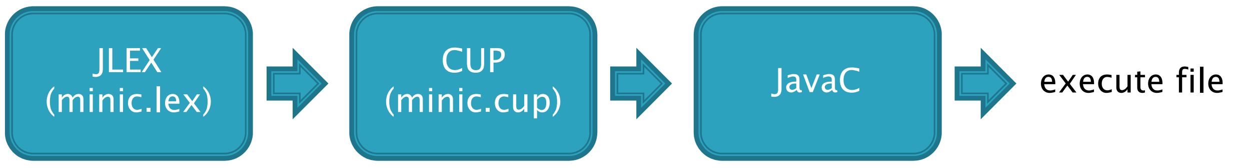
PA1&2

Build Process



- You must remove all conflict/reduce

```
$ java java_cup.Main –parser minic –expect 3
                –dump_grammar –
```

- Print productions (reverse RM derivation) and line no. and character position of the text.

AST Absyn.java

- 4 abstract classes & 2 enum
  - Stmt : Statements
  - Expr : Expressions
  - Dec: Declarations
  - Var: Variables – Simple & Array
  - Type : Types – enum INT, FLOAT
  - Binop : Binary operations – enum PLUS, …NEQ

AST.java

- 19 classes(?)
  - PRORAM, DECLARATION, IDENTIFIER, FUNCTION, PARAMETER, COMPOUNDSTMT, STMT, ASSIGN, CALL, ARGLIST, WHILE_S, FOR_S, IF_S, EXPR, UNOP,BINOP, ID_S, CASELIST, SWITCH_S ….

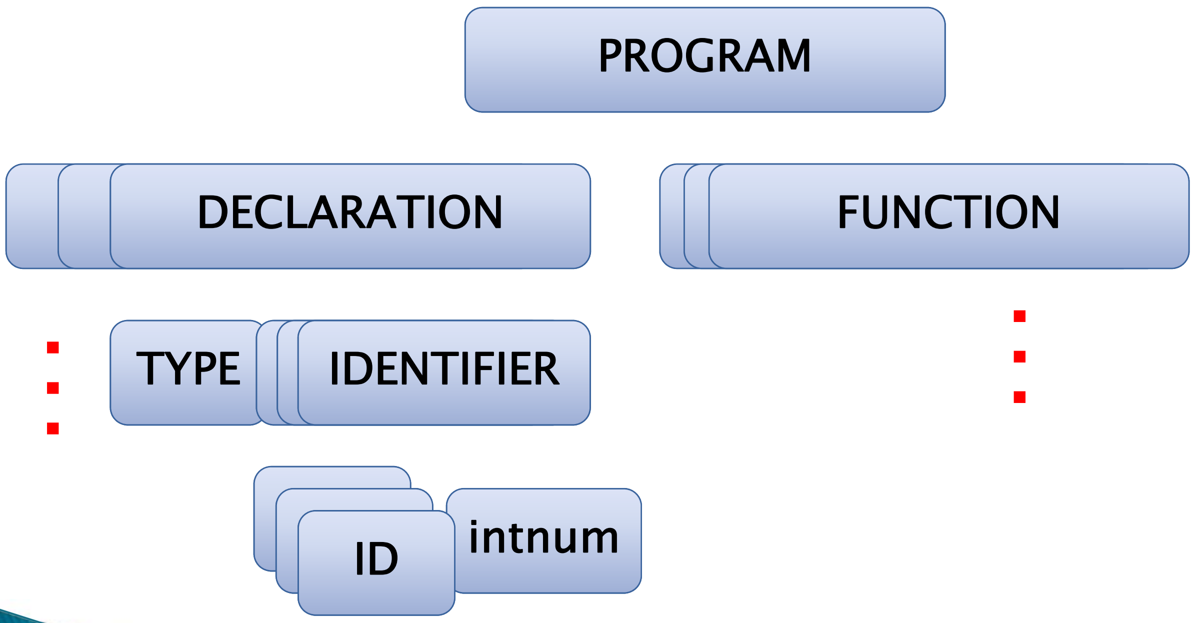
  - Each class is a node of AST
  - Consult with Absyn/*.java

AST

- You may use nodes in Absyn/* to build AST
- Declare root node
  - minic parser = new minic(new Yylex(inp,errorMsg), errorMsg);
  - parser.parse()
  - Absyn(?) = parser.parserResult;
- We will grade your submission by semi-pretty print;

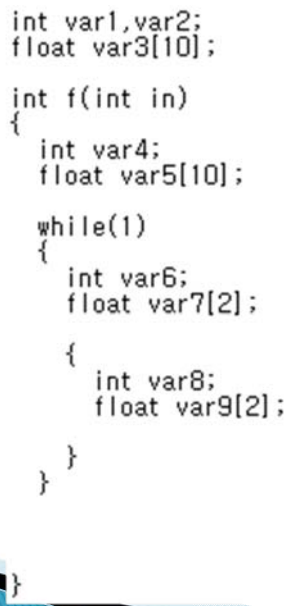
Ex) AST

PROGRAM absyn;



