General Computer Science I (320101) Fall 2011 Assignment 3: Relations, Functions and an Introduction to SML

(Given Sep. 30., Due Sep. 7.)

15pt

Problem 3.1 (Two functions)
Determine whether each of the following functions is injective, surjective or bijective:

1. $f\colon \mathbb{R} \to \mathbb{R}, f(x) := \begin{cases} x^2 & x \geq 2 \\ 3x-2 & x < 2 \end{cases}$
2. $g\colon \mathbb{R} \to \mathbb{R}, g(x) := 2(x^4) + 3(x^3) + 4$

25pt

Problem 3.2 (Injective functions)
Find all injective functions $f\colon \mathbb{N} \to \mathbb{N}$ (where $\mathbb{N}$ is the set of natural numbers), such that $\forall n \in \mathbb{N}. f(f(n)) \leq \frac{n+f(n)}{2}$.

Hint: One way to solve the problem would be to show that if $n < f(n)$, then it can be proven that $f^k(n) < f(n)$. Using this and the properties of injectivity, a contradiction would follow. Finally, argue about the case when $f(n) \leq n$ in order to discover the wanted functions.

15pt

Problem 3.3 (Relation Properties)
You are given the set $A = \{a, b, c, d, e\}$. Find the number of relations $R \subseteq A \times A$, which are simultaneously reflexive and symmetric.

25pt

Problem 3.4 (Closest Numbers)
Write an SML function closestTwoElements which takes a list (with at least two elements) as argument and returns (as a pair) the two elements in the list which are closest to each other in value. If there are more of such pairs, only return one of them, whichever you want. The signature of the function should be:

```
val closestTwoElements = fn : int list −> int ∗ int
```

Example:

```
− closestTwoElements [1, 50, 35, 4, 11, 33, 53];
val it = (33,35) : int ∗ int
```

30pt

Problem 3.5 (Playing with shapes)
This problem will involve “drawing” figures of ones - your functions should return lists of “0” and “1” (integers). Keep in mind that your functions should respect the specified signature.

1. Write an SML function that takes an integer n and prints a “square of ones” with length n.

   ```
   val square = fn : int −> int list list
   ```

   For example:

```
- square(4);
val it = [[1,1,1,1],[1,1,1,1],[1,1,1,1],[1,1,1,1]] : int list  list
```

2. Now, write an SML function that, given an integer n, will print a “diamond of ones” with side n.

   val diamond = **fn** : int −> int list list

   For example:

```
- diamond(4);
val it = [[1],[1,1],[1,1,1],[1,1,1,1],[1,1,1],[1,1],[1]] : int list  list
```

3. You should modify the previous function in order to have “inner elements” set to 0 instead of 1.

   val cdiamond = **fn** : int −> int list list

   For example:

```
- cdiamond(4);
val it = [[1],[1,1],[1,0,1],[1,0,0,1],[1,0,1],[1,1],[1]] : int list  list
```