Require Import ZArith.
Require Import String.

Open Scope string_scope.
Open Scope Z_scope.

Inductive binop : Set :=
| Add | Sub | Mul | Div | Mod | Lt | Lte | Conj | Disj.

Inductive expr : Set :=

Coercion Int : Z >-> expr.
Coercion Var : string >-> expr.

Notation "[+ X Y]" := (BinOp Add X Y) (at level 51, left associativity).
Notation "[- Y]" := (BinOp Sub X Y) (at level 51, left associativity).
Notation "[* Y]" := (BinOp Mul X Y) (at level 50, left associativity).
Notation "[/ X]" := (BinOp Div X Y) (at level 50, left associativity).
Notation "[% X Y]" := (BinOp Mod X Y) (at level 50, left associativity).
Notation "< X Y" := (BinOp Lt X Y) (at level 52).
Notation "<= X Y" := (BinOp Lte X Y) (at level 52).
Notation "&& X Y" := (BinOp Conj X Y) (at level 53, left associativity).
Notation "|| X Y" := (BinOp Disj X Y) (at level 54, left associativity).

Notation "[[X Y]]" := (Pair X Y) (at level 55).
Notation "X.'L'" := (ProjL X) (at level 55).
Notation "X.'R'" := (ProjR X) (at level 55).

Inductive stmt : Set :=

Notation "nop" := (Nop) (at level 60).
Notation "X := Y" := (Assign X Y) (at level 60).
Notation "X := Y" := (Assign X Y) (at level 60).
Notation "if X [[Y]]" := (Cond X Y) (at level 60).
Notation "while X [[Y]]" := (While X Y) (at level 60).