Exam

Algorithms and Data Structures (INF1)

30.1.2006

Jiri Srba

Full name	
CPR number	
e-mail at cs.aau.dk	

Read first

- Start by filling in your name, CPR number, and e-mail address. Remember to put your name and CPR number also on any additional sheet of paper you will use.
- The exam consists of three questions. All questions are relative to the topics covered during the lectures and to the content of the main textbook by Jeffrey H. Kingston.

  You have 3 hours to answer the questions. It is recommended to spend approximately one hour per question. Make sure that your writing font is readable.

  - In Question 1 mark the correct answer out of the offered choices, or write the result directly into the highlighted box. For any calculations you need to do use a separate sheet of paper (this paper is not a part of the answer, do not hand it in). Do not provide any additional justification of your answers in this question.
  - In Questions 2 and 3 read carefully what you are asked to do. Make sure that you answer all the subquestions, and only those. Your answers should be clear and precise. Whenever you are asked to write an algorithm, use your favorite pseudo-code (the one used during the lectures is recommended but not required). It is also worth to write two or three lines describing informally what the algorithm is supposed to do.
- It is not allowed to use any electronic devices like laptops and mobile phones, except for a calculator. You can freely use your notes from the lecture, textbooks, course material from the folder, and copies of slides.

Question 1 (*In total 50 points*)

a) (*8 points, each correct answer +2, each wrong answer -1, no answer 0*)
Mark the correct answer by putting a cross in the corresponding field.

$n^8 + 0.01n^9$ is $\Theta(n^8)$	☐ True	☐ False
$\log n \cdot \log n + 7n \log n$ is $O(n \log n)$	☐ True	☐ False
$n^6 + 2^n$ is $O(2^n)$	☐ True	☐ False
$5n \log(n^6)$ is $O(n)$	☐ True	☐ False

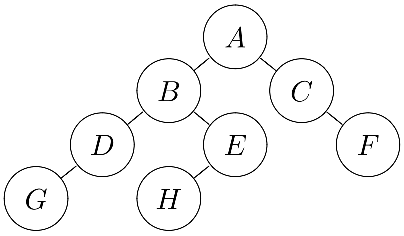
b) (*6 points, each correct answer +3, each wrong answer -2, no answer 0*)
Mark the correct answer by putting a cross in the corresponding field.

pre: $x \geq 0$ *and* $y = 0$
while $x > 4$ **do**
  $x := x + 1;\ y := y * x$
endwhile
post: $x \leq 4$ *and* $y = 0$

Is the algorithm totally correct? ☐ Yes ☐ No

Is the algorithm partially correct? ☐ Yes ☐ No

c) (*7 points, correct answer +7, wrong answer 0, no answer 0*)
Write the sequence of visited nodes in a postorder traversal of the binary tree.



, , , , , , ,

d) (*7 points, correct answer +7, wrong answer -2, no answer 0*)
What is the worst-case time complexity of the following algorithm (the input is a positive integer n)?
Mark the correct answer by putting a cross in the corresponding field.

```
x:=0;
for i:= 1 to n do
  for j:= 3 to n do
    for k:= 1 to 4987 do
      x:=x+k
    end for
  end for
end for
```

☐ $\Theta(n)$ ☐ $\Theta(n^2)$ ☐ $\Theta(n^2 \log n)$ ☐ $\Theta(n^3)$

e) (*7 points, correct answer +7, wrong answer 0, no answer 0*)
Write the sequence of values stored in a queue q after performing the following algorithm. The front of the queue is (as usual) on the left.

```
q: QUEUE; s: STACK; q.make; s.make
s.push(s.new_entry(1))
s.push(s.new_entry(2))
s.push(s.new_entry(3))
s.push(s.new_entry(4))
q.enqueue(s.pop)
s.pop
q.enqueue(s.pop)
q.enqueue(s.pop)
q.enqueue(q.dequeue)
```

, ,

f) (*8 points, correct answer +8, wrong answer 0, no answer 0*)
Draw a directed acyclic graph with nodes named A, B, C and D such that it has exactly two different topologically sorted sequences of nodes.

g) (*7 points, correct answer +7, wrong answer 0, no answer 0*)
One possible implementation of symbol table abstract data type is using binary search trees. Draw the resulting binary search tree (for nodes it is enough to write only the key values they store) after performing the following sequence of insert operations into a symbol table s.

```
s.make
s.insert(s.new_entry(8,"red"))
s.insert(s.new_entry(4,"blue"))
s.insert(s.new_entry(5,"orange"))
s.insert(s.new_entry(2,"black"))
s.insert(s.new_entry(7,"white"))
s.insert(s.new_entry(9,"yellow"))
```

