Booklet no.:_____

ID: _____

M.Sc. Qualification exam - 2013

Date: 05.04.13
Time: 3 hours
Answer **4 out of 6 questions**

Important: In the following table, circle the numbers of the four questions you chose to answer

Question	Grade
1	/25
2	/25
3	/25
4	/25
5	/25
6	/25

Question 1: Data structures

Note: in this question you have to write some code. You can use a Java-like pseudo-code, without worrying about exact syntax. If you don't know the name or signature of some necessary method, you may invent them. Your code will be judged according to its elegance and efficiency.

a) A *word-level palindrome* is a sequence of words that reads the same from the beginning, going forward, and from the end, going backward. Spaces and punctuation characters are ignored. Here are three examples of word-level palindromes (the number of words in a palindromes can be either odd or even):

```
Herb the sage eats sage, the herb.
Escher, drawing hands, drew hands drawing Escher.
You know, I did little for you, for little did I know you.
```

The boolean method `isPalindrome(String str)` returns true if `str` is a word-level palindrome and false otherwise. Write an implementation of this method, using the services of the `Stack` and `StringTokenizer` classes whose API's are given below. You can use all the methods in the API's, or just some of them, as you see fit.

b) What is the running time of your implementation (use big *O* notation)?

c) Write an efficient implementation of the given `Stack` class API, using an array (meaning, a regular array of `String` elements, not a list). Note that the stack size is unlimited.

Class Stack

Represents a LIFO (Last In, First Out) data structure containing elements of type `String`.

Constructor:

`Stack()`	Constructs an empty stack.

Methods:

`boolean`	`empty()`	Tests if this stack is empty.
`String`	`peek()`	Returns the item at the top of this stack without removing it from the stack.
`String`	`pop()`	Returns the item at the top of this stack and removes it from the stack.
`void`	`push(String item)`	Pushes the item onto the top of this stack.

Class StringTokenizer

This class allows an application to break a string into tokens, skipping space characters and punctuation characters (like , . ; : etc.)

Constructor:

`StringTokenizer (String str)`	Constructs a string tokenizer for the specified string.

Methods:

`int`	`countTokens()`	Calculates the number of times that this tokenizer's `nextToken` method can be called before it generates an exception.
`boolean`	`hasMoreTokens()`	Tests if there are more tokens available from this tokenizer's string
`String`	`nextToken()`	Returns the next token from this string tokenizer. Space characters and punctuation characters are skipped.

Question 2: Algorithms

a) (15 pts.) Given is an undirected connected graph G=(V,E), with an even number of edges. Half of the edges have weight less than C (possibly with repetitions), and the other half of edges all have weight C. Is it true that every Minimum Spanning Tree (MST) of the graph includes the same number of edges of weight C?

If **YES**, describe an algorithm, as efficient as you can, that returns the number of edges of weight C in every MST of the graph. Explain its correctness and analyze its time complexity. **Note**: The output is a number between 0 and |E|/2, no need to return an MST.

If **NO**, describe a weighted graph fulfilling the above description (with your choice of C) that has at least two different MSTs each having a different number of edges of weight C.
Note: You need to describe the graph as well as the two MSTs.

b) (10 pts.) **Copy to your notebook the most correct answer and justify:** Given a flow network , if for an edge e it holds that increasing the capacity of e by 3 increases the value of the max-flow in the network by 2 then:

- e belongs to some min-cut in the network.
- e belongs to all min-cuts in the network.
- The network has a single min-cut and e belongs to it.

Question 3: Computer Networks

Suppose an ISP owns the block of addresses of the form **192.0.0.0/7** and she wants to create **eight** subnets from this block such that:

subnet 1 gets 1/2 of the address space,
subnet 2 gets 1/4 of the address space,
subnet 3 gets 1/8 of the address space,
subnet 4 gets 1/16 of the address space,
subnet 5 gets 1/32 of the address space,
subnet 6 gets 1/64 of the address space,
subnet 7 gets 1/128 of the address space,
subnet 8 gets 1/128 of the address space.

What are the prefixes (of the form a.b.c.d/x) for the eight subnets?
Write the addresses in the table. No need to justify the answer.

