IMP

END MODULE

```
MODULE IMP-SYNTAX
   SYNTAX AExp ::= Int
                       String
                      Id
                       ++ Id
                       read ()
                      AExp / AExp [division( division()), strict( strict())]
                      AExp + AExp [strict( strict())]
                      (AExp) [bracket( bracket())]
   SYNTAX BExp ::= Bool
                      AExp \le AExp [seqstrict( seqstrict())]
                       ! BExp [strict( strict())]
                      BExp && BExp [strict( strict(1))]
                      (BExp) [bracket( bracket())]
   SYNTAX Block := \{\}
                     |\{Stmt\}|
   \mathtt{SYNTAX} \quad \mathit{Stmt} ::= \mathit{Block}
                     Id = AExp; [strict(strict(2))]
                      if (BExp)Block else Block [strict( strict(1))]
                      while (BExp)Block
                      int Ids ;
                      print (AExps) ; [strict( strict())]
                      halt ;
                      spawn Stmt
                     Stmt Stmt
   SYNTAX Ids ::= List\{Id, ", "\}
   SYNTAX AExps ::= List\{AExp, ", "\} [seqstrict(seqstrict())]
END MODULE
MODULE IMP
   SYNTAX KResult ::= Int
                       Bool
  CONFIGURATION:
            PGM:Stmt
                                    .Map
                                                    .Map
                                        store
                           X \mapsto N
                                                                                                                                                                                                                                                                    [lookup( lookup())]
  RULE
                                        N \mapsto I
                                                                                                                                                                                                                                                              [increment( increment())]
  RULE I1 / I2
                           requires I2 = /=_{Int} 0
          I1 \div_{Int} I2
  RULE I1 + I2
          \overline{I1 +_{Int} I2}
  Rule I1 \leq I2
          \overline{I1 \leq_{Int} I2}
          ! T
          \neg_{Bool} T
  RULE true && {\cal B}
              B
  RULE false && —
             false
                                                                                                                                                                                                                                                               [structural( structural())]
  RULE
                                                                                                                                                                                                                                                               [structural( structural())]
                X = I:Int;
  RULE S1:Stmt S2:Stmt
                                                                                                                                                                                                                                                               [structural() structural())]
               S1 \curvearrowright S2
  {\tt RULE} \quad {\tt if} \; ({\tt true}) S \; {\tt else} \, -\!\!\!\!\!-
  \mathsf{while}\;(B)S
                                                                                                                                                                                                                                                               [structural( structural())]
  RULE
          int X , Xs ;
  RULE
                                                                  .Map
                                        \rho[X \leftarrow N:Int]
  RULE int \bullet_{Ids} ;
                                                                                                                                                                                                                                                               [structural( structural())]
```