Name: Matriculation Number:

Midterm Exam 2
General CS I (320101)

November 12, 2013

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

The estimated time for solving this exam is 70 minutes, leaving you 5 minutes for revising your exam.

You can reach 109 points if you solve all problems. You will only need 70 points for a perfect score, i.e. 39 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	1.2	1.3	2.1	2.2	2.3	3.1	4.1	4.2	Sum	grade
total	15	5	6	15	8	10	15	20	15	109	
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!

1 GenCS Classics and Induction

Problem 1.1 (Induction on Addition Chains)

8pt

8min

Definition 1.1 The series $(a_0, a_1, a_2, \ldots, a_l)$, where $1 = a_0 < a_1 < a_2 < \ldots < a_l = n$ is called an **addition chain** of **length** l for $n \in \mathbb{N}$ iff for all $1 \leq i \leq l$ we have $a_i = a_j + a_k$ for some $j, k \in \mathbb{N}$ with $0 \leq j, k < i$.

Example 1.2 $(1, 2, 3, 5, 10, 15)$ is an addition chain for $n = 15$ of length $l = 5$ and $(1, 2, 3, 4, 7, 10, 14, 15)$ an addition chain for $n = 15$ with length $l = 7$.

Prove by induction or refute that the shortest addition chain for $n \in \mathbb{N}$ meets the condition $a_k \leq 2^k$ for all $k \in \mathbb{N}$ with $k \leq n$.

Solution:

Proof: by induction over k.

P.1 We have two cases:

P.1.1 Base case: $k = 0$:

P.1.1.1 $a_0 = 1$ by definition and $1 \leq 1$. □

P.1.2 Step case::

P.1.2.1 Assuming that $a_k \leq 2^k$ we have:

P.1.2.2 $a_{k+1} \leq a_k + a_k \leq 2^k + 2^k = 2 \cdot 2^k = 2^{k+1}$. □

P.2 This completes the induction. □

Problem 1.2 (Greek alphabet)

5pt

2min

Fill in the blanks in the following table of Greek letters. Note that capitalized names denote capital Greek letters.

Symbol		Φ		Γ		Σ		θ		χ
Name	zeta		pi		Psi		eta		Omega	

Solution:

Symbol	ζ	Φ	π	Γ	Ψ	Σ	η	θ	Ω	χ
Name	zeta	Phi	pi	Gamma	Psi	Sigma	eta	theta	Omega	chi

2 Sets, Relations and Functions

Problem 2.1 (Relations)

After taking too many maths courses, dr. Flipidon decided to revolutionize the world of lovers as a summer project and make the real-life relation called ”love” be an equivalence relation. 8pt 7min

However, he is still in the testing phase and has to make sure the following relation is an equivalence relation first:

Let $A := \{1, 2, 3, 4, 5\}$ and R a relation on A such that
$R := \{\langle 1,1\rangle, \langle 1,2\rangle, \langle 1,4\rangle, \langle 1,5\rangle, \langle 2,1\rangle, \langle 2,2\rangle, \langle 4,1\rangle, \langle 4,4\rangle, \langle 5,1\rangle, \langle 5,5\rangle\}$

Task: Is R an equivalence relation or not? Justify your answer and if it's not, adjust it such that it becomes an equivalence relation.

Solution:

- R is not relexive since $\langle 3,3\rangle \notin R$.
- An equivalence relation requires reflexivity, symmetry and transitivity.

$\Rightarrow$ R is not an equivalence relation since it is not reflexive.

- To make R an equivalence relation we could make it:

$$R' = R \cup \{\langle 3,3\rangle\} \setminus \{\langle 1,4\rangle, \langle 4,1\rangle, \langle 1,5\rangle, \langle 5,1\rangle\}$$

Problem 2.2 (Functions)

State in MathTalk: 6pt 6min

1. The definition of a partial function.
2. The definition of a total function.
3. The definition of an injective function.

Solution: See the slides.

