

## Plan for today

### PA7 overview

- MethodSTE.java got left out

### Expression Tree intermediate representation

#### Possible approaches for performing translation

- All visitor methods return a Translate.Exp reference
- Visitor methods return void, but map AST nodes to Tree.Exp nodes and a list of Tree.Stm

#### Translating program features to the Tree IR

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

CS453 Lecture

PA7 overview

1

## PA7 Overview

### TranslateToIRTree.java

- driver for PA7

### ast\_visitors/

- BuildSymTable.java has one new piece
- Translate.java
- Frag.java

### Tree/\*

### Temp/

- TempMap.java interface

### Frame/

- new functionality added to Frame.java interface and Access.java interface

### Mips/

- MipsFrame.java implements the new functionality
- InFrame.java implements the Tree.Exp exp(Tree.Exp) method

### SymTable/

- VarSTE, LocalSTE, and MemberSTE implement Tree.Exp exp(Tree.Exp)
- ClassSTE has getNumMembers() method
- MethodSTE has new constructor

CS453 Lecture

PA7 overview

2

## 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

PA7 overview

3

## ExpBINOP

```
public class ExpBINOP extends Exp {
    public int binop;
    public Exp left, right;
    public ExpBINOP(int b, Exp l, Exp r) {
        binop=b; left=l; right=r; }
}
```

```
public final static int
    PLUS=0,
    MINUS=1,
    MUL=2,
    DIV=3,
    AND=4,
    OR=5,
    ...
    XOR=9;
```

CS453 Lecture

PA7 overview

4

### Access to array variable

```
int [] y;
```

```
... y[77]...
```

