Data Structures 2018
Exercise 3 (Week 39)

- **Notice that based on university's new regulations on degrees, you can have a degree fail from the course which is put to the register.**

  If a student does not participate in the course and does not cancel his/her enrollment, or if he/she discontinues the course, he/she will be assigned a fail grade for the course in question.

- **Students who participate in exercise group must be in place before the exercise group begins (12.15/14.15/16.15). Students who come late do not get the exercise points.**
- **Check the numbers of exercises made before you come to the exercise group. By this means we can save a lot of time when filling the exercise point list.**
- **Remember to registrate for the Data Structures course and to the exercise group at the course webpage (`http://www.sis.uta.fi/~tira/`).**
- **Notice that pseudocode does not mean the same this as Java code. Pseudocode is not a programming language dependent presentation for an algorithm.**

1.-2. In this task you will need the following Java files: *Deque.java*, *DoubleLinkNode.java* and *Dequetest.java* (test program.) Implement additional operations `insertLast` and `removeLast` into the file Deque.java. `insertLast` operation adds a node at the end of the queue and `removeLast` removes and returns the last node from the queue. See how operations `insertFirst` and `removeFirst` are implemented. Test you implementation by uncommenting tests two and three from the test program.

3. Add operation `Reverse` into the file Deque.java of the previous assignment. `Reverse` operation reverses the order of the items in the queue. Manipulate the queue itself directly, so do not for example copy the items to another data structure. What is the time complexity of your implementation?

4. Find the complexity (big-$\mathcal{O}$) of the following algorithms without computing the number of primitive operations.

a)
```
s ← 0
for i ← 1 to n do
    s ← s + i
end for
```

b)
```
s ← 0
for i ← 1 to n do
    for j ← 1 to i do
        s ← s + i + j
    end for
end for
```

c)
```
s ← 0
i ← n
while i ≥ 1 do
    s ← s + i
    i ← i − 1
end while
```

d)
```
s ← 0
for i ← 1 to m do
    s ← s + i
end for
for i ← 1 to n do
    for j ← 1 to n do
        s ← s + i + j
    end for
end for
```

Remark. In d you cannot reason that $\mathcal{O}(m) + \mathcal{O}(n^2) = \mathcal{O}(n^2)$, why not ?

5. Let S be a stack and Q a queue. Depict their states after each of the given operation sequences. We assume that S and Q are empty in the beginning of each operation sequence.
   a) S.push(a); S.pop(); S.push(b); S.push (c);
   b) S.push(a); S.push(S.top()); S.push(b); S.push(S.pop()); S.push(S.top());
   c) Q.enqueue(a); Q.enqueue(b); Q.enqueue(c); Q.dequeue();
   d) S.push(a); Q.enqueue(b); Q.enqueue(S.top()); S.push(Q.dequeue()); S.pop(); S.push(Q.front());
6. Show in pseudocode how you could implement a queue by using stacks. That is, show pseudocode for the queue functions Enqueue(x) and Dequeue. You can assume that the stack functions Push(x) and Pop are known (ie. you can call them from your pseudocode without showing their implementation details). How many stacks does your stack-implementation of a queue need?

7. When measuring a noisy signal, for example the output of a positioning snesor, the effect of noise can be attenuated by applying the technique of moving average. Describe in pseudocode an algorithm using a **queue**, called MovingAverage, which computed the average of n consecutive values as efficiently as possible. For example, when $n = 2$ (that is to say, we want to compute the average of two consecutive values) and the input is signal "1 1 2 3 3", we get the output "0.5 1 1.5 2.5 3" assuming that only values "0" precede the first element of the input signal.
8. Describe in pseudocode an algorithm, which gets two nonnegative integers A and B expressed in linked lists as input and returns the sum of A and B in expressed in a list. The integers are expressed so that the first element of the list is the least significant digit of the integer. E.g. if $A = 986$, which expressed as a linked list is $(6, 8, 9)$ and $B = 103$ i.e. $(3, 0, 1)$ the the result is $986 + 103 = 1089$ so that the list $(9, 8, 0, 1)$ is returned.