---

3 Abstract Data Types and Abstract Procedures

Problem 3.1 (Abstract Procedures)

Even though the Us Government Shutdown is now over, you want to make sure it doesn't happen again so you and a group of your friends decide it is time to take over. You want to form your own underground party. The first thing you are going to need is a means to organize your party. The party will consist of a leader (who must be memorable and therefore have a name), some key helpers (they are not important enough to have a name, they do however need to be identifiable and therefore have a number) and followers (they don't need to be identifiable and you cant be expected to remember either names or numbers for them). 12pt 12min

We can model a party by the abstract data type

$$\mathcal{A} := \langle\{\mathbb{P}, \mathbb{H}, \mathbb{L}\}, \{[L\colon \mathbb{S} \to \mathbb{P}], [H\colon \mathbb{N} \to \mathbb{H}], [aH\colon \mathbb{P} \times \mathbb{H} \to \mathbb{P}], [aF\colon \mathbb{P} \to \mathbb{P}]\}\rangle$$

Where $\mathbb{P}$ is the sort of parties, $\mathbb{L}$ that of leaders, and $\mathbb{H}$ that of helpers. H creates a helper from a number ($\mathbb{N}$), and L a party from a leader's name ($\mathbb{S}$: strings). aH adds a helper to a party, and aF an (unidentified) follower.

1. Alas, we have forgotten to specify the sorts for names and numbers in $\mathcal{A}$. Create ADTs for numbers and strings (for simplicity, assume that the alphabet only has the 3 letters: s, m and l).
2. Create a ground constructor term for a party with *sml* the leader, 3 of her friends as helpers and 6 followers.

Now that you have *sml*s party organized, you want to know whether or not your party can emerge from the underground and take over yet. Your smartest scientists have discovered that a party can take over the US Government, if and only if it has at least twice as many followers as helpers.

3. Create an abstract procedure check (with signature $\mathbb{P} \to \mathbb{B}$, where $\mathbb{B}$ is the well-known sort of truth values) which returns "true" if your party can take over and "false" if not.
4. Now show the computation steps (and result) of your check procedure applied to the party you created previously

---

Hint: One possible solution: Use if and counting. You will need to define an if procedure yourself, but not doubling and comparison.

Solution:

1. $\langle\{\mathbb{N}\}, \{[z\colon \mathbb{N}], [s\colon \mathbb{N} \to \mathbb{N}]\}\rangle$ $\langle\{\mathbb{S}, \mathbb{L}\}, \{[s\colon \mathbb{L}], [m\colon \mathbb{L}], [l\colon \mathbb{L}], [es\colon \mathbb{S}], [al\colon \mathbb{L} \times \mathbb{S} \to \mathbb{S}]\}\rangle$
2. $aF(aF(aF(aF(aF(aF(aH(H(3), aH(H(2), aH(H(1), L(al(s, al(m, al(l, es)))))))))))))$
3. $\langle countF{::}\mathbb{P} \to \mathbb{N}\,;\, countF(aF(x)) \rightsquigarrow s(countF(x)), countF(aH(x)) \rightsquigarrow countF(x), countF(L(s)) \rightsquigarrow 0\rangle$

   $\langle countH{::}\mathbb{P} \to \mathbb{N}\,;\, countH(aH(x)) \rightsquigarrow s(countH(x)), countH(aF(x)) \rightsquigarrow countH(x), countH(L(s)) \rightsquigarrow 0\rangle$

   $\langle check{::}\mathbb{P} \to \mathbb{B}\,;\, check(x) \rightsquigarrow if(countF(x) \geq 2 * countH(x), T, F)\rangle$

   $\langle if{::}\mathbb{B} \times \mathbb{P} \times \mathbb{P} \to \mathbb{B}\,;\, if(T, x, y) \rightsquigarrow x, if(F, x, y) \rightsquigarrow y\rangle$
4. ...

---

4 Programming in Standard ML

Problem 4.1 (Arithmetic Expression Datatype)

Given the following SML data type for an arithmetic expressions 12pt 10min

```
datatype arithexp = aec of int (* constant 0,1,2,... *)
                  | aeadd of arithexp * arithexp (* addition *)
                  | aemul of arithexp * arithexp (* multiplication *)
                  | aesub of arithexp * arithexp (* subtraction *)
                  | aediv of arithexp * arithexp (* division *)
                  | aemod of arithexp * arithexp (* modulo *)
                  | aev of string (* variable *)
```

Write a (cascading) function eval : (string −> int) −> arithexp −> int that takes a variable assignment φ and an arithmetic expresson e and returns its evaluation as a value. Example:

```
exception unknown_identifier;

- fun phi("x") = 2
      | phi("y") = 3
      | phi(_) = raise unknown_identifier;
val phi = fn : string -> int

- (* 4x+3y *)
- eval phi (aeadd(aemul(aec(4),aev("x")),aemul(aec(3),aev("y"))));
val it = 17 : int
```

Note: A variable assignment is a function that maps variables to (integer) values, here it is represented as function φ of type string −> int that assigns $\varphi(n)$ to the variable aev(n).

