Quiz 3

Name: ____________________ Student ID: ______________

Do Not Open Until Instructed!

Before the Quiz starts:

- Read all of the instructions on this page
- Write your name and student ID on this page
- Locate your page of notes, if you have one
- Prepare your writing materials
- Put all other materials away

After the Quiz starts:

- Write your student ID (**not** your name) on all subsequent pages
- Announcements / corrections will appear on the projector
- Turn in your quiz and note page to Drew when finished.
- After the quiz time expires, answers may be presented but no new material will be given.

The quiz consists of 5 questions. You will have *35 minutes* to complete all questions. Work quickly and move on if you are stuck. If you'd like to pass the time before the quiz starts or before it ends, feel free to draw something NOT SPOOKY in the box below:

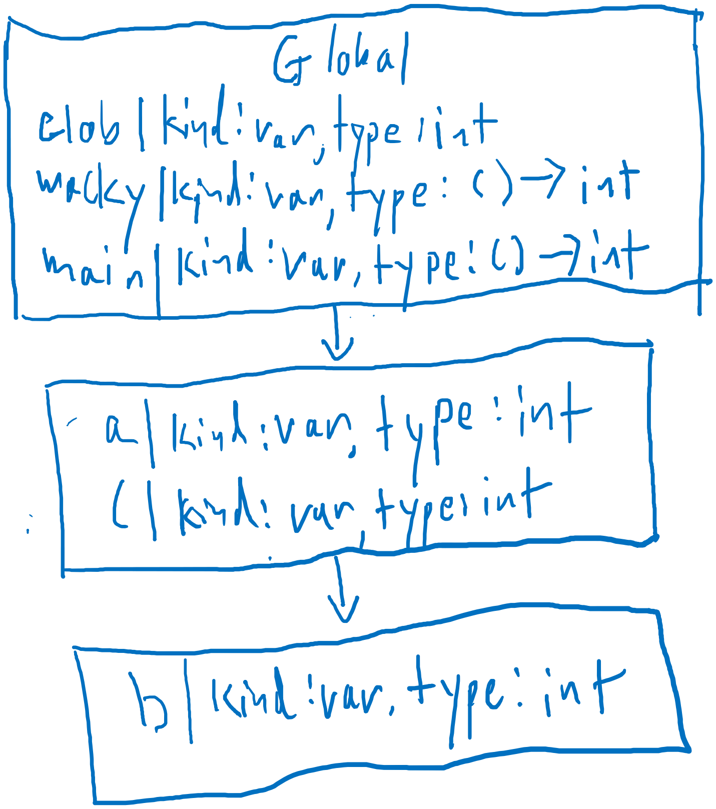
Student ID: ____________________

Question 1 (2 Points)

Consider the following program snippet:

```
 1.  int glob;
 2.  int wacky(){
 3.     glob = 9;
 4.     return glob + 2;
 5.  }
 6. int main(){
 7.     int a;
 8.     int c;
 9.     a = 12;
10.     if (glob > 4){
11.         int b;
12.         int d;
13.         b = a + 4 * 18;
14.     }
15.     c = a;
16.     return c;
17. }
18. int glob2;
```

Draw a representation of what the symbol table will contain after processing the program's AST for lines 1-11, inclusive (i.e. what does the symbol table look like after line 11?). Use the list-of-hashtables implementation of symbol tables.



Global

glob | kind: var, type : int

wacky | kind: var, type : () → int

main | kind: var, type: () → int

a | kind: var, type : int

c | kind: var, type: int

b | kind: var, type: int

Student ID: ____________

Question 2 (2 Points)

(Refer to the reference page for this question)
Assume that stmtA immediately preceeds stmtB. Fill in the type that would be assigned to each corresponding node during typechecking.

- If the node does not have a type, put "N/A" on the line or leave it blank.
- If the node is assigned a non-error type, put that type on the line.
- If the node generates a type error, put ERROR on the corresponding line below.
- If the node causes an error report to be output, also put REPORT on the line.

e.g, AST node 2 has type `int`, no errors, no reports, so write "int" for line 2.

Note that you do not have to write out entries for nodes without a number, though they MAY have a type. Do not ask why the question is structured this way.

