Name: Matriculation Number:

Midterm Exam
General CS I (320101)

October 31, 2011

You have 70 minutes(sharp) for the test;
Write the solutions to the sheet.
The estimated time for solving this exam is 68 minutes, leaving you 2 minutes for revising your exam.
You can reach 102 points if you solve all problems. You will only need 90 points for a perfect score, i.e. 12 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	2.1	2.2	3.1	3.2	4.1	4.2	4.3	Sum	grade
total	5	15	10	15	12	10	10	15	10	102	
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

5pt

Problem 1.1 (Greek Letters) 2min

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

Symbol		Ξ	η			Λ		ω	I	
Name	delta			sigma	Psi		Omega			chi

Solution:

Symbol	δ	Ξ	η	σ	Ψ	Λ	Ω	ω	I	χ
Name	delta	Xi	eta	sigma	Psi	Lambda	Omega	omega	Iota	chi

Problem 1.2 (Inductive tuples) 15pt 10min

You are given n-tuples $\langle x_1, x_2, \ldots, x_n\rangle$ in which $\forall i.x_i = \pm 1$, and the transformation rule

$$\langle x_1, x_2, \ldots, x_n\rangle \longmapsto \langle x_1x_2, x_2x_3, \ldots, x_nx_1\rangle.$$

Prove by induction or refute that if $n = 2^k$ and $k \geq 1$, then after some finite number of steps, an n-tuple consisting exclusively of 1's ($\langle 1, 1, \ldots, 1\rangle$) will be obtained.

Hint: For the step case of the proof, look at what is happening to the tuple after applying an even number of transformations. You may want to write down the first several transformations in order to observe what is conserved and what changes.

Solution:

Proof: We will prove that for $n = 2^k$, after a certain number of steps the tuple $\langle 1, 1, \ldots, 1\rangle$ will be obtained.

P.1 For the induction we have to consider the following cases:

P.1.1 $k = 1$**, i.e.** $n = 2$:

P.1.1.1 We have the following possibilities:

- $\langle 1, 1\rangle \longmapsto \langle 1, 1\rangle$
- $\langle 1, -1\rangle \longmapsto \langle 1, 1\rangle$
- $\langle -1, 1\rangle \longmapsto \langle 1, 1\rangle$
- $\langle -1, -1\rangle \longmapsto \langle 1, 1\rangle$

Therefore in each case after one application of the transformation we get the desired result. □

P.1.2 $k > 1$:

P.1.2.1 Now, assume that the assertion is true for a certain value of k, and respectively for a certain value of n, i.e. that taking the tuple $\langle x_1, x_2, \ldots, x_n\rangle$ after a certain number of steps we obtain $\langle 1, 1, \ldots, 1\rangle$.

P.1.2.2 Next, we take $k + 1$ and $n = 2^{k+1}$.
We observe that since $\forall i.x_i = \pm 1$, then $\forall i.x_i^2 = 1$.

If we apply two transformations, we get

$$\begin{aligned}
\langle x_1, x_2, \ldots, x_{n+1}\rangle &\longmapsto \langle x_1x_2, x_2x_3, \ldots, x_{n+1}x_1\rangle \\
&\longmapsto \langle x_1x_2^2x_3, x_2x_3^2x_4, x_3x_4^2x_5, \ldots, x_nx_{n+1}^2x_1, x_{n+1}x_1^2x_2\rangle \\
&= \langle x_1x_3, x_2x_4, x_3x_5, \ldots, x_nx_1, x_{n+1}x_2\rangle
\end{aligned}$$

This last tuple can be obtained by alternating the entries of the two n-tuples $\langle x_1x_3, x_3x_5, \ldots, x_nx_1\rangle$ and $\langle x_2x_4, x_4x_6, \ldots, x_{n+1}x_2\rangle$.

We observe that continuing doing the transformations, on each even-numbered transformation, we have a separation of the entries, i.e. they are taken alternatively from a transformation on the above two tuples. But they are n-tuples, i.e. the inductive assumption holds for them. Therefore after a certain number of steps they will contain only 1's, and therefore after the whole $(n+1)$-tuple will also contain only 1's. □

□

---

2 Relations and Functions

10pt

Problem 2.1 (Relation Properties) 7min

- You are given the set $A := \{1, 2, 3, 4\}$ and the relation

$$R \subseteq A \times A, \quad R := \{\langle 1,1\rangle, \langle 3,1\rangle, \langle 2,4\rangle, \langle 2,2\rangle, \langle 2,3\rangle, \langle 4,4\rangle, \langle 1,4\rangle, \langle 3,3\rangle\}$$

  Determine whether R is reflexive, symmetric, transitive or antisymmetric. If the relation does not have a certain property, give a counter-example to show that.
- How many such relations (with the determined properties of R) are there over the set A? How many would they be for a set A with $\#(A) = n$?

