Name: Matriculation Number:

Midterm Exam
General CS II (320102)

April 08, 2014

You have 75 minutes(sharp) for the test;
Write the solutions to the sheet.

The estimated time for solving this exam is 68 minutes, leaving you 7 minutes for revising your exam.

You can reach 76 points if you solve all problems. You will only need 70 points for a perfect score, i.e. 6 points are bonus points.

Different problems test different skills and knowledge, so do not get stuck on one problem.

	To be used for grading, do not write here										
prob.	1.1	2.1	3.1	4.1	4.2	5.1	6.1	6.2	6.3	Sum	grade
total	10	12	9	12	10	12	2	5	4	76	
reached											

Please consider the following rules; otherwise you may lose points:

- Always justify your statements. Unless you are explicitly allowed to, do not just answer "yes" or "no", but instead prove your statement or refer to an appropriate definition or theorem from the lecture.
- If you write program code, give comments, so that we can award you partial credits!
- You may use tags in your $\mathcal{L}$(`VM`) program to save some (counting) time. Use <`string`> for tags, where `string` is a string of lower-case english letters and place the tag before the instruction one would want to jump to when calling `jp` or `cjp`. For a jump place the tag after `jp` and `cjp` and omit writing the relative jump distance.
- Write your program clearly. Should you wish, you may write additional code in a higher level language (HLL) as a comment to help the grader understand what you are trying to do. HLL code without $\mathcal{L}$(`VM`) code will not give you points.

1 Graphs and Trees

Problem 1.1 (k-connected graph)

Let a **k-connected graph** be a graph in which every vertex is an endpoint of exactly k edges. Prove or refute that in every k-connected graph there is a path of exactly k edges such that no edge is included twice. 10pt 10min

Solution: We will prove this by construction. We start from an arbitrary vertex V and choose an arbitrary edge that has not been chosen before. From that vertex, choose the next edge in the same way. Such an edge will always exist since the current path has less than k edges in it and every vertex has exactly k edges connected to it. Therefore we can always choose arbitrary edges until we fulfill the requirement that the path has length k.

Solution: We will prove this by construction. We start from an arbitrary vertex V and choose an arbitrary edge that has not been chosen before. From that vertex, choose the next edge in the same way. Such an edge will always exist since the current path has less than k edges in it and every vertex has exactly k edges connected to it. Therefore we can always choose arbitrary edges until we fulfill the requirement that the path has length k.

2 Combinatorial Circuits

Problem 2.1 (Line or Square)

You are given a screen with 9 pixels (3*3 layout). Each pixel can be turned on and off. Your task is to design a circuit that on a two bit input displays the corresponding decimal digits on the screen (i.e. turns the necessary pixels on). The circuit should have 9 outputs, one corresponding to each pixel. 12pt 12min

The following table gives the input/output relation:

input	00	01	10	11
output	1 1 1	0 1 0	1 1 1	1 0 0
	1 0 1	0 1 0	0 1 0	1 1 1
	1 1 1	0 1 0	1 1 1	0 1 0

3 Positional Number Systems

Problem 3.1 (Binary Number System Definitions)

9pt 8min

1. Given an $n+1$-bit binary number $a = a_n a_{n-1} \ldots a_0$, provide a mathematical definition of the following number representations, i.e. provide a formula for calculating what decimal number they represent:
   a) $\langle\langle a \rangle\rangle$
   b) $(\langle\langle a \rangle\rangle^-)$
   c) $\langle\langle a \rangle\rangle_{n+1}^{2s}$
2. Compute each of these three numbers for $a = 11100101$

Solution:

