Name: Matriculation Number:

Midterm Exam
General CS I (320101)

October 14, 2013

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

The estimated time for solving this exam is 0 minutes, leaving you 75 minutes for revising your exam.

You can reach 0 points if you solve all problems. You will only need 100 points for a perfect score, i.e. -100 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.	Sum	grade
total	0	
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)

A functional equation is defined to be an equation where the variable you have to solve for is a function. A simple example is 15pt 10min

Find all functions $f\colon \mathbb{N}^+ \to \mathbb{N}^+$ such that

$$f(x+y) = f(x) + f(y) \quad \text{for all} \quad x, y \in \mathbb{N}^+$$

If f is a solution to the equation above, prove that $f(x) = c * x$ for some constant c.

In this problem, $\mathbb{N}^+$ is defined to be $\{1, 2, 3, \ldots\}$.

Hint: Think of what c might be. Remember, the equation holds for all positive natural numbers.

Solution:

Proof: Proof by induction

P.1.1 Base Case: Since both $c * 1$ and $f(1)$ are constants we can assume $c = f(1)$. □

P.1.2 Step Case:

P.1.2.1 Assume for some $f(k) = f(1) * k$ for some $k \in \mathbb{N}^+$.

P.1.2.2 Proof for $k+1$: $f(k+1) = f(k) + f(1) = k * f(1) + f(1) = (k+1) * f(1)$. □

P.2 Therefore, we have proven the theorem. □

Problem 1.2 (Greek Letters)

Fill in the blanks in the following table of Greek letters. Note that capitalized names denote capital Greek letters. 5pt 2min

Symbol		η	ν			O		ι	Φ	
Name	Tau			sigma	upsilon		Xi			chi

Solution:

Symbol	Δ	ψ	ν	σ	δ	Λ	Ξ	ω	Ψ	χ
Name	Delta	psi	nu	sigma	delta	Lambda	Xi	omega	Psi	chi

Problem 1.3 (UNN Powers)

Give the defining equations for the the power operation $\pi\colon \mathbb{N}_1 \times \mathbb{N}_1 \to \mathbb{N}_1$ on unary natural numbers. Assume the addition $\alpha\colon \mathbb{N}_1 \times \mathbb{N}_1 \to \mathbb{N}_1$ and multiplication $\mu\colon \mathbb{N}_1 \times \mathbb{N}_1 \to \mathbb{N}_1$ operations are already given. 6pt 3min

Solution: The defining equations are $\pi(n, o) = s(o)$ and $\pi(n, s(k)) = \mu(n, \pi(n, k))$

2 Sets, Relations and Functions

Problem 2.1 (Intersection of sets)

15pt

1. Prove or refute that if $A \subseteq B$ and $C \subseteq D$ then: 8min
   - $A \cup C \subseteq B \cup D$
   - $A \cap C \subseteq B \cap D$
2. Let $A \subseteq T$ and denote $\bar{A} = T \backslash A$ the complement of A. Also let $B \nsubseteq T$ with $B \cap T \neq \emptyset$. Knowing that $\#(B) = 2$ and $2 \nmid \#(\bar{A})$, find the size of the set $A \cap B$.
3. Let A, B, and C be sets. Prove or refute that $\#(A \cup B) = \#(A) + \#(B) - \#(A \cap B)$. Formulate the corresponding claim for three sets (A, B, and C) and prove or refute it.

Note: Points will be partly awarded for using MathTalk in all the solutions of the above questions! Venn-Euler diagrams may be used but the proofs need to be rigorous.

Solution:

Problem 2.2 (Love relations)

8pt

1. Love is in the air 5min

   Taking into consideration the love table below give the `love` relation.

Note: A tick (✓) on line r, column c means that the student in the beginning of row r loves the student on top of column c.

♡	Ann	Bill	Inna	Bob	Filip	Ivan
Ann	✓	✓		✓	✓	✓
Bill		✓		✓		
Inna			✓		✓	
Bob		✓		✓		✓
Filip						
Ivan	✓		✓			✓

`love` := {⟨Ann, Ann⟩, ⟨Ann, Bill⟩, ... ? ... ...

}

2. What kind of relation is this? Look back at the relation you have defined and prove/refute whether it is *reflexive*, *symmetric* and respectively *transitive*. Make sure you justify your answer

Solution:

1. `love` := $\{\langle$Ann, Ann$\rangle$, $\langle$Ann, Bill$\rangle$, $\langle$Ann, Bob$\rangle$, $\langle$Ann, Filip$\rangle$, $\langle$Ann, Ivan$\rangle$,
   $\langle$Bill, Bill$\rangle$, $\langle$Bill, Bob$\rangle$, $\langle$Inna, Inna$\rangle$, $\langle$Inna, Filip$\rangle$, $\langle$Bob, Bill$\rangle$,
   $\langle$Bob, Bob$\rangle$, $\langle$Bob, Ivan$\rangle$, $\langle$Ivan, Ann$\rangle$, $\langle$Ivan, Inna$\rangle$, $\langle$Ivan, Ivan$\rangle$
   $\}$

2. - `love` is not reflexive since Filip does not love himself. (e.g. $\langle$Filip, Filip$\rangle \notin$ `love`)
   - `love` is not symmetric since Ann loves Bill but Bill does not love Ann. (e.g. $\langle$Ann, Bill$\rangle \in$ `love`, but $\langle$Bill, Ann$\rangle \notin$ `love`)
   - `love` is not transitive since Ann loves Ivan, Ivan loves Inna but Ann does not love Inna. (e.g. $\langle$Ann, Ivan$\rangle \in$ `love` and $\langle$Ivan, Inna$\rangle \in$ `love`, but $\langle$Ann, Inna$\rangle \notin$ `love`)

Problem 2.3 (Function injectivity/surjectivity)

10pt
6min

1. Prove or refute that the function $f\colon \mathbb{N} \to \mathbb{N}; n \mapsto 2n+1$ is bijective.
2. Prove or refute that the function $g\colon \mathbb{N} \to \mathbb{N}\backslash\{0\}; n \mapsto n+1$ is bijective.

