Tietorakenteet 2018
Harjoitukset 9, ratkaisut (Viikko 45)

1. Tiedostossa Sort.java kansiossa Problem 1.

2.-3. Tiedostossa Sort.java kansiossa Problem 2-3.

4. Algoritmi 1. Oletetaan, että taulukon indeksointi alkaa ykkösestä, ei nollasta kuten taulukoiden kohdalla yleensä. Operaatio *PercolateDown(A,i)* on valmiiksi käytettävissä, ja se vie indeksistä i löytyvää avainta keossa alaspäin vaihtamalla sen lapsensa kanssa, kunnes molemmat lapsista ovat pienempiä tai avain menee pohjalle.

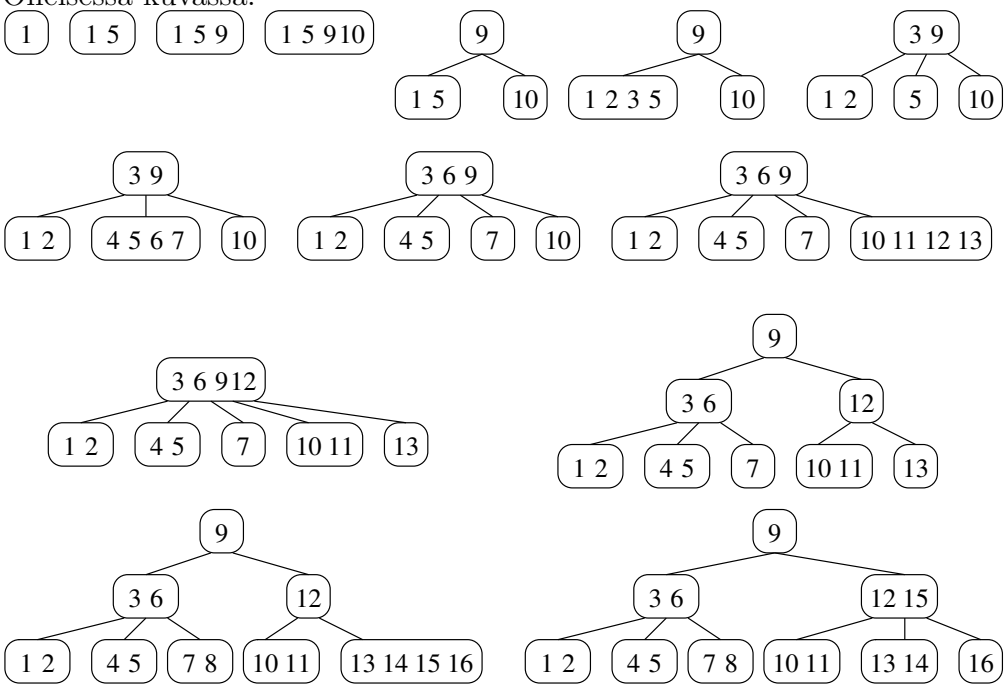
Algorithm 1 Lajittelee taulukon A nousevaan järjestykseen paikallaan toimivalla kekolajittelulla.

```
HeapSort(A)
  BuildMaxHeap(A)
  heapsize ← A.length
  for i ← A.length downto 2 do
    Swap(A[1],A[i])
    heapsize ← heapsize - 1
    PercolateDown(A,1)
  end for
```

Algorithm 2 Muodostaa maksimikeon taulukosta A paikallaan.

```
BuildMaxHeap(A)
  for i ← ⌊A.length/2⌋ downto 1 do
    PercolateDown(A,i)
  end for
```

