TPL PROJECT

on

INTERPRETER FOR UNTYPED ARITHMETIC EXPRESSIONS WITH EXTENSIONS

STAGE 2

Course : CS6124D Topics in Programming Languages

Submission date: 14.04.2021

1 Design

1.1 Syntax

t:= and t or t not t

Sample valid terms : and true or true not false, and (iszero 0) or (iszero (succ 0)) not (iszero (pred (succ 0)))

1.2 Operational Semantics

The axioms and inference rules for the evaluation of the new term are as follows:

and false or t2 not t3 → false
and true or true not t3 → true
and true or false not true → false
and true or false not false → true

$\frac{t1 \rightarrow t1'}{}$
and t1 or t2 not t3 → and t1' or t2 not t3

$\frac{t2 \rightarrow t2'}{}$
and true or t2 not t3 → and true or t2' not t3

$\frac{t3 \rightarrow t3'}{}$
and true or false not t3 → and t1 or t2 not t3'

2 Implementation Details

The term is of the form - ' and t1 or t2 not t3 '.

The term is implemented using a short circuit evaluation method as follows:

1. t1 is evaluated first. If it is false, the value false is returned without further evaluation.
2. if t1 evaluates to true, t2 is evaluated :
   a) If t2 is true , the value true is returned without further evaluation.
   b) If t2 is false, t3 is evaluated. :
      i. If t3 is true, then false is returned.
      ii. If t3 is false, then true is returned.
      iii. if t3 is neither true nor false, stuck term is produced.
   c) if t2 is neither true nor false, stuck term is produced.
3. if t1 is neither true nor false, stuck term is produced.

The input line read from the user is compared with the pre-defined tokens in order to tokenize it. The result is a double ended queue of different tokens identified from the input in the order in which they appear. The queue is further converted into a nested list with each component list forming a step in the derivation tree. Each component list in the nested list is to be evaluated using pre-defined rules. Each subterm in the term is evaluated recursively until a value or a stuck term is produced.

3 Results

```
sandra@sandra-vivobook:~/sandra/TPL/latest/Interpreter-for-Untyped-arithmetic-expressions$ python3 main.py
> and(true)or(false)not(false)
true
> and(false)or(succ(pred(0)))not(true)
false
> and(iszero(0))or(succ(pred(0)))not(true)
and(true)or(succ ( 0 ))not(true)
> and(if(true)then(true)else(false))or(iszero(succ 0))not(false)
true
> and(true)or(iszero(pred(0)))not(false)
true
>
```