1. We just have the definitions from class:
   a) $\langle\langle a\rangle\rangle = \sum_{i=0}^{n} a_i \cdot 2^i$
   b) if $a_n = 0$, $(\langle\langle a\rangle\rangle^-) = \sum_{i=0}^{n-1} a_i \cdot 2^i$
   if $a_n = 1$, $(\langle\langle a\rangle\rangle^-) = -\sum_{i=0}^{n-1} a_i \cdot 2^i$
   c) $\langle\langle a\rangle\rangle_{n+1}^{2s} = -a_n \cdot 2^n + \sum_{i=0}^{n-1} a_i \cdot 2^i$
2. And after converting the numbers we get:
   a) $\langle\langle 11100101\rangle\rangle = 2^7 + 2^6 + 2^5 + 2^2 + 2^0 = 229$
   b) $(\langle\langle 11100101\rangle\rangle^-) = -\langle\langle 1100101\rangle\rangle = -(2^6 + 2^5 + 2^2 + 2^0) = -101$
   c) $\langle\langle 11100101\rangle\rangle_{n+1}^{2s} = \langle\langle 100101\rangle\rangle_{n+1}^{2s} = -2^5 + 2^2 + 2^0 = -27$

4 Machine Programming

Problem 4.1 (Simple ALU in $\mathcal{L}$(VM))

Given that a number a is stored in $S(0)$, another number b is in $S(1)$ and a control value *cmd* in $S(2)$ write some $\mathcal{L}$(VM) code which stores in $S(0)$ the ALU operation $RESULT$, where: 12pt 12min

$$RESULT = \begin{cases} a+b & \text{if } cmd = 1 \\ a-b & \text{if } cmd = 2 \\ a^2+b^2 & \text{if } cmd = 3 \\ -1 & \text{else} \end{cases}$$

Solution:

```
con 1, peek 2, sub,                  // pushing cmd − 1 on the stack
cjp 21                               if cmd == 1, jumps to CASE1
con 2, peek 2, sub,                  // pushing cmd − 2 on the stack
cjp 22                               if cmd == 2, jumps to CASE2
con 3, peek 2, sub,                  // pushing cmd − 4 on the stack
cjp 23                               if cmd == 4, jumps to CASE3
con -1, poke 0, halt,                // otherwise we store the error code −1
peek 1, peek 0, add, poke 0, halt,   // CASE1; stores a + b
peek 1, peek 0, sub, poke 0, halt,   // CASE2; stores a − b
peek 1, peek 1, mul, peek 0,
peek 0, mul, add, poke 0, halt,      // CASE3; stores a^2 + b^2
```

Problem 4.2 (Integer Division Rest in SW)

Write a SW program using Concrete Syntax, respective Abstract Syntax, which takes two decimal numbers a and b and computes the value of $a \bmod b$ (where mod represents the rest of integer division). For example, for $a = 512$ and $b = 5$, the output should be 2. 10pt 0min

Solution: Concrete syntax:

```
var a:=512; var b:=5;
while ( a>=b ) do
    a := a−b;
end;
return a;
```

Abstract syntax:

```
([("a", Con 512), ("b", Con 5)],
While( Leq(Var "b", Var "a"),
    Seq [
        Assign(Var "a", Sub(Var "a", Var "b")),
    ]
),
Var "a")
```

5 Turing Machines

Problem 5.1 (Formal Languages and Turing Machines)

Design a Turing Machine that decides whether a string is of the form $(001)^n$. You can assume that the head will be positioned at the first non-# character. The TM is supposed to end in the state "yes" if the input matches the form and in the state "no" otherwise. 12pt 12min

Solution:

0	#	#	r	s0
s0	0	0	r	s00
s0	1	1	r	no
s0	#	#	l	yes
s00	0	0	r	s1
s00	*	*	*	no
s1	1	1	r	s0
s1	*	*	*	no

6 Internet and the WWW

Problem 6.1 (Routers & NICs)

Suppose there are three routers between a source host A and a destination host B. How many network interface controllers does an IP datagram have to pass when sent from A to B? 2pt 3min

Solution: Eight: one each for A and B, and two each for each router (in and out).

Problem 6.2 (Data Transfer on the Internet)

Describe how data is transferred and encapsulated from one computer to another. Give a short description of the layers the data is passed through. 5pt 5min

Problem 6.3 (Cookies for eCommerce)

An eCommerce site wants to keep a purchases record for each of its customers. Describe/-explain how this can be done with cookies (explain the concept of cookies as you go along). 4pt 6min

Solution: Cookies are little files that the browser can (upon request by the web server) deposit on the local hard drive of the user. They can also be read by the web server.

For every transaction the web server reads its cookie, adds the transaction record and sends the cookie back to the browser for storage.