Starting with 192.0.0.0/7 :

Subnet	Space size	address -- a.b.c.d/x
(biggest) 1	1/2	
2	1/4	
3	1/8	
4	1/16	
5	1/32	
6	1/64	
7	1/128	
(smallest) 8	1/128	

Question 4: Operating System

The **producer-consumer problem** is a classic synchronization problem. The problem describes two processes, the producer and the consumer, who share a common, fixed-size buffer. The producer's job is to generate a piece of data, put it into the buffer and start again. At the same time, the consumer is consuming the data (i.e., removing it from the buffer) one piece at a time. The producer does not add data into the buffer if the buffer is full (instead the producer goes to sleep until there is an empty slot in the buffer) and the consumer does not remove data from an empty buffer (instead the consumer goes to sleep until the buffer is not empty).

The following code is a solution for the Producer – Consumer problem, for a warehouse of size 20 elements, which is initially empty. Assume in, out are variables that are initialized to zero.

Producer:	**Consumer:**

While (1) {
t = new item;
down(empty);
down(mutex);
buf[in++ % 20] = t;
up(mutex);
up(full);
}
	While (1) {
	down(full);
	down(mutex);
	t = buf[out++ % 20]
	up(mutex);
	up(empty);
	<use t>
	}

a) Fill in the initial values of the following semaphores:

1. Semaphore empty is initialized to _____
2. Semaphore full is initialized to _____
3. Semaphore mutex is initialized to _____

b) Fill in all the possible values of the following semaphores:

1. Semaphore empty _____
2. Semaphore full _____
3. Semaphore mutex _____

c) In each of the following sentences, circle the correct option in the bolded text, so that the sentence will be correct:

1. The instruction down(empty) is intended to **block / release** a process when the warehouse is **empty / not empty / full / not full**
2. The instruction up(full) is intended to **block / release** a process when the warehouse is **empty / not empty / full / not full**

3. The instruction down(full) is intended to **block / release** a process when the warehouse is **empty / not empty / full / not full**

4. The instruction up(empty) is intended to **block / release** a process when the warehouse is **empty / not empty / full / not full**

Question 5: Programming Languages

a) [15 pts] For each of the following Java code fragments, describe all the dynamic memory allocations and deallocations that occur when the fragment is executed. Assume the garbage collector runs immediately after the last line of the fragment. For each memory allocation, describe what object(s) are created, the order in which they occur and which line of code is the cause.

Assume the following definition for class Node:

```
class Node {
        public String value;
        public Node next;
}
```

1.

```
1 String x = "This is a test";
2 x = "This is another test";
```

2.

```
1 String x = new String("This is a test");
2 x = "This is another test";
```

3.

```
1 String[] x = new String[1];
2 x[0] = "This is a test";
3 x = null;
```

4.

```
1 String[] x = new String[1];
2 x[0] = new String("This is a test");
3 String y = x[0];
4 x = null;
```

5.

```
1 Node x = new Node();
2 x.next = new Node();
3 x = null;
```

6.

```
1 Node x = new Node();
2 Node y = new Node();
3 x.next = y;
4 y.next = x;
5 x = null;
```

b) [10 pts] Design and implement in a Java-like language a method to find if a linked list has a loop. Each node in the list is represented by a class of type Node (as described above). The next field points to the next node in the list, and is **null** if there are no more nodes in the list (in a list with a loop, there is no node with a null next field). The method should have the following signature:

```
boolean hasLoop(Node node);
```

It must work in time *O(n)* and make only *O(1)* memory allocations, where *n* is the number of nodes in the list. You may not assume that *n* is known (i.e., your method can't make use of *n*) and node can be any node in the list (not necessarily the first node). You may change the nodes as the method runs (**Hint**: you can use the next field to indicate that you have already seen a node).

No points will be deducted for syntax errors. Instead, what we are looking for in this question is elegant pseudo code.

Question 6: Computability and Complexity

a) Consider the language $L_1 = \{<M> | M \text{ is a TM that accepts the input string } 010\}$.

Is $L_1 \in R$?

Define $\overline{L_1}$.

Is $\overline{L_1} \in RE$?

b) For each of the following statements, answer **True**, **False** or **Open question** according to our current state of knowledge of complexity theory. Give **brief** reasons for your answers.

1. If $L_1 \leq_p L_2$ and $L_2 \leq_m L_3$ then $L_1 \leq_p L_3$
2. $NTIME(n^c) \subseteq DTIME(n^{2^c})$
3. EXP = coEXP
4. Every language that is NP-complete is coNP-complete
5. If $P = NP \cap coNP$ then P = NP
6. If NP = coNP then P = NP
7. $\overline{SAT} \in PSPACE$
8. P contains infinite languages

GOOD LUCK