CMPE 152: Compiler Design

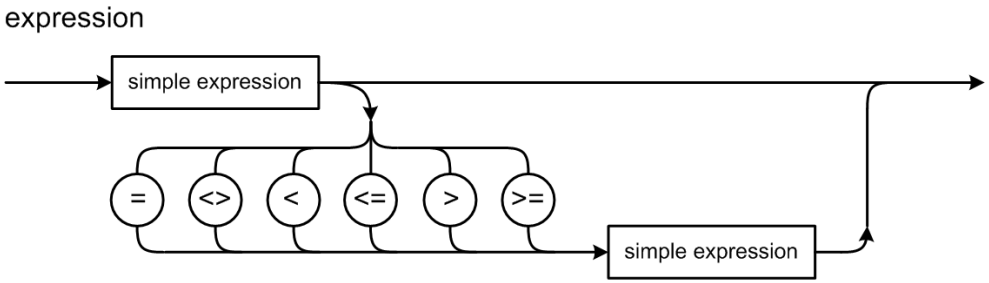
November 7 Lab

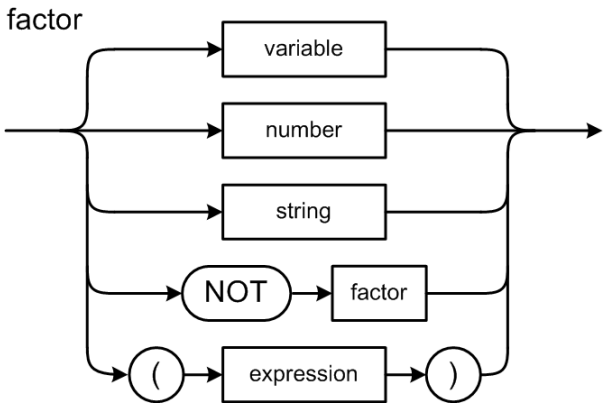
Department of Computer Engineering
San Jose State University

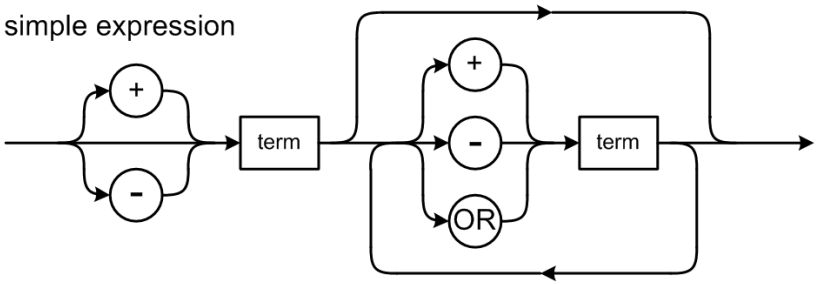
Fall 2017
Instructor: Ron Mak

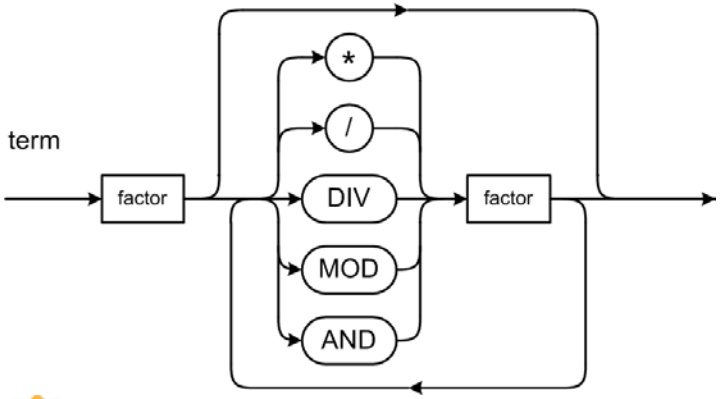
www.cs.sjsu.edu/~mak

Expression Syntax Diagrams









- What code should we generate for a relational expression?

