IMP

MODULE IMP-SYNTAX

SYNTAX *AExp* ::= *Int*
| *String*
| *Id*
| `++` *Id*
| `read` ()
| *AExp* `/` *AExp* [strict]
| *AExp* `+` *AExp* [strict]
| `spawn` *Block* [strict]
| *Id* `=` *AExp* [strict(2)]
| (*AExp*) [bracket]

SYNTAX *BExp* ::= *Bool*
| *AExp* ≤ *AExp* [strict]
| `!` *BExp* [strict]
| *BExp* && *BExp* [strict]
| (*BExp*) [bracket]

SYNTAX *Block* ::= {*Stmts*}

SYNTAX *Stmt* ::= *Block*
| *AExp* `;` [strict]
| `if` (*BExp*)*Block* `else` *Block* [strict]
| `while` (*BExp*)*Block* [strict]
| `int` *Ids* `;`
| `print` (*AExps*) `;` [strict]
| `halt` `;`
| `join` *AExp* `;` [strict]

SYNTAX *Ids* ::= *List*{*Id*, ","} [strict]

SYNTAX *AExps* ::= *List*{*AExp*, ","} [strict]

SYNTAX *Stmts* ::= *List*{*Stmt*, ""} [seqstrict]

END MODULE

MODULE IMP

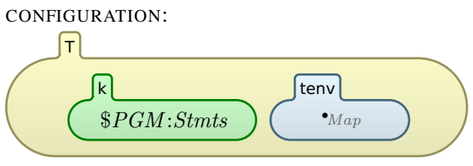
SYNTAX *BlockOrStmtType* ::= `block`
| `stmt`

SYNTAX *PrintableType* ::= `int`
| `string`

SYNTAX *Type* ::= *PrintableType*
| `bool`
| *BlockOrStmtType*

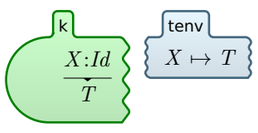
SYNTAX *KResult* ::= *Type*

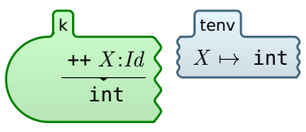
CONFIGURATION:



RULE —:Int / `int`

RULE —:String / `string`

RULE 

RULE 

RULE `read` () / `int`

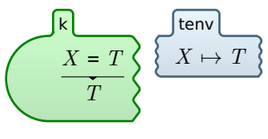
SYNTAX *AExp* ::= *Type*

RULE `int` `/` `int` / `int`

RULE `int` `+` `int` / `int`

RULE `string` `+` `string` / `string`

RULE `spawn` `block` / `int`

RULE 

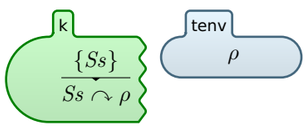
SYNTAX *BExp* ::= *Type*

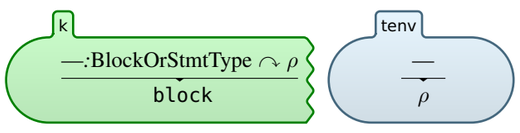
RULE —:Bool / `bool`

RULE `int` ≤ `int` / `bool`

RULE `!` `bool` / `bool`

RULE `bool` && `bool` / `bool`

RULE 

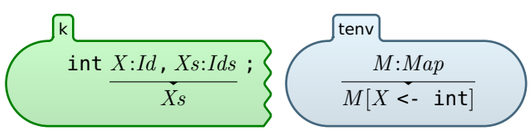
RULE 

RULE `int` `;` / `stmt`

SYNTAX *Block* ::= *Type*

RULE `if` (`bool`) `block` `else` `block` / `stmt`

RULE `while` (`bool`) `block` / `stmt`

RULE 

RULE `int` •*Ids* `;` / `stmt`

RULE `print` (—:PrintableType , *AEs*) `;` / *AEs*

RULE `print` (•*AExps*) `;` / `stmt`

RULE `halt` `;` / `stmt`

RULE `join` `int` `;` / `stmt`

RULE •*Stmts* / `stmt`

RULE —:BlockOrStmtType *Ss* / *Ss*

END MODULE