LiP Final project

Dipartimento di Matematica e Informatica, Univ. Cagliari

Deadline for submission Feb 3, 2013 (strict!)

Consider a variant of the imperative language IMP, called IMP2, which comprises constant/variable declarations, and expressions of the following kinds:

- boolean values;
- boolean conjunction, disjunction, negation;
- identifiers;
- unlimited precision integers ("big integers"). A big integer is represented as a list of (small) integers.
- addition of two big integers;
- comparison (=, <) of two big integers;

The abstract syntax of expressions is as follows:

```
type bigint = int list;;
type exp = True
         | False
         | Not of exp
         | And of exp * exp
         | Or of exp * exp
         | Int of bigint
         | Add of exp * exp
         | Eq of exp * exp
         | Lt of exp * exp
         | Val of ide
```

The language IMP2 features the following commands:

- skip;
- assignment to variables;
- sequencing;
- conditionals;
- break;
- repeat. The body of the repeat command is executed indefinitely until a break is encountered.

```
type com = Skip
         | Break
         | Assign of ide * exp
         | Cseq of com * com
         | If of exp * com * com
         | Repeat of com
```

The project comprises the following steps:

1. Write an Ocaml interpreter of IMP2.

2. Write a compiler from IMP2 to the assembly language ASM. The compiler should reject programs with ill-formed expressions and commands. Some instances of ill-formed expressions/commands are: a conjunction of a boolean and a big integer, a conditional with a non-boolean guard, *etc.* An interpreter of the ASM language can be found on the UnicaML wiki.

3. To test the compiler, write and compile an IMP2 program that computes the n-th Fibonacci number, for an arbitrary n.