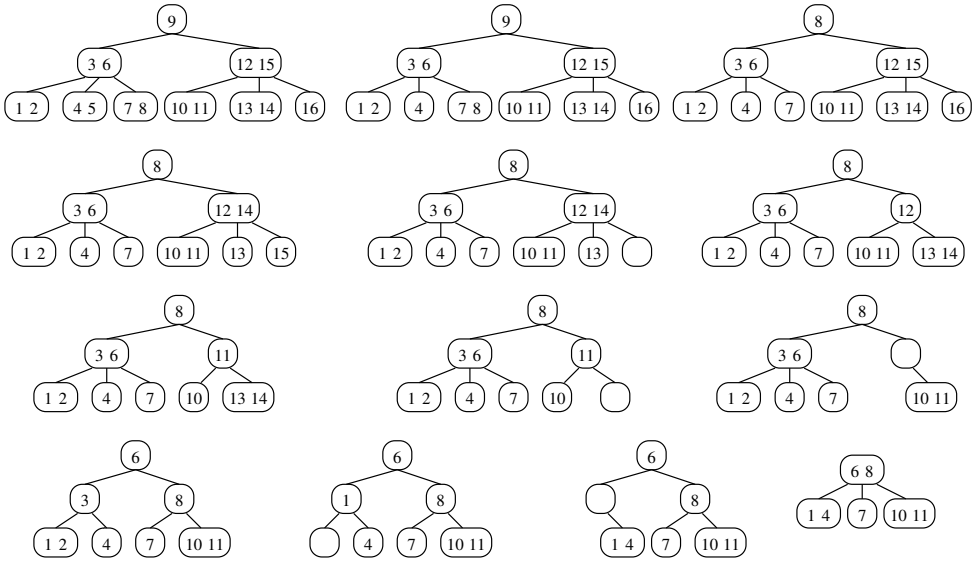
5. Oheisessa kuvassa.



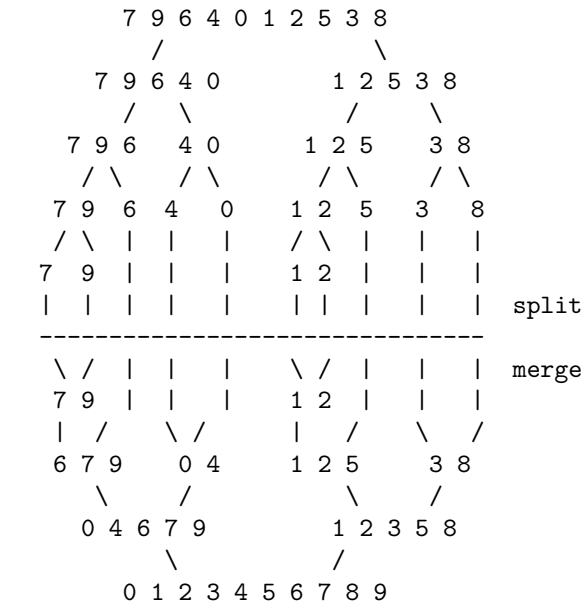
Selvennystä: (2,4)-puun solmussa on 1, 2 tai 3 avainta. Jos lisäyksen myötä solmussa olisi 4 avainta (ylivuoto), jaetaan solmu kahteen osaan. Solmun keskeltä nostetaan avain solmun vanhempaan, tai jos ei ole vanhempaa, luodaan uusi juuri ja avain nostetaan siihen (katso kuvan 1. rivi). Luentomonistetta mukaillen keskeltä valittu avain = 3. avain. Nostetun alkion vasemmanpuoleiset alkiot laitetaan yhteen solmuun, ja oikeanpuoleinen alkio toiseen solmuun. Nämä solmut liitetään vanhempaan nostetun avaimen vasemmalla ja oikella puolella oleviksi lapsisolmuiksi.

Jos nosto aiheuttaa ylivuodon, toistuu samanlainen jako-prosessi vanhemmassa (katso alkion 13 lisäys, kuvan riveillä 2-3).