Solution:

1. f is not bijective it is not surjective, For example $2 \in \mathbb{N}$ and there is no $n \in \mathbb{N}$ that $2 = 2n+1$.
2. g is bijective: Let $n \in (\mathbb{N}\backslash\{0\})$, then $n = g(n-1)$, so g is surjective. It is also injective by the third Peano axiom.

3 Abstract Data Types and Abstract Procedures

Problem 3.1 (Aaah the Roses)

15pt
10min

- Design an ADT for flower bouquets. There are 3 types of flowers: roses, lilies and gerberas. There are 2 types of bouquets: a formal and an informal one. The informal bouquet is made of (upto an infinite amount) of layers. Each layer contains 3 flowers of any type. The formal bouquet is also made of layers containing 5 flowers each.
- Give the representation of an informal bouquet containing 3 roses, 3 lilies and 3 gerberas in whichever order you want.
- Now create an abstract procedure that given a bouquet calculates the number of roses, gerberas and lilies in that bouquet.

Hint: Assume that an ADT for Integers is given, so use them normally.

Solution:

4 Programming in Standard ML

Problem 4.1 (Word remover)

In this problem you are required to write several small SML functions that build upon each other. You are allowed to use a function even if you fail to write it in any previous step. 20pt 16min

1. Write an SML function that checks of one list is a prefix of another, i.e. if the second list begins with the first list.
```
val prefix = fn : ''a list * ''a list -> bool
- prefix ([1,2,3],[1,2,3,4,5]);
val it = true : bool
```

2. Write an SML function that finds the first occurence of a sublist in a list.
```
val find_first = fn : ''a list * ''a list -> int
- find_first([1,2,3],[4,4,4,1,2,3,4]);
val it = 3 : int
```

3. Write an SML function that removes the first occurence of a sublist in a list.
```
val remove_first = fn : ''a list * ''a list -> ''a list
- remove_first([1,1],[4,4,1,1,4,4,1,1]);
val it = [4,4,4,4,1,1] : int list
```

4. Write an SML function that deletes all occurences of a word in a string.
```
val delete_word = fn : string * string -> string
- delete_word("not", "SML is not fun and not easy");
val it = "SML is fun and easy" : string
```

Solution:

```
fun prefix([],ls) = true
  | prefix(a::ls, []) = false
 | prefix(a::lsa, b::lsb) = if a=b then prefix(lsa,lsb) else false;

fun find_first(w,[]) = ~1
  | find_first(w,ls) = if prefix(w,ls) then 0 else
                       let val r = find_first(w,tl(ls))
                       in if r=(~1) then ~1 else r+1
                       end;

fun remove_first(w,ls) = let val i = find_first(w,ls)
                         in if i=(~1)
                            then ls
                            else List.take(ls,i)@List.drop(ls,i+length(w))
                         end;

fun remove_all(w,ls) = if find_first(w,ls) = ~1
                        then ls
                        else remove_all(w,remove_first(w,ls));

fun delete_word(w,s) = implode(remove_all(explode(w),explode(s)));
```

Problem 4.2 (Music notes)

In music theory, notes are divided into groups of 12 semitones. Those groups are called octaves. In each octave, we have the familiar 7 note classes - A, B, C, D, E, F and G (ordered from lowest to highest). In order to denote all 12 semitones, we use the special symbols operators: sharp ($\sharp$) and flat ($\flat$), which mean 1 semitone higher and lower, respectively. Now we only need to know the number of semitones between those classes: 15pt 10min

A to B: 2 semitones
B to C: 1 semitone
C to D: 2 semitones
D to E: 2 semitones
E to F: 1 semitone
F to G: 2 semitones
G to A: 2 semitones

This leads to many different representations of the same note. For example, $C = \sharp B = \flat\flat D = \sharp\flat C$

For a computer program, representing each note with a unique integer is much more convenient. To do that, we number all the notes by increasing pitch: A from octave 0 becomes 0, $\sharp A$ from octave 0 becomes 1, A from octave 1 becomes 12, B from octave 1 become 14, etc.

You are given an SML datatype for notes:

```
datatype noteclass = A | B | C | D | E | F | G |
                     | sharp of noteclass
                     | flat of noteclass;

datatype note = note of int * noteclass; (* octave number and note class *)
```

Write an SML function that converts a list of notes from the datatype to the integer representation described above. Example and signature:

```
val convert = fn : note list -> int list

- convert([note(0,sharp(A)), note(1,flat(flat(B)))]);
val it = [1,12] : int list
```

Hint: Consider writing a helper function to convert a single note.

Solution:

```
fun toneval(note(x,A)) = x*12
  | toneval(note(x,B)) = x*12+2
  | toneval(note(x,C)) = x*12+3
  | toneval(note(x,D)) = x*12+5
  | toneval(note(x,E)) = x*12+7
  | toneval(note(x,F)) = x*12+8
  | toneval(note(x,G)) = x*12+10
  | toneval(note(x,sharp(n))) = toneval(note(x,n)) + 1
  | toneval(note(x,flat(n))) = toneval(note(x,n)) - 1;
```

```
fmiun convert(ls) = map toneval ls;
```