Name: Matriculation Number:

Midterm Exam
General CS I (320101)

October 19, 2009

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

The estimated time for solving this exam is 62 minutes, leaving you 3 minutes for revising your exam.

You can reach 45 points if you solve all problems. You will only need 40 points for a perfect score, i.e. 5 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	3.1	3.2	4.1	Sum	grade
total	3	5	5	5	8	5	10	4	45	
reached										

Good luck to all students who take this test

1 Mathematical Foundations

3pt
2min

Problem 1.1: Fill in the blanks in the table of Greek letters. Note that capitalized names denote capital Greek letters.

Symbol					γ	Σ	π	Φ
Name	alpha	eta	lambda	iota				

Solution:

Symbol	α	η	λ	ι	γ	Σ	π	Φ
Name	alpha	eta	lambda	iota	gamma	Sigma	pi	Phi

Problem 1.2 **(Properties of Relations)** 5pt 8min

You are given the following definitions:

An irreflexive relation is a relation in which no element is related to itself.

An asymmetric relation R on a set S is a relation which has no symmetric pairs. If $\langle a, b\rangle \in R$ there exists no $\langle b, a\rangle \in R$ with $\forall a, b.S$.

Your tasks:

a. Find a symmetric, an asymmetric and an antisymmetric relation on the set $\{1, 2, 3, 4, 5\}$.

b. Prove that a relation is asymmetric if and only if it is both antisymmetric and irreflexive.

Solution:

a. Symmetric: $R_1 := \{\langle 1, 2\rangle, \langle 2, 1\rangle\}$;

Antisymmetric: $R_2 := \{\langle 1, 3\rangle\}$;

Asymmetric: $R_3 := \{\langle 2, 3\rangle\}$;

b. "$\rightarrow$" The relation is asymmetric implies that there are no two elements $a, b \in S$ for which there exist both $\langle a, b\rangle$ and $\langle b, a\rangle$ in R (even if $a = b$). This covers the proof for antisymmetry, which is defined as a relation in which there cannot exist both $\langle a, b\rangle$ and $\langle b, a\rangle$ in R except for the case in which $a = b$. Hence antisymmetry is implied by asymmetry.

Since the relation is asymmetric, if $\exists a.\mathbb{R}$ such that $\langle a, a\rangle \in R$, then this would contradict the given definition. Then $\forall a \in S.\langle a, a\rangle \notin R$, hence asymmetry also implies irreflexivity.

"$\leftarrow$" If a relation is assymetric it is implied that there are no symmetric pairs of reals $\langle a, b\rangle \in R$ and $\langle b, a\rangle \in R$ except for the case when $a = b$. However, because we have irreflexivity, we cannot have $\langle a, a\rangle \in R$ for all $a \in S$. This leads us the definition of assymetry.

Problem 1.3 (Induction on Addition Chains) 5pt 8min

Following is the definition of Addition Chain:

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 for $n \in \mathbb{N}$ if and only if $\forall i \mathbb{N} 1 \leq i \leq l$ and there are $j, k \in \mathbb{N}$ such that $0 \leq j$, $k < i$ and $a_i = a_j + a_k$.

We will call l the length of the chain.

For example: $(1, 2, 3, 5, 10, 15)$ is an addition chain for $n = 15$ of length $l = 5$. $(1, 2, 3, 4, 7, 10, 14, 15)$ would also be an addition chain for $n = 15$ with length $l = 7$.

Your Task:

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

Solution:

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

Induction is complete. □

---

2 Abstract Data Types and Abstract Procedures

Problem 2.1 (ADT for sentences)

5pt
6min

A *sentence* is a (possibly empty) list of **non-empty** words over the alphabet $\{a, b, \ldots, z\}$

Create an Abstract Data Type for *sentences*. Start with the constructor declarations $[a : Letter], [b : Letter], \ldots, [z : Letter]$. Make sure that any sentence has a *unique* representation in your ADT.

Don't forget that the words are non-empty, but the sentence can be empty.

Solution:

$\langle\{Letter, Word, Sentence\}, \{[a\colon Letter], [z\colon Letter], [baseW\colon Letter \to Word], [stepW\colon Letter \times Word \to Word],$

Problem 2.2 (Addition on integers) 8pt 10min

In the General Computer Science Exam, John was given the following abstract datatype for extended natural numbers.

$\langle\{N\},\{[o:N],[s:N->N],[n:N->N]\}\rangle$,

Where n is the negative of a natural number and s is the successor operation. He was then asked to come up with an abstract procedure for simple addition on all possible combination of numbers present in this datatype.

John gave the following solution:

{ add(x,o)→x, add(o,x)→s(x), add(x,s(y))→s(add(x,y)), add(s(x),n(s(y)))→add(x,n(y)), add(n(s(x)),s(y))→add(n(x),y), add(x,o)→x, add(x,o)→x }

He was awarded 3 out of 6 points for this answer.

Please provide all the possible errors in his solution and also all other reasons for this point deduction.