Relational Expressions

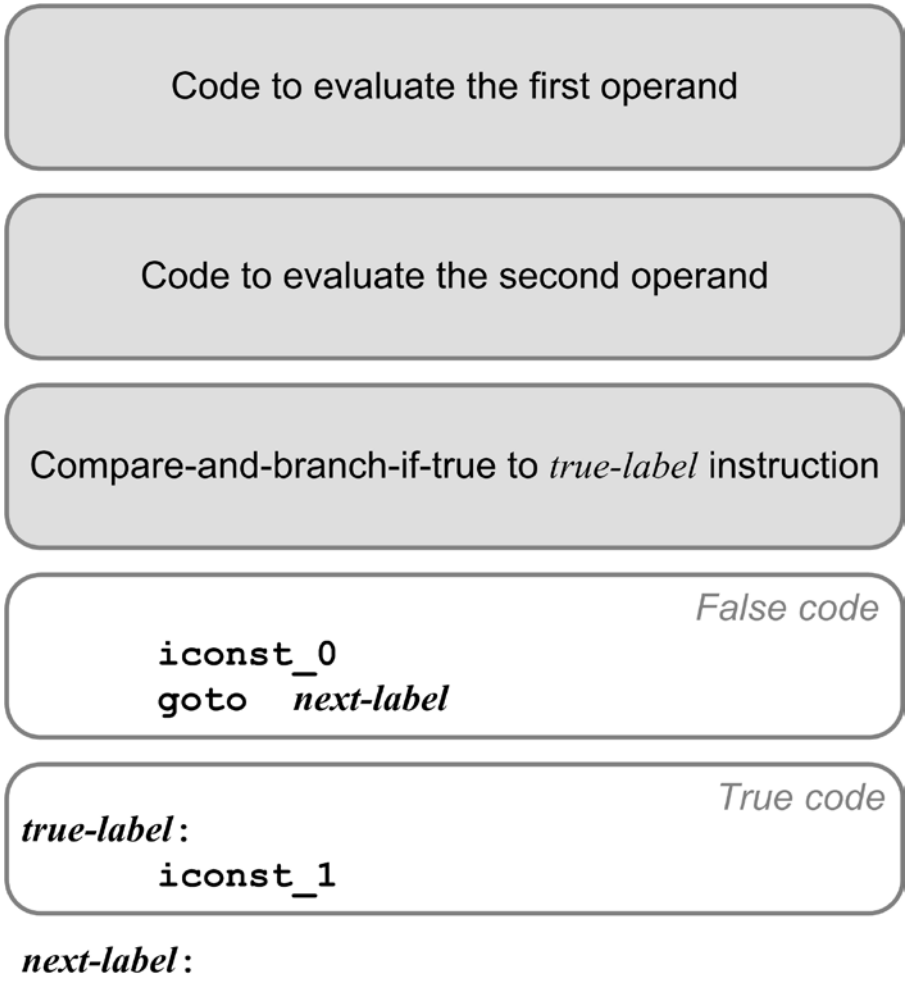
- Suppose **i** and **j** are local integer variables, and that:
  - **i** ➔ slot #0
    j ➔ slot #1
- 0 represents false and
  1 represents true.
- For the expression **i < j** leave either 0 or 1 on top of the operand stack:

The code in **red** are the only parts that change based on the expression.

```
   iload_0          ; push the value of i (slot #0)
   iload_1          ; push the value of j (slot #1)
   if_icmplt L003   ; branch if i < j
   iconst_0         ; push false
   goto L004        ; go to next statement
L003:
   iconst_1         ; push true
L004:
```

Your code generator also needs to emit labels.

