< <relarith primary>

↓

arith exp

<arithterm> <addop> <arithterm>

|

arithprimary

|

arith factor

|

id

|

d

+

arith primary

arith factor

id

decimal number

↓

2

<rel_expr> $\xrightarrow{1}$ <rel_term> {(==, !=) <rel_term>}

<rel_expr> → <relterm> {(==, !=, _relterm)}

→ <rel primary> {(>, >=, <, <=) <rel-primary>} {= !=}

→ <arith-exp> {> < >= <=}

→ rel

new office hours

parser.cpp

M 12-130
T 2-330

ExprN

ExprNode tokea Token

virtual void Evaluate(SymTab &Symtab

Token
need look ahead token

get token / unget token
let's us look ahead 1 token

Attribute Grammars
- mechanism for putting constraints on grammar

Project posting today

Finished Ch. 4
Ch5

460 2/18/19

parsing for statements account for semicolons but not eol

---

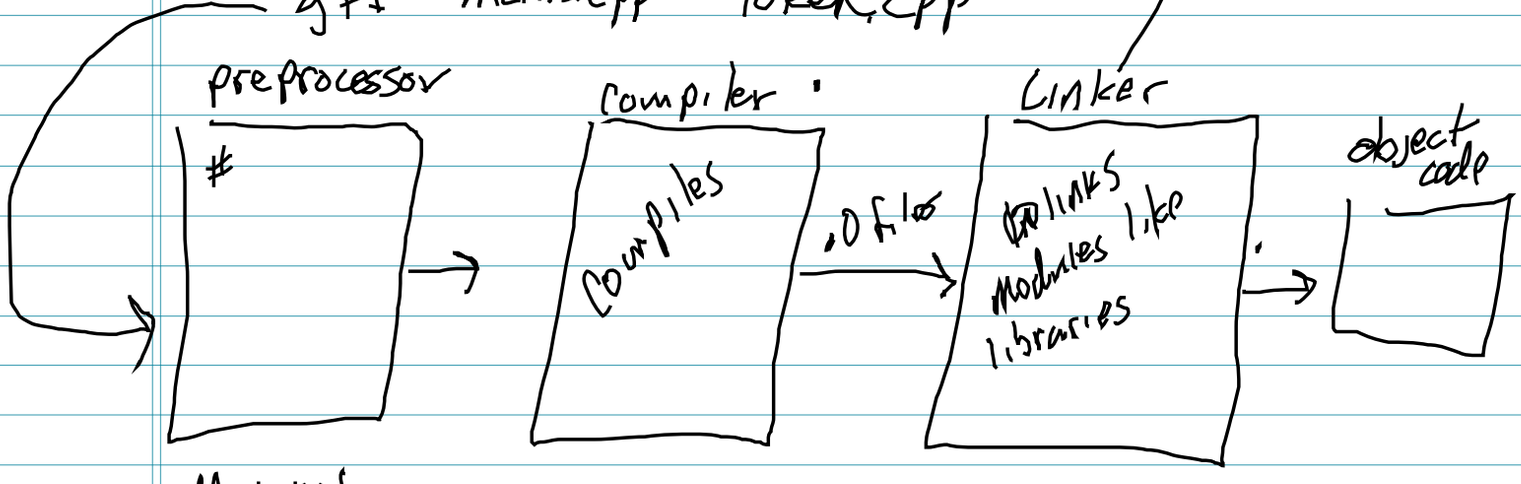
token class
need new variable
string type for relational operators

Ch. 5

Note: How compilers compile

links external references

Ex
g++ main.cpp Token.cpp



eg. add text of include file to main

We know most of Ch 5, just putting all together into context

Names, Bindings, scope

Names: identifiers reference nameable entities
- variable
- function
- key word

Binding - association between entity and attributes
- data types
- between operator / variable

Scope - range of statements w/in which a name is visible