Hint: Hint : Verify the correctness, completeness and accuracy of his solution

---

Also give the complete and correct version of this procedure and then,

Analyze its Termination Properties.

Represent numbers 4 and -2 in this datatype and compute their sum using the given procedure (show evey step of the computation).

Solution: Errors in this datatype:

Missing conditions:

```
add(n(x),n(y))->n(add(x,y)) ,
add(n(o),y)->y ,
add(x,n(o))->x ,
```

Error:

add(o,x)→s(x) should be add(o,x)→x

Syntax:

The procedure definition add: N*N→N is missing.

New procedure terminates on all positive negative inputs.

$4 = s(s(s(s(o))))$ and $-2 = n(s(s(o)))$.

Addition would be:

```
add(s(s(s(s(o)))) , n(s(s(o)))) = add(s(s(s(o))) , n(s(o)))
                                = add(s(s(o)) , n(o))
                                = s(s(o))
```

---

3 Programming in Standard ML

Problem 3.1 (SML types)

5pt
8min

Determine the signatures of the following SML functions:

```
fun f1 (a,b,c,d) = [a]@(b::c)@d;
fun f2 (a,c) = fn (l) => l(a(c)) + a(c);
fun f3 (a)(b,c) = c(foldl(a)(b));
```

An example signature is:

```
- fun example(n,m) = (n,m);
val example = fn : 'a * 'b -> 'a * 'b
```

Hint: The signature of `foldl` is: `val foldl = fn : ('a * 'b -> 'b) -> 'b -> 'a list -> 'b`

Solution:

```
val f1 = fn : 'a * 'a * 'a list * 'a list -> 'a list
val f2 = fn : ('a -> int) * 'a -> (int -> int) -> int
val f3 = fn : ('a * 'b -> 'b) -> 'b * (('a list -> 'b) -> 'c) -> 'c
```

Problem 3.2 (Another Number 4) 10pt 15min

It is known that each natural number can be obtained from number 4 and applying the following operations:

A - add 4 at the end of the number

B - add 0 at the end of the number

C - the current number is divided by 2

A GenCS student coded the room numbers of his colleagues using sequences of characters (A, B, C), which are representations of the operations needed to obtain the number from 4. Notice how these numbers are built (mind the operation order):

$$120 \rightsquigarrow \mathrm{B(C(A(C(4))))} \rightsquigarrow \text{"BCAC"}: \begin{cases} 4/2 = 2 \rightsquigarrow \mathrm{C} \\ 2 \leftarrow 4 = 24 \rightsquigarrow \mathrm{A} \\ 24/2 = 12 \rightsquigarrow \mathrm{C} \\ 12 \leftarrow 0 = 120 \rightsquigarrow B \end{cases}$$

Your task is to help your friends by decoding their room numbers.

1. Write a SML function `decode` that takes a list of tuples containing the name of the student and his/her coded room number. Your function should output the updated list containing names and correct room numbers. If a coded string contains other characters than the ones specified, raise exception Undefined.

   ```
   val decode = fn : (string * string) list -> (string * int) list
   ```

2. Create a `lookup` function where, given a name and the initial list, displays his/her correct room number. If the name is not in the list, raise exception `NotFound`.

   ```
   val lookup = fn : (string * string) list -> string -> int
   ```

Note: You can use functions from previous parts in subsequent parts without defining them. Examples:

```
val l = [("Tom", "CCAB"), ("Werner", "ABC")];
decode l;
val it = [("Tom",101),("Werner",204)] : (string * int) list

val l1 = [("Lena", "ACAXX")];
decode l1;
uncaught exception Undefined

lookup l "Tom";
val it = 101 : int

lookup l "Jerry";
uncaught exception NotFound
```

Solution:

```
exception Undefined;
exception NotFound;
```

```
fun convert([])   = 4
| convert(#"A"::t)  = 4 + 10 * convert(t)
| convert(#"B"::t) = 10 * convert(t)
| convert(#"C"::t)  = convert(t) div 2
| convert(_::t)  = raise Undefined;

fun pair(a:string, b:string) = (a, convert(explode(b)));

fun decode [] = []
| decode list = map pair list;

fun lookup1([], name)   = raise NotFound
| lookup1((a,b)::t, name) = if name = a then b else lookup1(t, name);

fun lookup list name = let val
                             l = decode(list) in
                               lookup1(l, name)
                       end;
```

4 Formal Languages

Problem 4.1 (Lexical Order) 4pt 5min

Let $A := \{c, o, m, p, u, t, e, r\}$ be an alphabet and $L := \{cute, more, rum\}$, $M := \{cut, root, etc\}$ formal languages in A. Let $<$ with $r < e < t < u < p < m < o < c$ be a lexical order on A. Order the words in $L \cup M \cup \mathrm{conc}(L, M)$ in the lexical order $<_{lex}$ induced by $<$.

Solution: $Rum < rumroot < rumetc < rumcut < root < etc < more < moreroot < moreetc < morecut < cut < cute < cuteroot < cuteetc < cutecut$