File scanning loops
&
loop invariants

A Simple Java Indenter

We talked about ...

Auto-indenting to make it easier to see missing
and extra braces

Let’s build our own!

Strategy?

Even before pseudocode ...
What general strategy can we use?

We want to ignore the original indentation, and
apply indentation according to program structure.
It’s not hard!

(from Zinn and the Art of Road Bike Maintenance)
Better tools for this job ...

String s;
s.charAt(i) the character at line i (from 0)
s.length() how many characters in s
s.trim() copy of s without leading spaces

plus the File manipulation we saw last time
and “noggin” (brain).

A strategy

Read line by line, printing modified line
Discard spaces at beginning of each line
Keep track of counts of ‘{‘ and ‘}’
Or just keep track of their difference;
call that “block level”
Insert enough spaces for current block level

Printing with indentation

*Code will look like this:

```java
while ( ... ){
    ++blockLevel
    body
    if ( ... ) {
        ++blockLevel
        ...
    } else {
        --blockLevel
    }
}
```

Controlling indentation
What’s the right order?
Adjust block level before or after printing line?

Pseudocode?
Assume we’ve already got a scanner ...
what’s the main loop look like?

For each line ...
Trim leading blanks
Print leading spaces
Print line
Adjust block level for next line

Better ...
For each line
  adjust block level according to first character
  print line indented for current block level
  adjust block level according to the rest

This will handle } else { correctly
Loop invariants

“Invariant”: Something that doesn’t change
Loop invariant:
  When I here in the loop,
  the invariant is true every time

Reasoning with invariants

initialize before loop;
// Invariant must be true here
while ( ... ) {
    // Invariant must also be true here
    ...
}

Invariants, basis, progress

... // Establish the invariant
while ( ... ) {
    if (basis case) {
        ...
    } else { // progress case
        ... // maintain the invariant
    }
}
// basis case

Invariants, basis, progress (2)

... // Establish the invariant
while ( not basis case ) {
    ... // progress case:
    ... // maintain the invariant while
    ... // reducing problem size
}
// basis case
Invariant for indentation

Block level = 0;  // Invariant established
For each line {
    adjust block level according to first character
    print line indented for current block level
    adjust block level according to the rest
    // Invariant maintained: Block level is the
    // difference between counts of ‘{‘ and ‘}’
}