6. Oheisessa kuvassa.



7. Lomituslajittelu



Pikalajittelu

```
7 9 6 4 0 1 2 5 3 8
                8    pivot (E)
7 6 4 0 1 2 5 3   9  L, G
      3
0 1 2   7 6 4 5
    2     5
0 1     4   7 6
  1         6
0             7
------------------
0 1 2 3 4 5 6 7 8 9
```

Lomituslajittelun aliohjelmakutsut. Split jakaa sekvenssin kahtia ja Merge lomittaa kaksi järjestettyä sekvenssiä.

```
MergeSort((7 9 6 4 0 1 2 5 3 8))
        Split((7 9 6 4 0 1 2 5 3 8)) => (7 9 6 4 0), (1 2 5 3 8)
        MergeSort((7 9 6 4 0))
                Split((7 9 6 4 0)) => (7 9 6), (4 0)
                MergeSort((7 9 6))
                        Split((7 9 6)) => (7 9), (6)
                        MergeSort((7 9))
                                Split ((7 9)) => (7), (9)
                                MergeSort((7)) => (7)
                                MergeSort((9)) => (9)
                                Merge((7), (9))
                        => (7, 9)
                        MergeSort((6)) => (6)
                        Merge((7,9),(6))
                => (6,7,9)
                MergeSort((4,0))
                        Split((4, 0)) => (4), (0)
                        MergeSort((4)) => (4)
                        MergeSort((0)) => (0)
                        Merge((0), (4))
                => (0, 4)
                Merge((6,7,9), (0,4))
        => (0, 4, 6, 7, 9)
        MergeSort((1 2 5 3 8))
                Split((1 2 5 3 8)) => (1 2 5), (3 8))
                MergeSort((1 2 5))
                        Split((1 2 5)) => (1 2), (5)
                        MergeSort((1 2))
                                Split ((1 2)) => (1), (2)
                                MergeSort((1)) => (1)
                                MergeSort((2)) => (2)
                                Merge((1), (2))
                        => (1, 2)
                        MergeSort((5)) => (5)
                        Merge((1,2),(5))
                => (1,2,5)
                MergeSort((3,8))
                        Split((3, 8)) => (3), (8)
                        MergeSort((3)) => (3)
                        MergeSort((8)) => (8)
                        Merge((3), (8))
                => (3, 8)
                Merge((1,2,5), (3,8))
        => (1, 2, 3, 5, 8)
        Merge((0, 4, 6, 7, 9), (1, 2, 3, 5, 8))
=> (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
```

Pikalajittelun aliohjelmakutsut. Partition jakaa sekvenssin sekvensseihin L, E, G (pivotina viimeinen alkio), Append liittää 3 sekvenssiä peräkkäin.

```
QuickSort((7 9 6 4 0 1 2 5 3 8))
        Partition((7 9 6 4 0 1 2 5 3 8)) => (7 6 4 0 1 2 5 3), (8), (9)
        QuickSort((7 6 4 0 1 2 5 3))
                Partition((7 6 4 0 1 2 5 3)) => (0 1 2), (3), (7 6 4 5)
                QuickSort((0 1 2))
                        Partition((0 1 2)) => (0 1), (2), ()
                        QuickSort((0 1))
                                Partition((0 1)) => (0), (1), ()
                                QuickSort((0)) => (0)
                                QuickSort(()) => ()
                                Append((0), (1), ())
                                => (0 1)
                        => (0 1)
                        QuickSort(()) => ()
                        Append((0 1), (2), ())
                        => (0 1 2)
                => (0 1 2)
                QuickSort((7 6 4 5))
                        Partition((7 6 4 5)) => (4), (5), (7 6)
                        QuickSort((4)) => (4)
                        QuickSort((7 6))
                                Partition((7 6)) => (), (6), (7)
                                QuickSort(()) => ()
                                QuickSort((7)) => (7)
                                Append ((), (6), (7))
                                => (6 7)
                        => (6 7)
                        Append ((4), (5), (6 7))
                        => (4 5 6 7)
                => (4 5 6 7)
                Append((0 1 2), (3), (4 5 6 7))
                => (0 1 2 3 4 5 6 7)
        => (0 1 2 3 4 5 6 7)
        QuickSort((9)) => (9)
        Append((0 1 2 3 4 5 6 7), (8), (9))
        => (0 1 2 3 4 5 6 7 8 9)
=> (0 1 2 3 4 5 6 7 8 9)
```