

# AE – Arithmetic Expressions

## 1 INTRODUCTION

AE is a toy language for the [COSE212](#) course at Korea University. AE stands for a language of arithmetic expressions that supports the following features:

- **integers**
- **basic arithmetic operators**: addition (+) and multiplication (\*)

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

## 2 CONCRETE SYNTAX

The concrete syntax of AE 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>+

// expressions
<expr> ::= <number> | <expr> "+" <expr> | <expr> "*" <expr> | "(" <expr> ")"
```

The precedence and associativity of operators are defined as follows:

Operator	Associativity	Precedence
*	left	1
+	left	2

## 3 ABSTRACT SYNTAX

The abstract syntax of AE is defined as follows:

Expressions  $e ::= n$  (Num)  
                   $| e + e$  (Add)      where      Integers  $n \in \mathbb{Z}$  (BigInt)  
                   $| e \times e$  (Mul)

## 4 SEMANTICS

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

$$\boxed{\vdash e \Rightarrow n}$$
$$\text{Num} \frac{}{\vdash n \Rightarrow n} \quad \text{Add} \frac{\vdash e_1 \Rightarrow n_1 \quad \vdash e_2 \Rightarrow n_2}{\vdash e_1 + e_2 \Rightarrow n_1 + n_2} \quad \text{Mul} \frac{\vdash e_1 \Rightarrow n_1 \quad \vdash e_2 \Rightarrow n_2}{\vdash e_1 \times e_2 \Rightarrow n_1 \times n_2}$$