Milestone 1

Tormod Hellen and Guillaume Pansier

03.06.2014

1 BNF syntax

```
Prog ::= Def \#
        | Def Prog
Def ::= DEF Lhs == Expr (Def)
Lhs ::= MAIN: Type
                                     (MAIN)
        | id (A) : Type
                                     (Function_def)
{\bf Type} \ ::= \ {\bf nat}
                                     (Type)
        bool
                                              (Type)
                                     (Params)
A ::= id : Type(B)
B ::= , id : Type B
        id : Type B
         е
\operatorname{Expr} ::= \operatorname{number}
                           (Number)
         true
                                     (bool)
          false
                                              (bool)
                                     (Function, Var)
         if Expr then Expr E fi
                                              (ITE)
C ::= (Expr D)
D ::= , Expr D
        e
E ::= ELSE Expr
       e
```

2 BNF syntax without left-recursion

```
\begin{array}{lll} \operatorname{Prog} & ::= & \operatorname{Def} & \operatorname{Prog1} \ \# \\ \operatorname{Prog1} & ::= & \operatorname{Def} & \operatorname{Prog1} \\ & & | & \operatorname{e} \\ \operatorname{Def} & ::= & \operatorname{DEF} & \operatorname{Lhs} & == & \operatorname{Expr} & (\operatorname{Def}) \\ \operatorname{Lhs} & ::= & \operatorname{MAIN} & : & \operatorname{Type} \\ & & | & \operatorname{id} & (\operatorname{A} \ ) & : & \operatorname{Type} \end{array} \tag{MAIN} \\ & & | & \operatorname{id} & (\operatorname{A} \ ) & : & \operatorname{Type} \end{array} \tag{Function\_def}
```

```
A \ ::= \ id \ : \ Type \ B
                                         (Params)
B \ ::= \ , \ id \ : \ Type \ B
       e
                                (Type)
\mathbf{Type} \ ::= \ \mathbf{nat}
 bool
                                (Type)
Expr := number
                      (Number)
                                (bool)
       true
        false
                                         (bool)
                                (Function, Var)
        id C
       if Expr then Expr E fi
                                        (ITE)
C ::= (Expr D)
e
E ::= ELSE Expr
     e
```

All test files in skelton