Solution:

1. The relation is obviously reflexive and antisymmetric according to the definitions.
   It is not symmetric since, e.g. it contains $\langle 3, 1\rangle$, but not $\langle 1, 3\rangle$.
   It is not transitive because, e.g. it has $\langle 3, 1\rangle$ and $\langle 1, 4\rangle$, but not $\langle 3, 4\rangle$.

2. Thus the relations we are looking for have to be reflexive and antisymmetric.
   For A, since it has to be reflexive, there are $16 - 4 = 12$ pairs to choose from. If we choose one of those pairs, then we must not choose its symmetric part, so we are left with 6 options. Every subset of them will give a relation that is reflexive and antisymmetric. So in total there are 2^6 relations.
   Following the same logic for a set with cardinality n, we get $2^{\frac{n^2-n}{2}}$ relations.

Problem 2.2 (Function classics and injective functions) 15 min

- Let A and B be sets. State the definition of the concept of a **partial function** with domain A and codomain B. Also state the definition of a **total function** with domain A and codomain B.
- Let $n \geq 1$ be a natural number and let $f\colon \{1, 2, \ldots, 2n\} \to \{1, 2, \ldots, 2n\}$ be an injective function such that:

$$f(1) + f(3) + \cdots + f(2n-1) = f(2) + f(4) + \cdots + f(2n)$$

  1. Give an example of such a function f for $n = 4$.
  2. Prove that n is divisible by 2.

Solution:

- Let A and B be sets, then a relation $R \subseteq AB$ is called a **partial/total function**, iff for each $a \in A$, there is at most/exactly one $b \in B$, such that $\langle a, b\rangle \in R$.
- As f is injective, it means that the values of f are a permutation of the set $\{1, 2, \ldots, 2n\}$. Denote by S the sum on both sides of the equation given in the problem formulation. Then we have $S + S = f(1) + f(2) + \cdots + f(2n) = 1 + 2 + \cdots + 2n = n(2n+1)$, so 2 must divide n.
  An example function would be $f(1) = 1$, $f(2) = 2$, $f(3) = 3$, $f(4) = 4$, $f(5) = 6$, $f(6) = 5$, $f(7) = 8$ and $f(8) = 7$.

---

3 Abstract Data Types and Abstract Procedures

Problem 3.1 (Rowing Boats) 12pt 8min

1. Design an abstract data type with sort $\mathbb{B}$ for rowing boats. There are two variations of a rowing boat:
   - **Sculling boat** – In this boat, there is an **arbitrary** ($n \geq 1$) number of rowers, each of which has two oars, one on **each** side.
   - **Sweeping boat** – In this boat, there is always an **even** ($2n \geq 2$) number of rowers, each of which rows with one oar on either **left** or **right** of them in an alternating manner.

   Note that every rowing boat can optionally have **one coxswain** – a person who sits in the boat without rowing and just steers it.
   Make sure to use four different sorts for every possible function of a person in the boat (two oars, one oar on left, one oar on right, coxswain).
   Also, feel free to define additional sorts as needed.

2. Now represent the following boats using your ADT:
   (a) a sculling boat with two rowers and without a coxwain
   (b) a sweeping boat with four rowers and with a coxwain

Solution:

1.
$$\langle \{\mathbb{E}, \mathbb{L}, \mathbb{R}, \mathbb{C}, \mathbb{S}_c, \mathbb{S}_w, \mathbb{B}\}, \left\{ \begin{array}{l} [each\colon \mathbb{E}], [left\colon \mathbb{L}], [right\colon \mathbb{R}], [coxswain\colon \mathbb{C}], \\ [sc\colon \mathbb{E} \to \mathbb{S}_c], [addSc\colon \mathbb{S}_c \times \mathbb{E} \to \mathbb{S}_c], [coxSc\colon \mathbb{S}_c \times \mathbb{C} \to \mathbb{B}], \\ [finSc\colon \mathbb{S}_c \to \mathbb{B}], [sw\colon \mathbb{L} \times \mathbb{R} \to \mathbb{S}_w], [addSw\colon \mathbb{S}_w \times \mathbb{L} \times \mathbb{R} \to \mathbb{S}_w], \\ [coxSw\colon \mathbb{S}_w \times \mathbb{C} \to \mathbb{B}], [finSw\colon \mathbb{S}_w \to \mathbb{B}] \end{array} \right\} \rangle$$

