loop terminates (termination).

6. Give arguments that the answer TRUE is returned only if the array was sorted, and FALSE only if it was not sorted.

(Weight: 40% of this CW)

4. A Mysterious 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 := ⌈2 · range/3⌉
4:     if range = 2 and A[l] > A[r] then
5:         SWAP(A, l, r)
6:     else if range ≥ 3 then
7:         MYSTERY(A, l, l + subrange − 1)
8:         MYSTERY(A, r − (subrange − 1), r)
9:         MYSTERY(A, l, l + subrange − 1)
```

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$(A, 1, A.\text{length})$ 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 $A[l..l + \textit{subrange} - 1]$, $A[r - (\textit{subrange} - 1)..r)$, and $A[l..l + \textit{subrange} - 1]$ 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.

(Weight: 35% of this CW)

Deliverables. For Question 2, submit two copies of your code (and tests):

- one via Codeboard (instructions are available here),
- one via the OLE submission page of your lab (together with the other deliverables).

The other questions must be answered in a PDF document.
Combine all deliverables into one zip file, which you submit via the OLE submission page of your lab. Please include name, student ID and email address in your submission.

Submission until Thursday, 29 November 2018, 23:55, to Codeboard and the OLE submission page of:

Lab A / Lab B