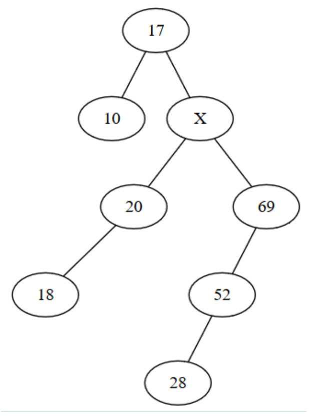
Example of an exam subject



1. Short problems (Some are multiple choice.) (2p – 4*0.5p).
   Answer the next questions and justify your answers.

1.1. Consider the binary search tree from the figure. What of the following values could be in the node with X?
   a) 40 b) 30 c) 19 d) 15 e) 26 f) 23

1.2. To which complexity class does the following expression belong to: $12*n^3 + n*\log_2 n + 52$?
   a) $\Omega(n^3)$ b) $\Omega(1)$ c) $O(n^4)$ d) $\Theta(n*\log_2 n)$ e) $\Theta(n^4)$ f) $O(n^2)$

1.3. Starting from an empty Stack we push into it, in the given order the following elements: 1, 2, 3, 4, 5, 6. Then we pop an element and push it into another, initially empty stack. We do this three more times (so we pop a total of 4 times). After this, we pop an element from the second stack. What value is now on the top of the second stack?
   a) 5 b) 4 c) 3 d) 1 e) 2 f) 6

1.4. What is the result of the following expression in postfix form made of single digit numbers and the regular operators (+, -, *, /): 4 6 7 + 1 - 2 5 * + +

2. Drawing questions (2.25p – 3*0.75p) . Answer the next questions and justify your answers.Starting from an initially empty binary search tree (built with the regular "≤" relation), insert into it, in the given order, the following elements: 41, 54, 60, 23, 73, 68, 98, 16, 13, 19, 36, 100, 76. It is enough to draw the final tree. Show the two possible trees after the removal of 41.

2.2. We have a hashtable with m = 8 positions in which separate chaining is used as a collision resolution method, but every position contains the root of a binary search tree. Show how the following elements can be added into this hashtable (which is initially empty): 8, 99, 40, 19, 56, 16, 17, 28, 62. It is enough to draw the final hashtable, but show how you computed the position for every element. Specify the load factor of the table.

2.3. Is the following array a binary heap? If not, transform it into a binary heap by swapping two elements:
   [41, 54, 13, 73, 68, 98, 63, 100, 76].
   In the (possibly modified) heap add the following elements (in this order): 19, 18, 59. After adding these 3 elements, remove an element from the heap. Draw the heap after every operation (4 drawings).

3. (2.5p) Design a data structure with the following properties:

- For search, insert and remove , it has the same complexity as in the case of a hash table
- The iteration order is the insertion order. Operations of the iterator are in Θ(1)

Requirements:

- Representation
- Draw the DS containing elements 12, 41, 73, 82, 64 , added in this order into an initial empty container.
- Specify and implement operation remove. Explain, in short, how this operation works.

4. (2.25p) Show how a queue can be implemented using two stacks. You can assume that the Stack ADT is already implemented. You will have to implement the Queue ADT using the operations of the Stack.

Requirements:

- Give the specifications for the operations of the Stack.
- Give the representation of the Queue.
- Specify and implement ALL Queue operations.
- Specify the complexity (BC, WC, total amortized) for every Queue operation, assuming Θ(1) complexity for the Stack operations.
- Show the content of the two stacks used to represent the queue after pusing 1 ,2 ,3 into an empty queue. Then after popping 1 element, then after adding element 4 and then after popping one more element. (4 drawings in total.)