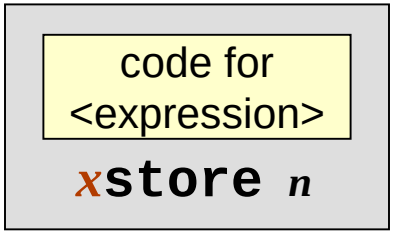
Relational Expression Code Template




```
    iload_0
    iload_1
    if_icmplt L003
    iconst_0
    goto L004
L003:
    iconst_1
L004:
```

Assignment Statement Code Template

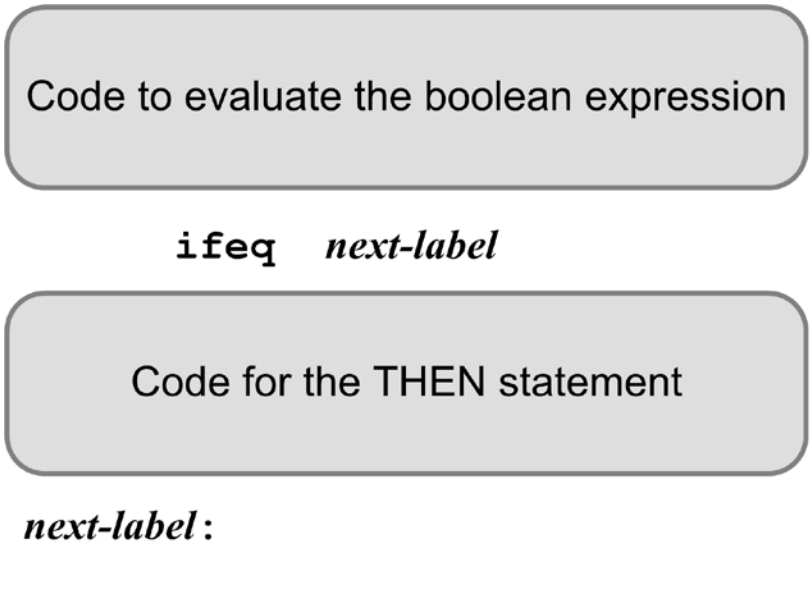
- The code template for an assignment statement to a local variable <variable> := <expression>

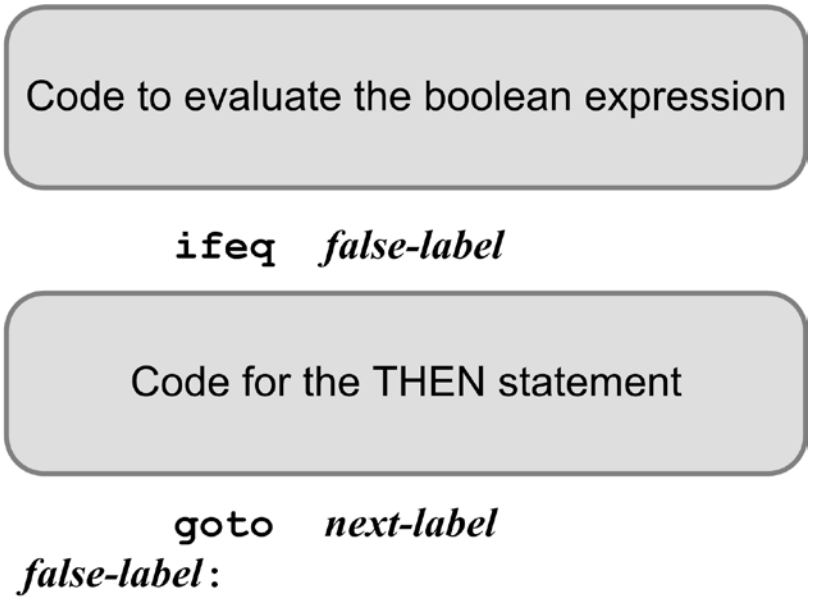


Where *x* is **i**, **l**, **f**, or **d** depending on the type of the computed value of <expression>.

- You can generate a shortcut store instruction such as **istore_3** (3 is a slot number) whenever possible.