2. (a) $finSc(addSc(sc(each), each))$
   (b) $coxSw(addSw(sw(left, right), left, right), coxswain)$

Problem 3.2 (Subset) 10 pt

Consider the following ADT: $\mathcal{A} := \langle \mathcal{S}^0, \Sigma \rangle$, where $\mathcal{S}^0 := \{\mathbb{N}, \mathbb{S}, \mathbb{B}\}$ ($\mathbb{N}$ for unary natural numbers, $\mathbb{B}$ for truth values, and $\mathbb{S}$ for sets over $\mathbb{N}$) and

$$\Sigma := \{[o\colon \mathbb{N}], [s\colon \mathbb{N} \to \mathbb{N}], [\Phi\colon \mathbb{S}], [i\colon \mathbb{N} \to \mathbb{S}], [u\colon \mathbb{S} \times \mathbb{S} \to \mathbb{S}], [T\colon \mathbb{B}], [F\colon \mathbb{B}]\}$$

Given the abstract procedures corresponding to:

- the logical operator "and":

$$\langle \wedge :: \mathbb{B} \times \mathbb{B} \to \mathbb{B}\,;\, \{\wedge(T,T) \rightsquigarrow T, \wedge(T,F) \rightsquigarrow F, \wedge(F,T) \rightsquigarrow F, \wedge(F,F) \rightsquigarrow F\}\rangle$$

- the logical operator "or":

$$\langle \vee :: \mathbb{B} \times \mathbb{B} \to \mathbb{B}\,;\, \{\vee(T,T) \rightsquigarrow T, \vee(T,F) \rightsquigarrow T, \vee(F,T) \rightsquigarrow T, \vee(F,F) \rightsquigarrow F\}\rangle$$

- the equality of unary natural numbers:

$$\langle = :: \mathbb{N} \to \mathbb{N}\,;\, \{=(o,o) \rightsquigarrow T, =(o,s(x)) \rightsquigarrow F, =(s(x),o) \rightsquigarrow F, =(s(x),s(y)) \rightsquigarrow =(x,y)\}\rangle$$

Write the abstract procedure for the "subset of" ("$\subseteq$") operator for sets of unary natural numbers.

Solution: First, we define the the "element of" operator as a helper abstract procedure:

$$\langle \in :: \mathbb{N} \times \mathbb{S} \to \mathbb{B}\,;\, \{\in(x,\Phi) \rightsquigarrow F, \in(x,i(x)) \rightsquigarrow =(x,y), \in(x,u(a,b)) \rightsquigarrow \vee(\in(x,a), \in(x,b))\}\rangle$$

Now, we can define the "subset of":

$$\langle \subseteq :: \mathbb{S} \times \mathbb{S} \to \mathbb{B}\,;\, \{\subseteq(\Phi,x) \rightsquigarrow T, \subseteq(i(x),y) \rightsquigarrow \in(x,y), \subseteq(u(x,y),z) \rightsquigarrow \wedge(\subseteq(x,z), \subseteq(y,z))\}\rangle$$

---

4 Programming in Standard ML

10pt
7min

Problem 4.1 (Cars)

Bob, the policemen in a certain coastal town, has his own unique (though perhaps not very efficient) way of recording the number and the arrangement of parked cars on a street. He uses a string in which the substring `"car"` represents a car on the street and each character between two `"car"`s represents one meter of space between the cars. So in Bob's notation `"car␣a␣car␣treecar␣"` represents a street with 3 parked cars with 3m between the first two, 5m between the second and third and 1m after the third. Your tasks are:

1. Write an SML function that given a string counts the number of cars in it:

   ```
   val number_of_cars = fn : string -> int
   - number_of_cars("car␣tree␣car␣car_␣car␣");
   val it = 4 : int
   ```