1: int

2: int

3: int

4: N/A

5: N/A

6: N/A

7: error REPORT

8: bool

9: int

10: N/A

11: int

12: int→int

13: <int>

14: int

15: int→int

16: int

Question 3 (2 Points)

(Refer to the reference page for this question)
Assume that any errors in the program have been fixed, and all variables are now of type `int`. Draw the 3AC code that would be generated for stmtA and stmtB. If you need to make any additional assumptions about the program, explain your assumptions. You may use the following 3AC instructions, plus any others you need (explain the functionality of the instruction if you add it, it must obey the rules of 3AC).
3AC reference:

```
x = y op z
x = y
ifz ( x ) goto L
goto L
call p
retrieve x
return x
enter p
get_arg k, x (where k is the arg position)
set_arg k, x (where k is the arg position)
leave
label L
```

```
if(c==4){
  int c;
    c=7;
}

addOne(addOne(c))
```

```
  cond = c ==4
  ifz (cond) goto After
  c_inner =7
After: set_arg 1, c
       call addOne
       retrieve temp
       set_arg 1, temp
       call addOne
```

Student ID: ________________

Question 4 (2 Points)

Check out this simple language of declarations (decls) and uses (statements):

```
program -> LCURLY declList stmtList RCURLY #1
declList -> declList decl                  #2
      | epsilon                            #3
stmtList -> stmtList stmt                  #4
      | epsilon                            #5
decl -> BOOL ID SEMICOLON                  #6
stmt -> ID ASSIGN exp SEMICOLON            #7
exp -> exp EQUALS exp                      #8
     | ID                                  #9
     | BOOLLITERAL                         #10
```

Let the SDT goal be to interpret the program as **the set of identifier names that are used without being declared.** Fill out the SDT rules to meet this goal.