`IF` Statement Code Templates





- The code that evaluates the boolean expression leaves either 0 (false) or 1 (true) on top of the operand stack.
  - **`ifeq`** branches if [TOS] is 0 (the expression is false)

Example: **IF** Statement

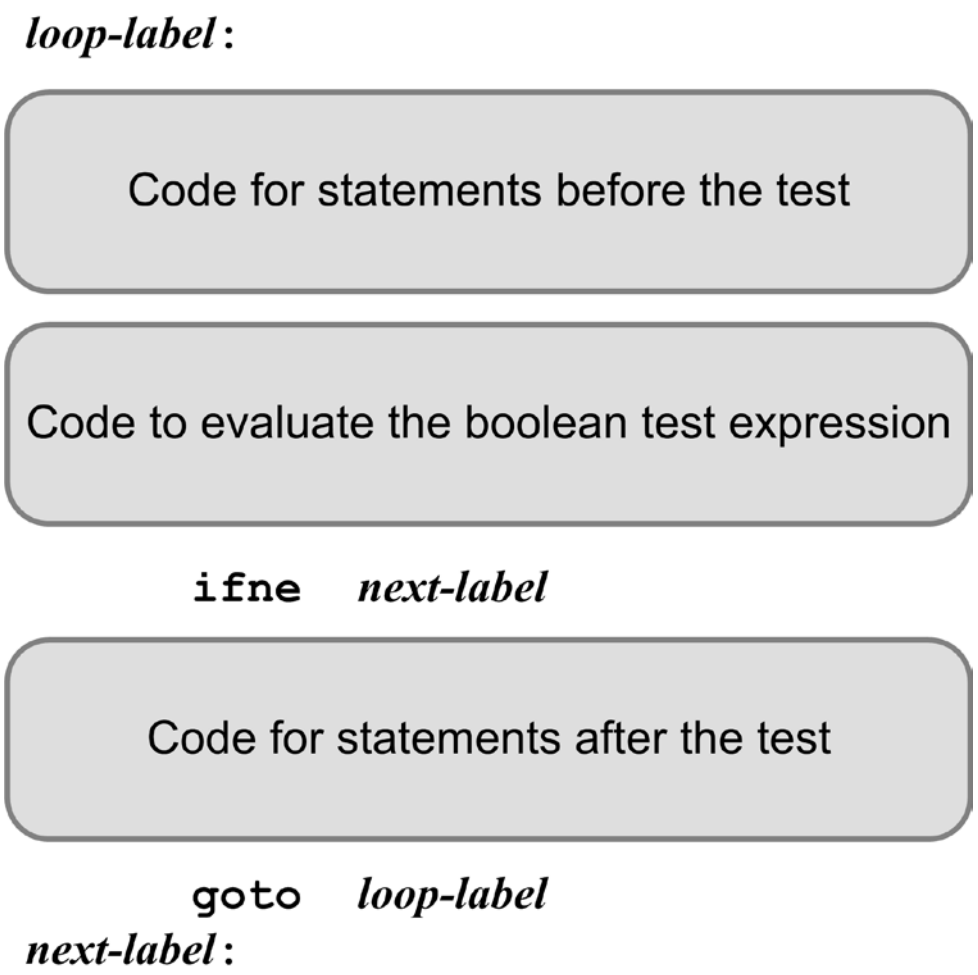
```
PROGRAM IfTest;
VAR
    i, j, t, f : integer;

BEGIN {IF statements}
    ...
    IF i < j THEN t := 300;

    IF i = j THEN t := 200
             ELSE f := -200;
    ...
END.
```

```
                getstatic       iftest/i I
                getstatic       iftest/j I
                if_icmplt       L002
                iconst_0
                goto    L003
L002:
                iconst_1
L003:
                ifeq    L001
                sipush  300
                putstatic       iftest/t I
L001:
                getstatic       iftest/i I
                getstatic       iftest/j I
                if_icmpeq       L005
                iconst_0
                goto    L006
L005:
                iconst_1
L006:
                ifeq    L007
                sipush  200
                putstatic       iftest/t I
                goto    L004
L007:
                sipush  200
                ineg
                putstatic       iftest/f I
L004:
```

Looping Statement Code Template



- The code that evaluates the boolean expression leaves either 0 (false) or 1 (true) on top of the operand stack.
  - `ifne` branches if [TOS] is **not 0** (the expression value is **true**)
- There might not be any code before or after the test.

