Suggested solutions – Exam – Datastrukturer

DIT960 / DIT961, VT-18
Göteborgs Universitet, CSE

Day: 2018-10-12, *Time:* 8:30-12.30, *Place:* SB

Exercise 1

a) **For G:** $O(n \log n)$, the loop runs n times, and `member` + `insert` takes $O(\log n)$ time. The `append` operation takes amortised $O(1)$ time, so the sequence of n appends takes $O(n)$ time.

For VG: $O(n \log m)$, the set `S` always contains at most m elements.

b) Answer: $O(n)$ because hash table operations take (expected) $O(1)$ time.

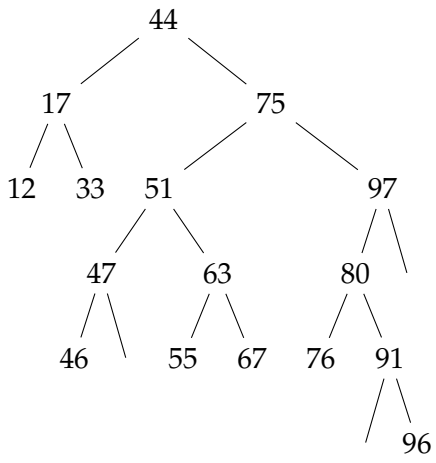
Exercise 2

a) The answers are B and D.

The idea is: if node x is an ancestor of node y, then node x must have been added before y.

In A, 67 is added before 76, but 76 is an ancestor of 67. In C and E, 67 is added before 63, but 63 is an ancestor of 67.

b) Here is the resulting binary tree (assuming you removed the greatest element from the left subtree):



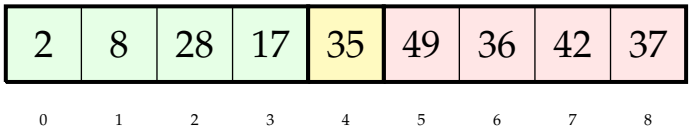
c) A possible solution

```
int search(Comparable[] array, Comparable key) {
  int low  = 0;
  int high = array.length - 1;

  while(low <= high) {
    int mid = (low + high) / 2;

    if (array[mid].compare(key) < 0) {
      // array[mid] < key
      low = mid+1;
    } else if (array[mid].compare(key) > 0) {
      // array[mid] > key
      high = mid-1;
    } else {
      // array[mid] == key
      return mid;
    }
  }
  return -1;
}
```

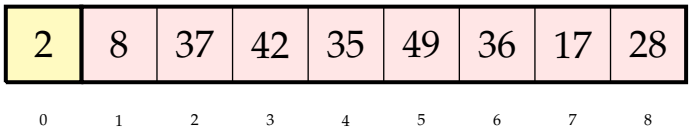
Exercise 3

It is important that all elements to the left of the pivot are *smaller* and to the right are *larger*, that is, the pivot should be in the right place. The elements in the to be sorted arrays should be in the correct order, reflecting how the quicksort algorithm works. The subarrays next to the pivot should *not* necessarily be sorted.

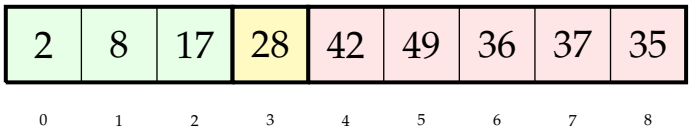
a)



b)



c)



For a VG:

```
sort :: [Bool] -> [Bool]
sort = rec 0
 where
  rec n []           = replicate n True
  rec n (True  : xs) = rec (n + 1) xs
  rec n (False : xs) = False : rec n xs
```

Exercise 4

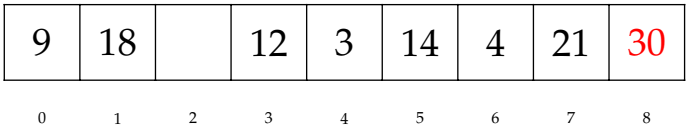
a) AD, BD, AC, CF, FG, EG, EH

b) H : 0, E : 1, G : 3, F : 6, B : 7, c : 7, D : 9, A : 9

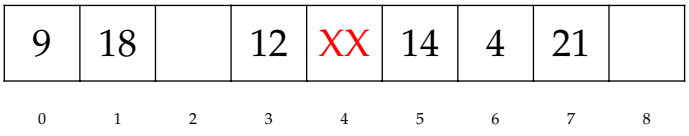
Exercise 5

a) The correct answers are: C and D.

b)



c)



Answer: 3 is replaced by a 'deleted' marker.

Exercise 6

For a G: sort the array, then add up the first k elements.

For a VG: use a min heap. The idea is the heap will contain the k largest elements we have seen so far.

Pseudocode:

```
h = new min-heap
for each x in array
  add x to h
  if the size of h is > k then
    delete min element from h
sum all elements of h
```