#1: program.trans = stmtList.trans − declList.trans

#2: declList$_1$.trans = declList$_2$.trans ∪ decl.trans

#3: declList$_1$.trans = { }

#4: stmtList$_1$.trans = stmtList$_2$.trans ∪ stmt.trans

#5: stmtList$_1$. trans = { }

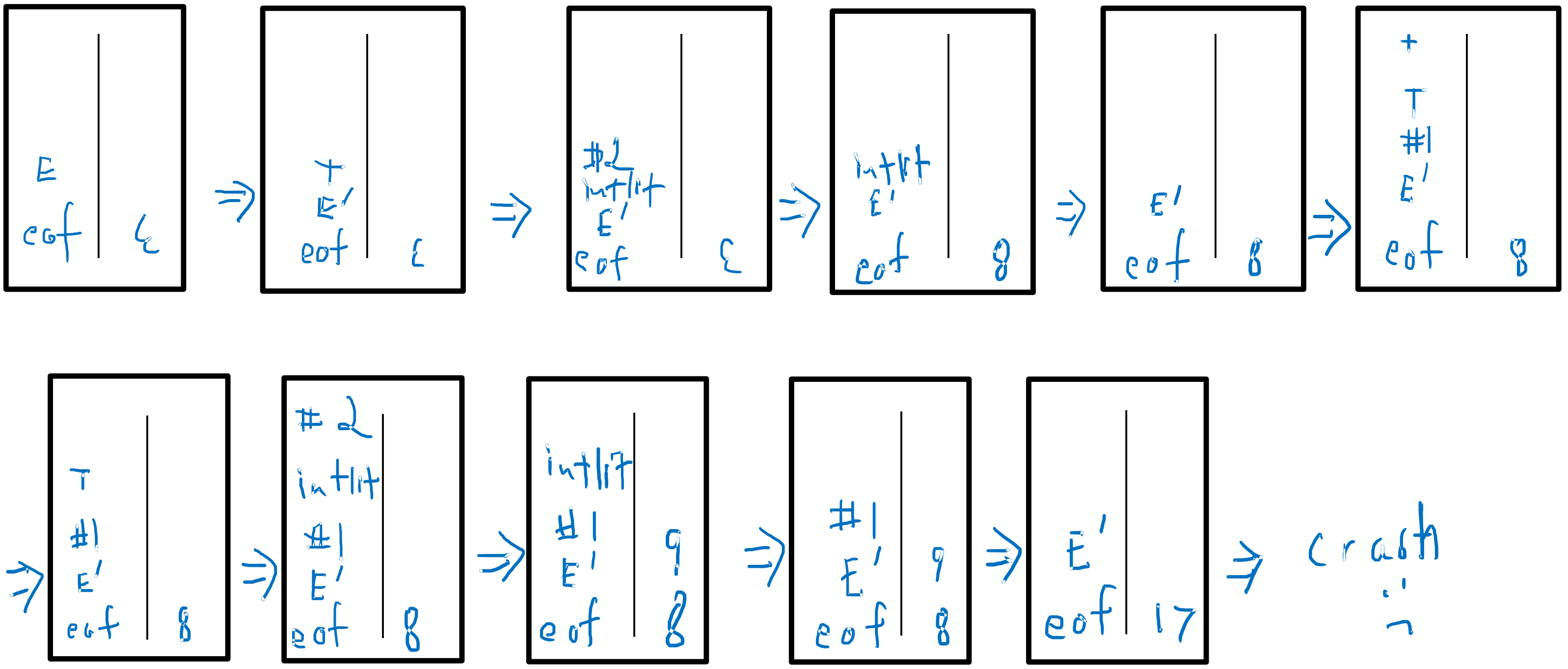
#6: decl$_1$.trans = ID.value

#7: stmt$_1$.trans = ID.value ∪ exp$_1$.trans

#8: exp$_1$.trans = exp$_2$.trans ∪ exp$_3$.trans

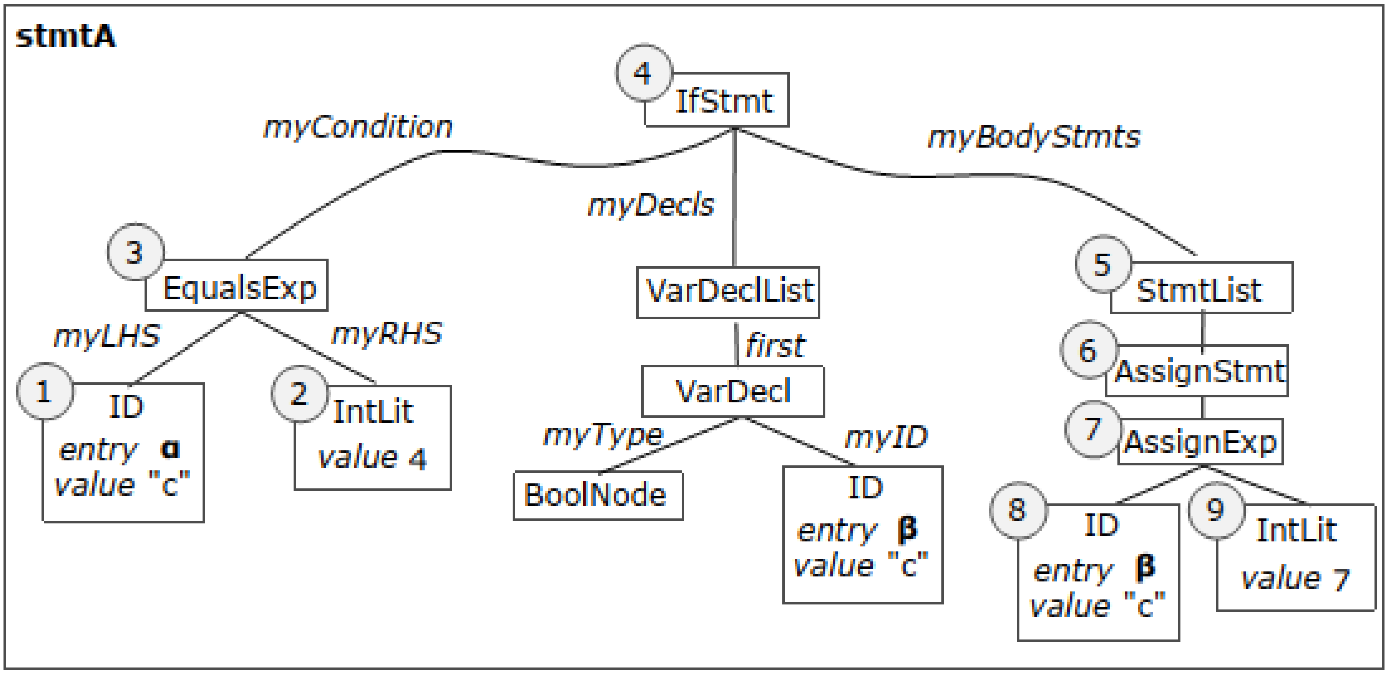
#9: exp.trans = { }

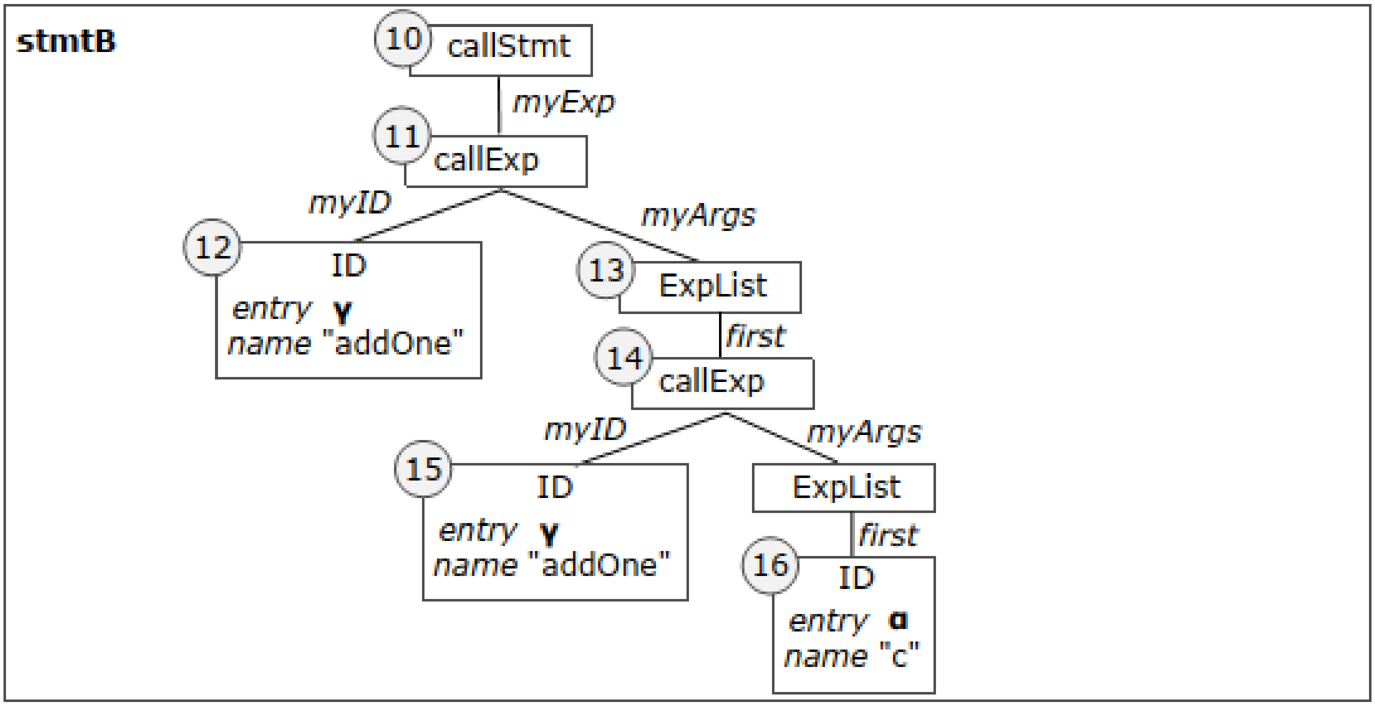
#10: exp.trans = { }


E
eof
ε
T
E'
eof
ε
#2
intlit
E'
eof
ε
intlit
E'
eof
8
E'
eof
8
+
T
#1
E'
eof
8
T
#1
E'
eof
8
#2
intlit
#1
E'
eof
8
intlit
#1
E'
eof
9
8
#1
E'
eof
9
8
E'
eof
17
crash
:-(

Student ID: ____________________

REFERENCE PAGE. PUT YOUR KUID ON THIS PAGE AND TURN IT IN WITH YOUR QUIZ





α kind: VarDecl
type: Int

β kind: VarDecl
type: Bool

γ kind: FnDecl
type: Int -> Int

Detached SymbolEntries
(note that this is not a symbol table.
These entries are not necessarily
in the same scope)