

RFAE – FAE with Recursion and Conditionals

1 INTRODUCTION

RFAE is a toy language for the [COSE212](#) course at Korea University. RFAE stands for an extension of the [FAE](#) language with **recursion** and **conditionals**, and it supports the following features:

- **number (integer) values** (0, 1, -1, 2, -2, 3, -3, ...)
- **arithmetic operators**: negation (-), addition (+), subtraction (-), multiplication (*), division (/), and modulo (%)
- **immutable variable definitions** (val)
- **first-class functions** (=>) and **recursive functions** (def)
- **conditionals** (if-else)
- **boolean values** (true and false)
- **comparison operators**: equality (== and !=) and relational (<, >, <=, and >=)
- **logical operators**: conjunction (&&), disjunction (| |), and negation (!)

This document is the specification of RFAE. First, Section 2 describes the concrete syntax, and Section 3 describes the abstract syntax with the desugaring rules. Then, Section 4 describes the big-step operational (natural) semantics of RFAE.

2 CONCRETE SYNTAX

The concrete syntax of RFAE is written in a variant of the extended Backus–Naur form (EBNF). The notation `<nt>` denotes a nonterminal, and `"t"` denotes a terminal. We use `?` to denote an optional element and `+` (or `*`) to denote one or more (or zero or more) repetitions of the preceding element. We use **butnot** to denote a set difference to exclude some strings from a producible set of strings. We omit some obvious terminals using the ellipsis (...) notation.

```
// basic elements
<digit>    ::= "0" | "1" | "2" | ... | "9"
<number>   ::= "-"? <digit>+
<alphabet> ::= "A" | "B" | "C" | ... | "Z" | "a" | "b" | "c" | ... | "z"
<idstart>  ::= <alphabet> | "_"
<idcont>   ::= <alphabet> | "_" | <digit>
<keyword>  ::= "val" | "true" | "false" | "def" | "if" | "else"
<id>       ::= <idstart> <idcont>* butnot <keyword>

// expressions
<expr>     ::= <number> | "true" | "false" | <id>
            | <uop> <expr> | <expr> <bop> <expr>
            | "(" <expr> ")" | "{" <expr> "}"
            | "val" <id> "=" <expr> ";" <expr>
            | <id> "=" <expr> | <expr> "(" <expr> ")"
            | "def" <id> "(" <id> ")" "=" <expr> ";" <expr>
            | "if" "(" <expr> ")" <expr> "else" <expr>

// operators
<uop>      ::= "-" | "!"
<bop>      ::= "+" | "-" | "*" | "/" | "%" | "&&" | "||"
            | "==" | "!=" | "<" | "<=" | ">" | ">="
```

The precedence and associativity of operators are defined as follows:

Description	Operator	Precedence	Associativity
Unary	$-, !$	1	right
Multiplicative	$*, /, \%$	2	left
Additive	$+, -$	3	
Relational	$<, <=, >, >=$	4	
Equality	$==, !=$	5	
Logical Conjunction	$\&\&$	6	
Logical Disjunction	$ $	7	

3 ABSTRACT SYNTAX

The abstract syntax of RFAE is defined as follows:

Expressions	$\mathbb{E} \ni e ::= n$	(Num)
	$ b$	(Bool)
	$ x$	(Id)
	$ e + e$	(Add)
	$ e * e$	(Mul)
	$ e / e$	(Div)
	$ e \% e$	(Mod)
	$ e == e$	(Eq)
	$ e < e$	(Lt)
	$ \lambda x. e$	(Fun)
	$ \text{def } x(x) = e; e$	(Rec)
	$ e(e)$	(App)
	$ \text{if } (e) e \text{ else } e$	(If)

where

Numbers	$n \in \mathbb{Z}$	(BigInt)
Booleans	$b \in \mathbb{B} = \{\text{true}, \text{false}\}$	(Boolean)
Identifiers	$x \in \mathbb{X}$	(String)

The semantics of the remaining cases are defined with the following desugaring rules:

$\mathcal{D}[- e]$	$= \mathcal{D}[e] * (-1)$	$\mathcal{D}[e_1 != e_2]$	$= \mathcal{D}[\neg (e_1 == e_2)]$
$\mathcal{D}[\neg e]$	$= \text{if } (\mathcal{D}[e]) \text{ false else true}$	$\mathcal{D}[e_1 <= e_2]$	$= \mathcal{D}[(e_1 < e_2) (e_1 == e_2)]$
$\mathcal{D}[e_1 - e_2]$	$= \mathcal{D}[e_1] + \mathcal{D}[- e_2]$	$\mathcal{D}[e_1 > e_2]$	$= \mathcal{D}[\neg (e_1 <= e_2)]$
$\mathcal{D}[e_1 \&\& e_2]$	$= \text{if } (\mathcal{D}[e_1]) \mathcal{D}[e_2] \text{ else false}$	$\mathcal{D}[e_1 >= e_2]$	$= \mathcal{D}[\neg (e_1 < e_2)]$
$\mathcal{D}[e_1 e_2]$	$= \text{if } (\mathcal{D}[e_1]) \text{ true else } \mathcal{D}[e_2]$	$\mathcal{D}[\text{val } x = e_1; e_2]$	$= (\lambda x. \mathcal{D}[e_2])(\mathcal{D}[e_1])$

The omitted cases recursively apply the desugaring rule to sub-expressions.

4 SEMANTICS

We use the following notations in the semantics:

Values	$\mathbb{V} \ni v ::= n$	(NumV)
	$ b$	(BoolV)
	$ \langle \lambda x. e, \sigma \rangle$	(CloV)
Environments	$\sigma \in \mathbb{X} \xrightarrow{\text{fin}} \mathbb{V}$	(Env)

The big-step operational (natural) semantics of RFAE is defined as follows:

$$\boxed{\sigma \vdash e \Rightarrow v}$$

$$\begin{array}{c}
\text{Num} \frac{}{\sigma \vdash n \Rightarrow n} \quad \text{Bool} \frac{}{\sigma \vdash b \Rightarrow b} \quad \text{Id} \frac{x \in \text{Domain}(\sigma)}{\sigma \vdash x \Rightarrow \sigma(x)} \\
\\
\text{Add} \frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 + e_2 \Rightarrow n_1 + n_2} \quad \text{Mul} \frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 * e_2 \Rightarrow n_1 \times n_2} \\
\\
\text{Div} \frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2 \quad n_2 \neq 0}{\sigma \vdash e_1 / e_2 \Rightarrow n_1 \div n_2} \quad \text{Mod} \frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2 \quad n_2 \neq 0}{\sigma \vdash e_1 \% e_2 \Rightarrow n_1 \% n_2} \\
\\
\text{Eq} \frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 == e_2 \Rightarrow n_1 = n_2} \quad \text{Lt} \frac{\sigma \vdash e_1 \Rightarrow n_1 \quad \sigma \vdash e_2 \Rightarrow n_2}{\sigma \vdash e_1 < e_2 \Rightarrow n_1 < n_2} \\
\\
\text{Fun} \frac{}{\sigma \vdash \lambda x.e \Rightarrow \langle \lambda x.e, \sigma \rangle} \quad \text{Rec} \frac{\sigma' = \sigma[x_0 \mapsto \langle \lambda x_1.e_2, \sigma' \rangle] \quad \sigma' \vdash e_3 \Rightarrow v_3}{\sigma \vdash \text{def } x_0(x_1) = e_2; e_3 \Rightarrow v_3} \\
\\
\text{App} \frac{\sigma \vdash e_0 \Rightarrow \langle \lambda x.e_2, \sigma' \rangle \quad \sigma \vdash e_1 \Rightarrow v_1 \quad \sigma'[x \mapsto v_1] \vdash e_2 \Rightarrow v_2}{\sigma \vdash e_0(e_1) \Rightarrow v_2} \\
\\
\text{If}_T \frac{\sigma \vdash e_0 \Rightarrow \text{true} \quad \sigma \vdash e_1 \Rightarrow v_1}{\sigma \vdash \text{if } (e_0) e_1 \text{ else } e_2 \Rightarrow v_1} \quad \text{If}_F \frac{\sigma \vdash e_0 \Rightarrow \text{false} \quad \sigma \vdash e_2 \Rightarrow v_2}{\sigma \vdash \text{if } (e_0) e_1 \text{ else } e_2 \Rightarrow v_2}
\end{array}$$