

Plan for today

Translating program features to the Tree IR

Last time

- arithmetic and some binary operations
- assignments
- accesses to array variables
- array assignment

This time:

- functions in the MiniJava run-time library
- allocating class instances
- allocating arrays
- length expression
- this expression
- call expressions
- less than operator
- if and while statements

CS453 Lecture

More translation to Tree IR

2

Functions in MiniJava runtime library

Assume the following will be available

- void printint(int x)
 - Prints given integer followed by a newline
- void * halloc(int n)
 - Allocates n consecutive bytes in the heap and returns the address of the first byte allocated.
- void initArray(int [] a, int n)
 - Assumes that a points at an array allocation of size (n+1)*wordsize
 - Puts the length n into a[0]
 - Initializes a[1] through a[n] to zero

Calling the functions within Tree IR

- Exp initExp = frame.externalCall("initArray", arglist);
- Frame.externalCall places an “_” in front of all function names

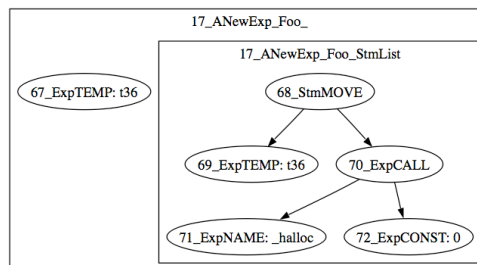
CS453 Lecture

More translation to Tree IR

3

Allocating class instances

```
class Foo { ... }  
... new Foo ...
```



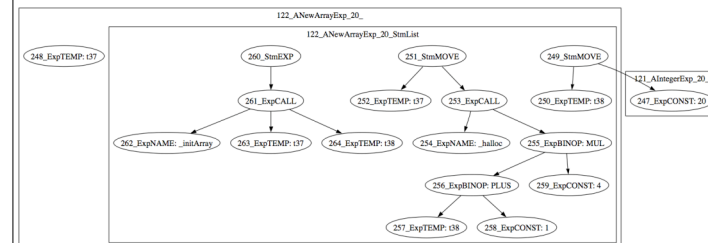
CS453 Lecture

More translation to Tree IR

4

Allocating an array

```
int [] y;  
y = new int [20];
```



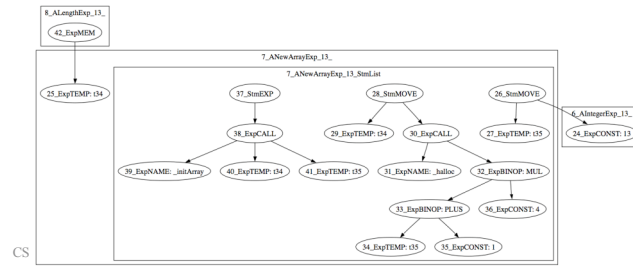
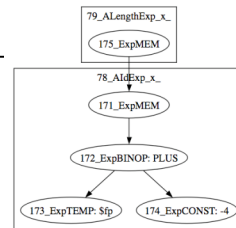
CS453 Lecture

More translation to Tree IR

5

Length expression

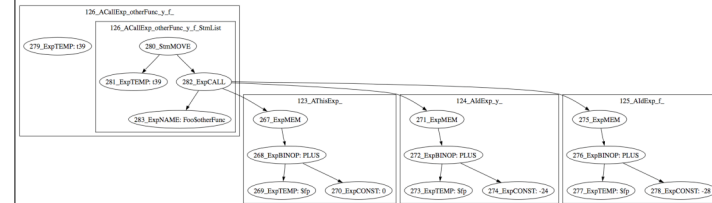
```
int [] y;
... y.length ...
... new int [13] . length ...
```



CS

Call expression

```
... this . otherFunc(y,f)...
```



CS453 Lecture

More translation to Tree IR

7

StmCJUMP

```
public class StmCJUMP extends Stm {
    public int relOp;
    public Exp left, right;
    public Label iftrue, iffals;
    public StmCJUMP(int rel, Exp l, Exp r, Label t,
        Label f) {
        ... }
}
```

```
public final static int
EQ=0,
NE=1,
LT=2,
GT=3,
...
```

CS453 Lecture

More translation to Tree IR

8

Tree intermediate representation

ExpCONST(int i) - The integer constant i

ExpNAME(Label n) - Constant address. Corresponds to label in assembly.

ExpTEMP(Temp t) - Temporary t. "Infinite" Temps in Tree IR.

ExpBINOP(int binop, Exp left, Exp right) - left binop right

ExpMEM(Exp exp) - If left child of move, then store into address calculated by exp. Otherwise, fetch value at address calculated by exp.

ExpCALL(Exp func, List<Exp> args) - evaluate func to find func address, then evaluate args left to right.

StmMOVE(Temp t, e) - Eval e and put result in t.

StmMOVE(ExpMEM(e1), e2) - Eval e2 and store into address e1.

StmEXP(Exp e) - Eval e and ignore result

StmJUMP(Label targ) - Transfer control to given label.

StmCJUMP(int rel, Exp l, Exp r, Label t, Label f) - if l rel r then goto t else goto f

StmLABEL(Label l) - Label in assembly.

CS453 Lecture

More translation to Tree IR

9