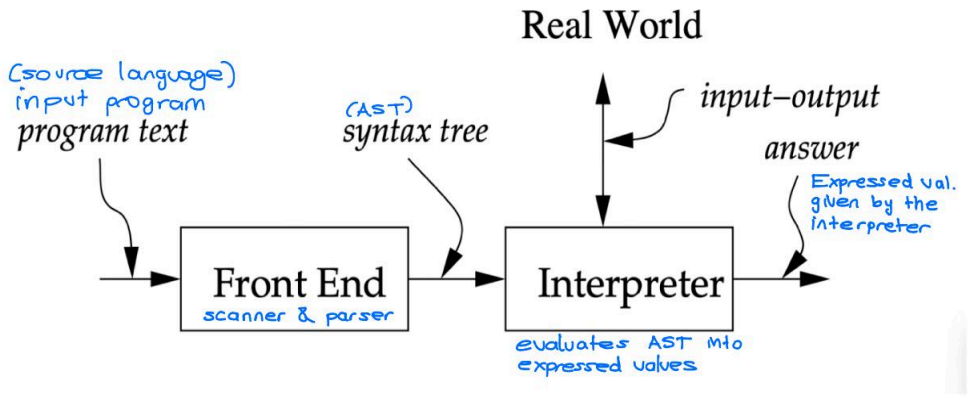
Beste Kalaycı 76766
Beyda Dur 75881
Doğa Ceylan 76168

COMP301 Project 2 Report

A)



The corresponding 5 components of MY-LET language is:

- **Lexical Specification** - Lexical specification is defined in **lang.rkt.** It specifies how an input program written in the source language gets scanned by the Scanner.
- **Grammar (Syntax)** - Grammar specification is defined in **lang.rkt**. It specifies the source language (defined language) syntax, and how it gets parsed into an AST by the Parser.
- **Scanner & Parser -** Implemented in scan&parse in **lang.rkt.** It takes a program in MY-LET and builds an AST.
- **Expressed Values -** Defined and handled in **data-structures.rkt.** It includes num, rational, bool, and list-of-nums, which are structs with fields to hold the corresponding values in the implementation language.
- **Interpreter -** Evaluation of the MY-LET program is handled in value-of-program in **interp.rkt,** which uses the recursive procedure value-of to handle the different expressions of MY-LET. Interpreter evaluates the different expressions of MY-LET and extracts their semantic value (evaluation).It uses an environment to keep track of bound variables.

B)

The initial environment is defined in init-env in **environments.rkt**.

```
;;; initialize environment with x, y, z
(define init-env
  (lambda ()
    (extend-env 'z (num-val 3)
                (extend-env 'y (num-val 2)
                            (extend-env 'x (num-val 1)
                                        (empty-env))))))
```

We used frame-based representation in EOPL, p.61,

(init-env) => ρ
= [z=[3]] ([y=[2] ([x=[1]] []))
= [z=[3]] ([y=[2] [x=[1]])
= [z=[3]] [y=[2], x=[1]]
= [z=[3], y=[2], x=[1]]

C)

Each language has at least two such sets: the expressed values and the denoted values. The expressed values are the possible values of expressions, and the denoted values are the values bound to variables.

ExpVal = Int + Bool + Pair<Int,Int> + List<Int>
DenVal = Int + Bool + Pair<Int,Int> + List<Int>