Example: Newton's Square Root Function

```
                  #0
FUNCTION sqrt(x : real) : real;

VAR
  #1 i : integer;
  #2 root : real;

BEGIN
    i := 0;
    root := x;

    REPEAT
        root := (x/root + root)/2;
        i := i + 1;
    UNTIL i > 10;

    sqrt := root;
END;
```

```
    iconst_0
    istore_1   ; i := 0
    fload_0
    fstore_2   ; root := x
L000:
    fload_0    ; x
    fload_2    ; root
    fdiv       ; /
    fload_2    ; root
    fadd       ; +
    fconst_2   ; 2.0
    fdiv       ; /
    fstore_2   ; ==> root
    iinc 1 1   ; i := i + 1;
    iload_1    ; i
    bipush 10  ; 10
    if_icmpgt L001  ; if i > 10 goto L001
    iconst_0        ; false
    goto L002
L001:
    iconst_1        ; true
L002:
    ifne L001  ; if true goto L003
    goto L000
L003:
    fload_2
    freturn    ; return root
```

Example: **FOR** Statement

```
PROGRAM ForTest;
VAR
    j, k, n : integer;

BEGIN {FOR statements}
    ...

    FOR k := j TO 5 DO BEGIN
        n := k;
    END;

    ...
END.
```

Remember that **program variables** are translated into **Jasmin static fields**, and so they have names, not slot numbers.

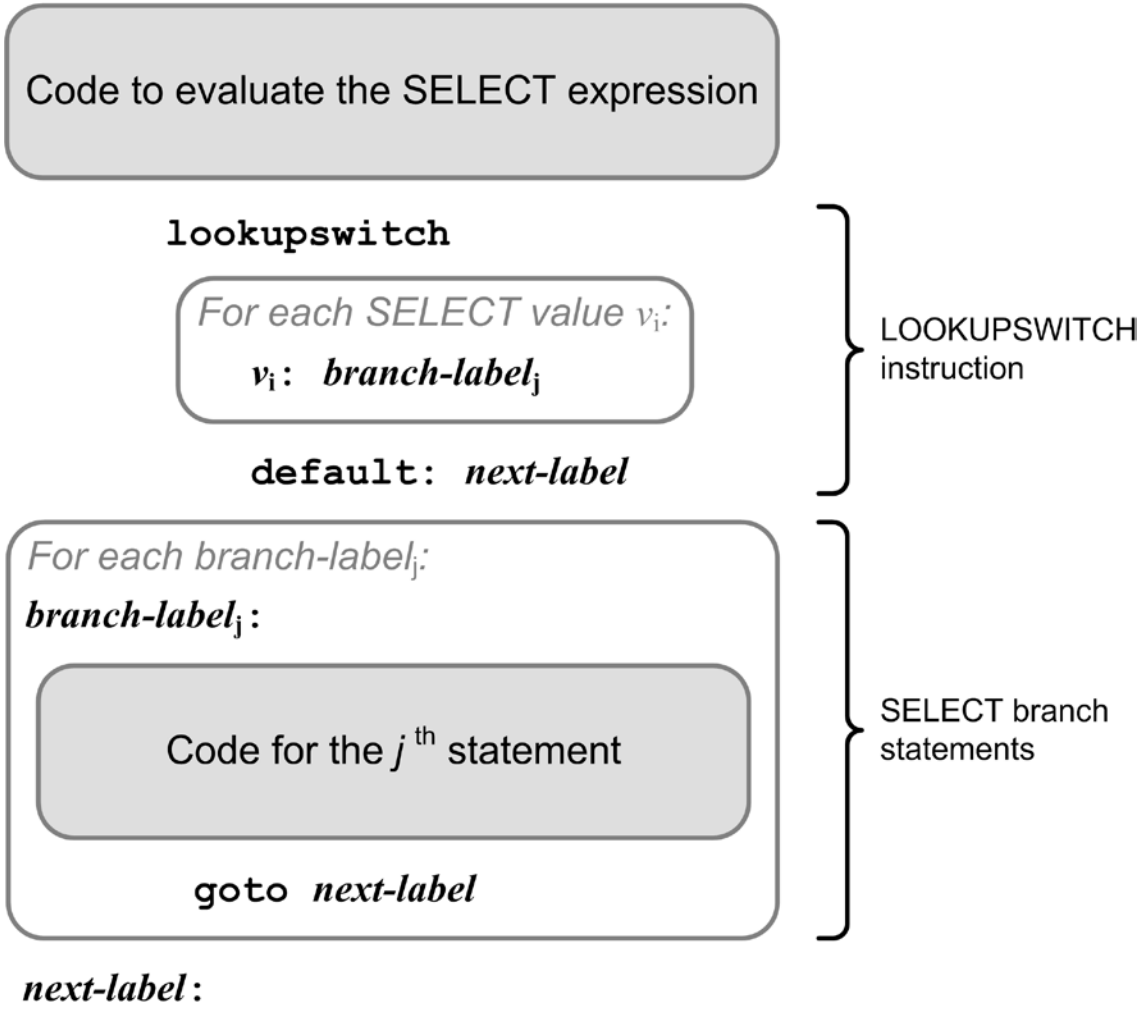
```
        getstatic       fortest/j I
        putstatic       fortest/k I
L001:
        getstatic       fortest/k I
        iconst_5
        if_icmpgt       L003
        iconst_0
        goto    L004
L003:
        iconst_1
L004:
        ifne    L002

        getstatic       fortest/k I
        putstatic       fortest/n I

        getstatic       fortest/k I
        iconst_1
        iadd
        putstatic       fortest/k I
        goto    L001
L002:
```