Question 2 (*25 points*)

Consider the STACK abstract datatype storing integer values with the following (**restricted**) set of basic operations (note that there is no operation *pop*):

- *make*
  s.make initializes a stack s to empty
- *empty: BOOLEAN*
  s.empty returns *TRUE* if s is empty, otherwise it returns *FALSE*
- *new_entry(v: integer): ENTRY_TYPE*
  s.new_entry(v) returns a new entry of the stack s containing the integer v
- *push(x: ENTRY_TYPE)*
  s.push(x) inserts the entry x on the top of the stack s
- *top: ENTRY_TYPE*
  s.top returns the top entry of the stack s, the stack remains unchanged
- *remove*
  s.remove deletes the top entry of the stack s but does not return anything

In the following tasks when accessing a stack, **use only the operations** from the interface given above. We also assume that a given implementation of the abstract datatype STACK ensures that every operation listed above can be performed in time $O(1)$.

a) Write a sequence of stack operations which will create a new empty stack called s_1, create two new entries with values 2 and 3 and push them into s_1 such that the entry with the value 2 ends up on the top of the stack.

b) Write a function **do_pop(s: STACK): ENTRY_TYPE** which returns the entry stored on the top of a nonempty stack s and deletes the entry from the stack. Formally, we require the following.

   pre: $s = < a_1, a_2, \ldots, a_n >$, $n \geq 1$, *is a nonempty stack*

   $res :=$ do_pop(s)

   post: $res = a_n$ *and* $s' = < a_1, a_2, \ldots, a_{n-1} >$

   What is the time complexity of your algorithm?

c) Write a function **sum(s: STACK): integer** which returns the sum of the integers stored in all entries of the stack s. After finishing, the stack s should not be modified. Formally, we require the following.

   pre: $s = < a_1, a_2, \ldots, a_n >$, $n \geq 0$, *is a stack*

   $res :=$ sum(s)

   post: $s' = s$ *and* $res = a_1 + a_2 + a_3 + \cdots + a_n$

   What is the time complexity of your algorithm?
   Hint: you are allowed to use an additional data structure.

Question 3 *(25 points for a) – d) plus 10 extra points for e))*
In this question you will be solving the **John's Network Problem**.

Let us a consider a small hardware company called NetCom constructing computer networks. A customer owning a number of servers can ask the NetCom company for realizing physical connections between the servers. The connections are always bi-directional, which means that on every link between servers A and B, a communication is possible both from A and B as well as from B to A. A network is called connected if for any two servers there is the possibility to communicate with each other, possibly not directly but via a number of links.
NetCom's manager John is a good mathematician. Recently, a customer owning 4 servers arrived to the company and John could quickly answer that they are going to create 3 links to put the servers into a connected network. You shall now try to solve John's network problem by answering the following questions.

a) If a customer has n servers and wants to create a network such that all servers get connected, what is the minimum number of links that have to be created?

   In what follows we shall call this minimal number of links connecting n servers by ℓ_n. Hence for example $\ell_4 = 3$ because to connect 4 servers into a single network requires at least 3 links.

b) Suggest a mathematical formalization of the problem. What abstract datatype can be used to represent the problem? How do you represent servers in your abstract datatype and how do you represent the links?

The company NetCom was doing pretty well, but one day, a disaster happened. The workers creating a large network with n servers for one particularly important customer made a design error. Even thought they created exactly ℓ_n links (as they were told by John), the final network was not connected! Luckily, they immediately called John to tell him about the problem. John had a look at the created links and after some quick computation on his laptop, he was able to tell what went wrong and how to solve the problem.

Answer the following questions:

c) Show an example of a network of 4 servers using 3 links such that the resulting network is not connected (i.e. there is a pair of servers which cannot communicate). A drawing is sufficient.

d) As you could see, the reason why your example from question c) is not connected is because it contains a cycle. Write a general algorithm which for a given network with n servers and ℓ_n links decides, whether it contains a cycle or not.

   Make sure that you use correctly the interface of the abstract datatype you selected in part b). (If you are not able to do that, describe at least the ideas of the algorithm in words.)

   What is the worst-case time complexity of your algorithm?

e) Consider your algorithm from part d). In case that there is a cycle in the network, can you moreover tell which servers are on the cycle? If yes, how?