

Review of basic Java 2

(contains some material from slides accompanying
Horstmann: Java for Everyone: Late Objects,
John Wiley and Sons Inc)

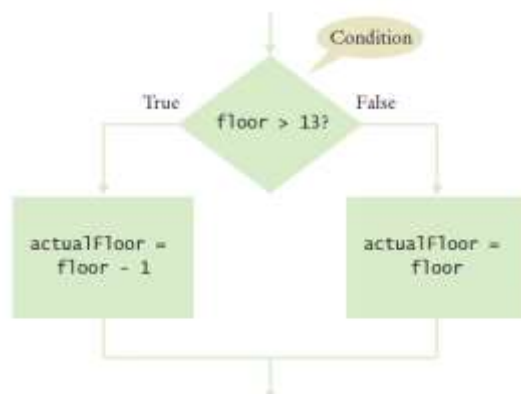
Overview

- Review of basic Java:
 - If statements
 - Boolean values and conditions
 - Loop statements
- With a focus on:
 - Formal syntax definition
 - Compiling schemes

Flowchart of the two-branch if statement

One of the two branches is executed once:

True (if) branch or False (else) branch



```
int actualFloor;  
  
if (floor > 13)  
{  
    actualFloor = floor - 1;  
}  
else  
{  
    actualFloor = floor;  
}
```

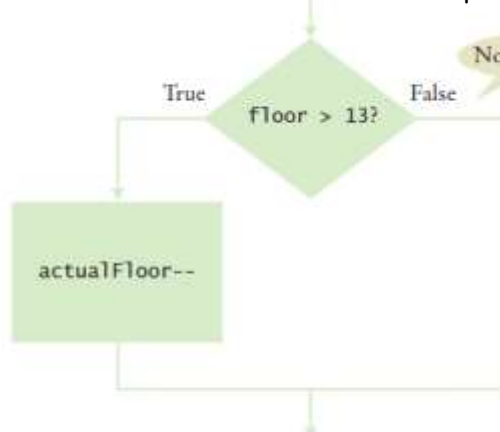
CSCU9A2 Review of basic Java 2
© University of Stirling 2017

3

Flowchart with only true branch – one-branch if

An if statement does not need a 'False' (else) branch

Choice between True branch and "skip"



```
int actualFloor = floor;  
  
if (floor > 13)  
{  
    actualFloor--;  
} // No else needed
```

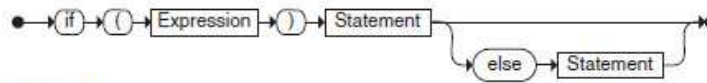
CSCU9A2 Review of basic Java 2
© University of Stirling 2017

4

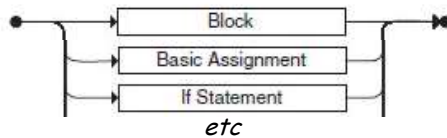
If statement: syntax details

Railroad diagrams from
<http://markettorrent.com/topic/9359>

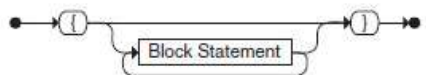
If Statement



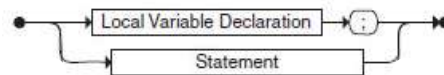
Statement



Block



Block Statement



CSCU9A2 Review of basic Java 2
 © University of Stirling 2017

5

If statement: syntax details

Reference syntax rules: (slightly condensed and simplified)

IfThenStatement:

if (Expression) Statement

IfThenElseStatement:

if (Expression) Statement else Statement

(Expression must have type boolean, or a compile-time error occurs)

Statement:

Block
 Assignment
 WhileStatement
 IfThenStatement
 ...

