

$(a+1) - b$

Example

$E \rightarrow E_1 + T$

$E \rightarrow E_1 - T$

$E \rightarrow T$

$T \rightarrow (E)$

$T \rightarrow id$

$T \rightarrow num$

{ $E.ast = mkNode("+", E_1.ast, T.ast)$ }

{ $E.ast = mkNode("-", E_1.ast, T.ast)$ }

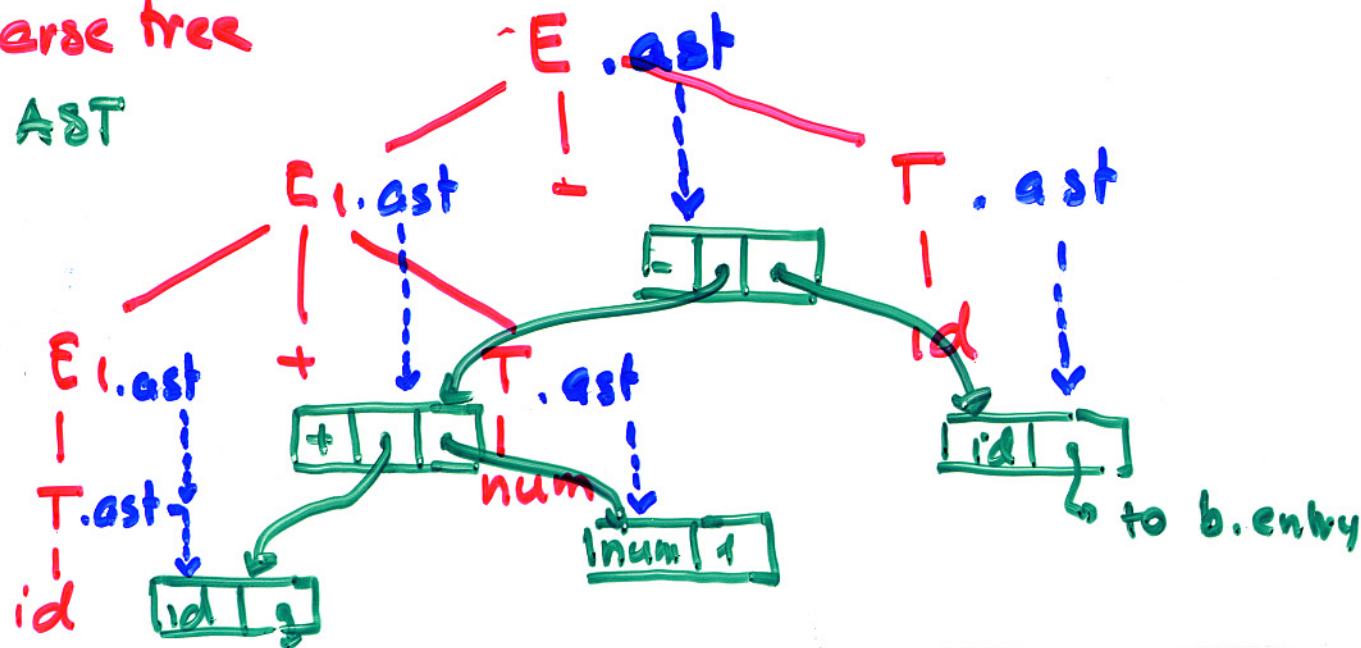
{ $E.ast = T.ast$ }

{ $T.ast = E.ast$ }

{ $T.ast = mkLeaf("id", id.entry)$ }

{ $T.ast = mkLeaf("num", num.val)$ }

red - parse tree
green - AST



Data Structures

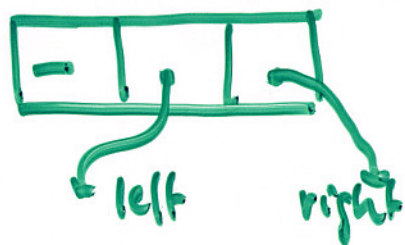
mkNode



mkLeaf



mkLeaf



Optional information is given by the class

Entry in ST

∈ AST

mkNode ('-', left, right)

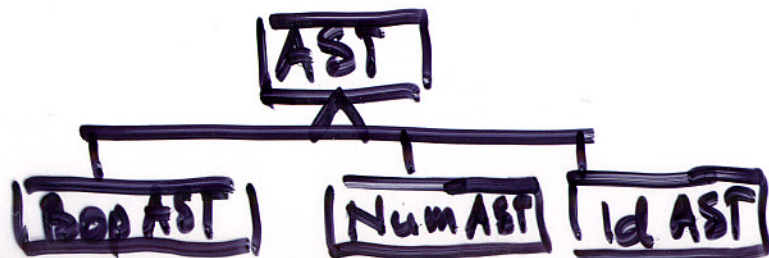
mkLeaf ('num', 4)

mkLeaf ('id', entry)

= new BopAST("-", left, right)

= new NumAST("num", 4) ∈ Entry Block

= new IdAST("id", entry)



BopAST BopAST (String, AST, AST)

NumAST NumAST (String, Int)

IdAST IdAST (String, Entry Block)

AST Tree

a+b*c

extend Simple Node
 ↳ children
 ↳ visitor
 ↳ jit This

```
ASTStart() #Start : 1
Exp() ";" {return jitThis}
```

default void

```
void Exp(): 1 {
    AddExp() #void
}
```

```
void AddExp(): 1 {
    (MulExp() ("+" | "-") MulExp()) *
    #Add (> 1) #void
}
```

```
void MulExp(): 1 {
    UnExp() ("*" | "/" | "^") UnExp() *
    #Mul (> 1) #void
}
```

```
void UnExp(): 1 { Identifier() | Integer() }
```

```
void Identifier() {Token t; if (t == <IDENTIFIER>) {jitThis.
    set Name(t.name);}
```