2. Write an SML function that given a "street" returns a list consisting of of `0`s and `1`s with `1` representing a car on the street and `0` representing 1 meter of space between the cars:

   ```
   val cars = fn : string -> int list
   - cars("car_car_space_car␣");
   val it = [1,0,1,0,0,0,0,0,0,0,1,0] : int list
   ```

Solution:

```
fun street([]) = ([],0)
  | street((#"c")::(#"a")::(#"r")::l) = let val (m,a) = street(l) in (1::m,a+1)
end
  | street(a::l) = let val (m,a) = street(l) in (0::m,a) end;
fun number_of_cars(l) = let val (m,a) = street(explode(l)) in a end;
fun cars(l) = let val (m,a) = street(explode(l)) in m end;
```

Problem 4.2 (Fibonacci) 15pt 10min

Let $n \geq 1$ be a natural number. We say that a sequence of integers $a_1, \ldots, a_n$ satisfies the "Fibonacci property" if either $n \leq 2$ or $a_k = a_{k-1} + a_{k-2}$, for all k with $(3 \leq k \leq n)$. Do not confuse this concept with the well-known sequence of "Fibonacci numbers" which does have the Fibonacci property, but other sequences also do.

Write an SML function `longestFibonacci` which takes a list $[x_1, x_2, x_3, \ldots, x_n]$ of integers as argument, and returns the longest sublist $[x_i, x_{i+1}, x_{i+2}, \ldots, x_j]$ that satisfies the Fibonacci property.

Examples:

```
- longestFibonacci [1,2,3,5,8,15,16,2,18,20,38,58,3,4,5];
bval it = [16,2,18,20,38,58] : int list
- longestFibonacci [1,2,4];
val it = [1,2] : int list
```

Note: If there are more of these longest sequences, you can return any of them, whichever you want. You may use the function `length: 'a list -> int` without defining it.

Hint: To find the longest subsequence with a property p, recursively compare the longest subsequence of a list with that of its tail.

Solution:

```
fun firstFibonacci ls =
    if length ls < 3 then ls
    else let val (a :: b :: c :: t) = ls
         in if a + b = c
              then a :: firstFibonacci (b :: c :: t)
              else [a,b]
         end

fun longestFibonacci ls =
    if length ls < 3 then ls
    else let val h :: t = ls
             val l = longestFibonacci t
             val nl = firstFibonacci (h :: t)
         in if length l < length nl then nl else l
         end
```

Problem 4.3 (Angry Birds) 10pt 7min

Your new favorite game is Angry Birds, and, after a lazy afternoon when you have played the game, you observed the following rules for deducing the score:

- the red bird will always add 5000 points to your score (no matter what it hits)
- the blue bird always is split into 3 smaller birds, and every bird adds 1000 points if it hits an object
- the yellow bird will add 2500 birds only if it hits a green pig
- the white bird is considered peaceful and will add no points to your score

Your version of Angry Birds permits you to choose K from a list of birds which you will fire. Therefore, you design the following SML datatype:

```
datatype angrybird = white | red | blue of int | yellow of bool;
```

The `int` parameter of the `blue` bird tells you how many birds hit an object, and the `bool` parameter of the `yellow` bird tells you if the bird hits a pig or not.

You are required to write an SML function `getScore` which takes a list of Angry Birds and the number K of birds you can fire and returns the maximum score you can get by firing your choice of K birds:

```
val getScore = fn : angrybird list * int -> int;
- getScore( [white, white, red, blue(3)], 3 );
val it = 8000 : int; (* red, blue and one of the whites are chosen *)
- getScore( [red, yellow(false), yellow(true), blue(2)], 2);
val it = 7500 : int; (* red and yellow(true) are chosen *)
```

Hint: For sorting a list of ints in descending order, you can use the following SML snippet:

```
- ListMergeSort.sort (op<) [1,5,1,2];
val it = [5,2,1,1] : int list;
```

Solution:

```
datatype angrybird = white | red | blue of int | yellow of bool;

fun convert(white) = 0
  | convert(red) = 5000
  | convert(blue(x)) = x*1000
  | convert(yellow(x)) = if x then 2500 else 0;

fun getScore(l,k)=
       let
              val l1 = map convert l
              val l2 = ListMergeSort.sort (op<) l1
       in
              foldl (op+) 0 (List.take(l2, k))
       end;
```