Block:

{ BlockStatements_{opt} }

BlockStatement:

LocalVariableDeclarationStatement
 Statement

CSCU9A2 Review of basic Java 2
 © University of Stirling 2017

6


If statement: implementation in machine code

Example: A one-branch if statement

```
if ( a != b )      (!= is very convenient for the  
{ ... "then" branch...    Brookshear machine! )  
}  
...
```

Could compile into Brookshear machine code:

```
MOV [a] -> R0  
MOV [b] -> R1  
JMPEQ next, R1      compares R1 and R0  
...code for "then" branch...  
next: ...
```



next is a *label* at the next machine instruction

- The address is calculated by the compiler
- and inserted in the JMPEQ instructions

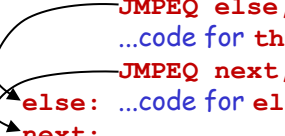
If statement: implementation in machine code

Example: A two-branch if statement

```
if ( a != b )  
{ ...then branch...  
}  
else  
{ ...else branch...  
}  
...
```

Could compile into Brookshear machine code:

```
MOV [a] -> R0  
MOV [b] -> R1  
JMPEQ else, R1      compares R1 and R0  
...code for then branch...  
JMPEQ next, R0      always jump  
else: ...code for else branch...  
next: ...
```



If statement: implementation in machine code

In general, the condition might be *any boolean expression*

- Not a comparison neatly matching a machine instruction

```
if (...cond...)
{ ...then branch...
}
else
...
```

We need a representation of boolean false and true:

- Let's choose 0 and 1
- (byte values 00 and 01)

So: Conditions evaluate to leave 0 or 1 in a register

Example on next slide...

If statement: implementation in machine code

Conditions evaluate to leave 0 or 1 in a register:

- For example: for `a == b` with result in `R1` could be evaluated in the Brookshear machine:

```
MOV [a] -> R0
MOV [b] -> R1
JMPEQ t, R1      jump if ==
MOV 00 -> R1     load false into R1
JMPEQ next, R0   always jump
t:  MOV 01 -> R1  load true into R1
next: ...
```

Finally we can look at if statements in general - next slide...

If statement: implementation in machine code

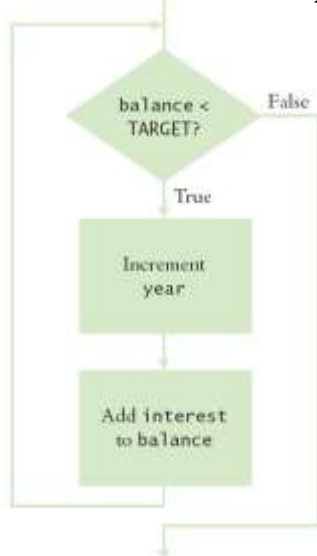
A general if statement:

```
if (...cond...)  
{ ...then branch...  
}  
else  
{ ...else branch...  
}  
...
```

Could compile into Brookshear machine code:

```
...evaluate cond into R1...    it's either 0 or 1  
MOV 00 -> R0                 load false into R0  
JMPEQ else, R1                compares R1 and R0  
...code for then branch...    always jump  
JMPEQ next, R0  
else: ...code for else branch...  
next: ...
```

While loops – now dealt with "easily"!



A loop executes instructions repeatedly while a condition is true.

```
while (balance < TARGET)  
{  
    year++;  
    double interest = balance * RATE/100;  
    balance = balance + interest;  
}
```

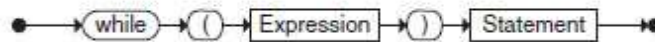
Syntax of while statements

- Reference syntax rule:

WhileStatement:
while (Expression) Statement

- Railroad diagram:

While Statement



- In both, remember that a Statement can be:
 - A simple statement
 - Or a Block

Implementing while statements

- A while loop like this:

```
while (...cond...)  
{ ...loop body...  
}  
...
```

- Could compile into Brookshear machine code like this:

```
rep: ...evaluate cond into R1...  
    MOV 00 -> R0      load false  
    JMPEQ past, R1     exit if false  
    ...compiled code for loop body...  
    JMPEQ rep, R0      always jump back  
past: ...
```

End of lecture