Data Structures 2018
Exercise 5, solutions (Week 41)

1.-3. The example solution can be found in LinkedList.java file located in the exercise folder of the course home page.

4. a) 7, 1, 0, 3, 2, 5, 4, 6, 9, 8, 10

   b) 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

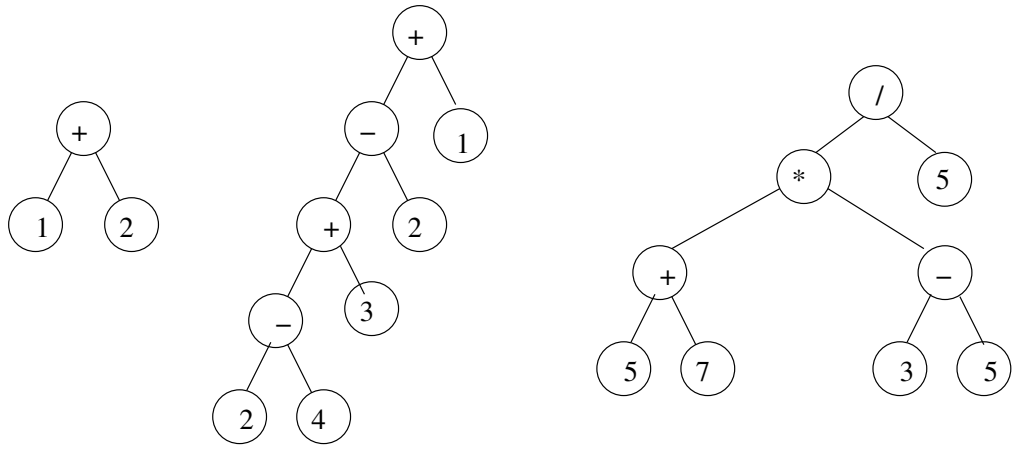
   c) 0, 2, 4, 6, 5, 3, 1, 8, 10, 9, 7

5. Algoritmi 1.

Algorithm 1 Printing binary tree in postorder with the help of stack

```
PrintPostOrder(n)
  Create new stack S
  S.push(n)
  repeat
    n ← S.top()
    if n.isInternal() then
      S.push(flagNode)
      S.push(n.rightChild)
      S.push(n.leftChild)
    else if n.isFlagNode then
      S.pop()
      n ← S.pop()
      Print n
    else
      Print n
      S.pop()
    end if
  until S is empty
```

6.



7. (a) Algorithm 2

   (b) Algorithm 3

   (c) Algorithm 4

Algorithm 2 Computes the number of nodes (root node is given as a parameter).

```
CountNodes(N)
  if N.isLeaf() then
    return 1
  else
    return 1 + CountNodes(N.leftChild) + CountNodes(N.rightChild)
  end if
```

Algorithm 3 Counts the number of leaf nodes (the root node is given as a parameter).

```
CountLeaves(N)
  if N.isLeaf() then
    return 1
  else
    return CountLeaves(N.leftChild) + CountLeaves(N.rightChild)
  end if
```

Algorithm 4 Computes the height of a tree.

```
Height(N)
  if N.isLeaf() then
    return 0
  else
    return 1 + max{Height(N.leftChild),Height(N.rightChild)}
  end if
```

8. Algorithm 5.

Algorithm 5 Calculates the number of nodes in the level k of a binary tree. r is the root node of the binary tree.

```
LevelNodes(k, r)
  if r = NULL then
    return 0
  else
    if k = 0 then
      return 1
    else
      return LevelNodes(k − 1, r.leftChild) + LevelNodes(k − 1, r.rightChild)
    end if
  end if
```