Solution:

```
datatype arithexp = aec of int (* 0,1,2,... *)
| aeadd of arithexp * arithexp (* addition *)
| aemul of arithexp * arithexp (* multiplication *)
| aesub of arithexp * arithexp (* subtraction *)
| aediv of arithexp * arithexp (* division *)
| aemod of arithexp * arithexp (* modulo *)
| aev of string (* variable *)

(* aesub(aeadd(aemul(aec(4),aev(1)),aec(5)),aemul(aec(3),aev(1))) *)

fun eval phi =
let
    fun calc (aev(x)) = phi(x) |
        calc (aec(x)) = x |
        calc (aeadd(e1,e2)) = calc(e1) + calc(e2) |
        calc (aesub(e1,e2)) = calc(e1) - calc(e2) |
        calc (aemul(e1,e2)) = calc(e1) * calc(e2) |
        calc (aediv(e1,e2)) = calc(e1) div calc(e2) |
        calc (aemod(e1,e2)) = calc(e1) mod calc(e2);
in fn x => calc(x)
end;
```

```
(* Test:
- eval (fn 1=>6) (aesub(aeadd(aemul(aec(4),aev(1)),aec(5)),aemul(aec(3),aev(1))));
stdIn:14.7-14.14 Warning: match nonexhaustive
1 => ...

val it = 11 : int
- *)
```

Problem 4.2 (Longest Nondecreasing Subsequence)

15pt

15min

1. Design an SML function that takes a list L and returns a list containing all the subsequences of L. Signature and example:

   ```
   val subSeq = fn : 'a list -> 'a list list
   - subSeq [1,2,3];
   val it = [[1,2,3],[1,2],[1,3],[1],[2,3],[2],[3],[]] : int list list
   ```

2. Now design an SML function which takes as argument a list L of integers. The function returns the longest nondecreasing subsequence in L.

   ```
   val LNSS = fn : int list -> int list
   - LNSS([2,4,3,3,6,8,7,7]);
   val it = [2,3,3,6,7,7] : int list
   ```

Hint: You might find map and List.filter useful.

Solution:

```
Control.Print.printLength := 1000;

fun append x ll = map (fn ls => x :: ls) ll;

fun subSeq [] = [[]]
    | subSeq (h :: t) = let val ps = subSeq t in append h ps @ ps end;

fun is_non_dec(nil) = true
| is_non_dec(x::nil) = true
| is_non_dec(x::y::ls) = (x<=y) andalso is_non_dec(y::ls);

fun max([]) = []
| max(x::ls) = let val m=max(ls) in if length(x)>length(m) then x else m end;

fun LNSS(ls) = max(List.filter is_non_dec(subSeq(ls)));
```

---

5 Formal Languages and Codes

Problem 5.1 (Codes)

Given the following mapping from characters to binary strings:

10pt

8min

a = 00001	b = 01010	c = 10100	d = 00100	e = 1110	f = 01001
g = 00101	h = 1001	i = 10111	j = 01111	k = 10101	l = 00000
m = 01100	n = 01000	o = 110	p = 01101	q = 10110	r = 11111
s = 00010	t = 1000	u = 11110	v = 00011	w = 01011	x = 01110
y = 00111	z = 00110				

1. Is this a character code?
2. Is this a prefix code?
3. Decode the string 0100011010000101000000100100 into the english alphabet.

For the first two tasks say what you are checking and restrict yourselves to the characters a to l.

Solution:

1. Yes, it is a valid code since every letter gets mapped to a different one.
2. Yes, it is a prefix code since no character is a prefix of any other one.
3. The decoded string is "notbad"