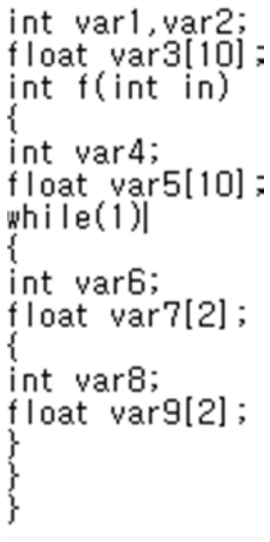
Print AST

- Write your own DFS_print to print out similar to C
- Ex:



```
int var1,var2;
float var3[10];

int f(int in)
{
  int var4;
  float var5[10];

  while(1)
  {
    int var6;
    float var7[2];

    {
      int var8;
      float var9[2];

    }
  }

}
```

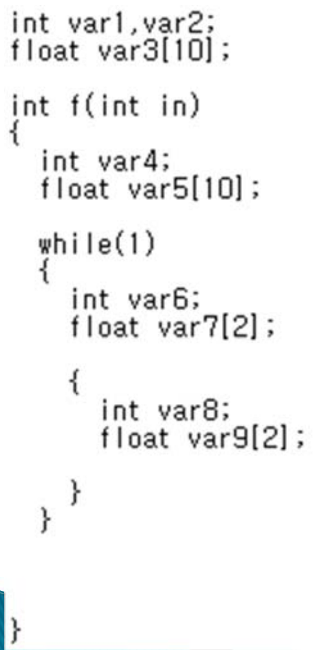
Parsing & DFS_print



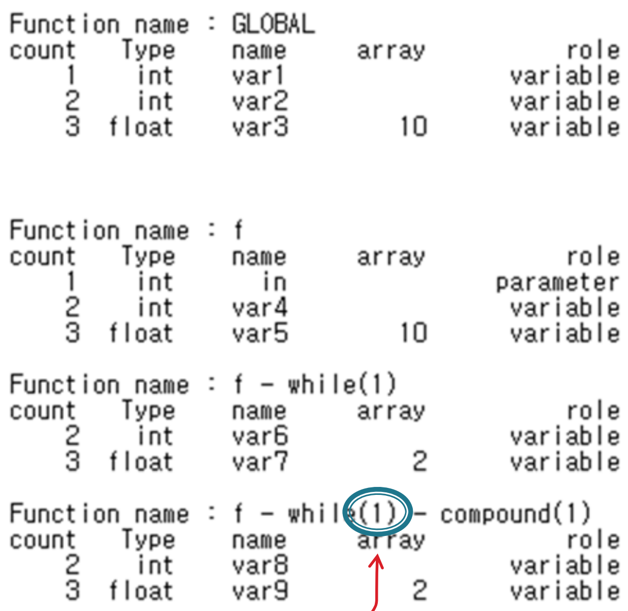
```
int var1,var2;
float var3[10];
int f(int in)
{
int var4;
float var5[10];
while(1)
{
int var6;
float var7[2];
{
int var8;
float var9[2];
}
}
}
```

Sample output

Ex : Symbol-table (example)



```
int var1,var2;
float var3[10];

int f(int in)
{
  int var4;
  float var5[10];

  while(1)
  {
    int var6;
    float var7[2];

    {
      int var8;
      float var9[2];

    }
  }

}
```



```
Function name : GLOBAL
count    Type    name     array          role
    1     int    var1                variable
    2     int    var2                variable
    3   float    var3        10      variable


Function name : f
count    Type    name     array          role
    1     int      in               parameter
    2     int    var4                variable
    3   float    var5        10      variable

Function name : f - while(1)
count    Type    name     array          role
    2     int    var6                variable
    3   float    var7         2      variable

Function name : f - while(1) - compound(1)
count    Type    name     array          role
    2     int    var8                variable
    3   float    var9         2      variable
```

first while_stmt in f function

- Print out “Symbol-Table” similar to the form in the previous slide – not strictly the same
- But must include <location, type, name, array, role>

- The name of output files
  - Print AST (C-like program) -> “tree.txt”
  - Print symbol-table -> “table.txt”

submit

- Tar the following files to “yourstudentid.zip” (your eclipse workspace, sample test input…)
  - Absyn/*.jav
  - *.cup, *.lex
  - Makefile
  - Readme.txt
  - example.c
  - your code (printAST.java ,printSymtab.java ….)

- Due by April 15 (1st step)
- Due by May 11 (2nd step)

Grade guide

- 20(?) test cases
- How correctly you generate following outputs
  - Grammar (include shift/reduce error)
  - Symbol table
  - AST
- Please submit your program with the README

TIP!

- You may need to define some types for AST node
- You can easily get an information from Google