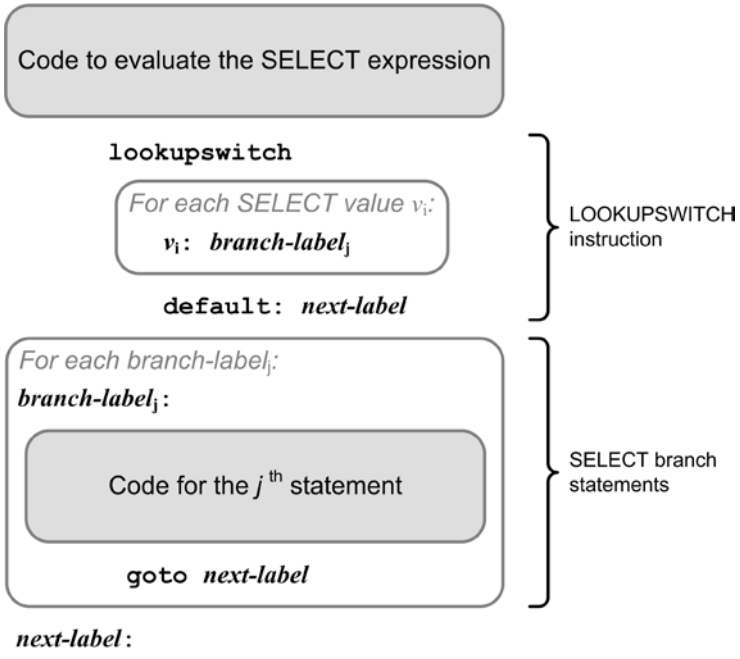
This is code emitted for a general > test. It can be much improved!

SELECT Statement



Example: **CASE** Statement

```
VAR i, j : integer;
...
CASE i OF
  100,105:     j := 1000;
  200,256,282: j := 2000;
END
```




```
    iload_0  ; i

    lookupswitch
      100: L010
      105: L010
      200: L020
      256: L020
      282: L020
      default: L099
L010:
    sipush 1000
    istore_1  ; j := 1000
    goto L099
L020:
    sipush 2000
    istore_1  ; j := 2000
    goto L099
L099:
```