```
.method static fahrenheitToCelsius (F)V
.limit stack 2
.var 0 is fahrenheit from begin to end_of_computation
.var 0 is celsius from end_of_computation to end
begin:
   fload_0              ; Push fahrenheit in variable 0
   ldc 32.0             ; Subtract 32
   fsub
   ldc 5.0              ; Multiply by 5
   fmul
   ldc 9.0              ; Divide by 9
   fdiv
end_of_computation:
   fstore_0             ; Now variable 0 is celsius
   getstatic java/lang/System/out Ljava/io/PrintStream;
   fload_0              ; Print variable 0
   invokevirtual java/io/PrintStream/println (F)V
   return
end:
.end method
```

It's also possible in Java for two different variables to have the same name in different parts of a method:

```
{
   int i;
   /* i is variable 1 */
}
{
   int j;
   int i;
   /* Here, j is variable 1 and i is variable 2 */
}
```

To let the debugger know which variable is named what, use this Oolong code:

```
.var 1 is i I from scope1begin to scope1end
.var 1 is j I from scope2begin to scope2end
.var 2 is i I from scope2begin to scope2end
scope1begin:
   ; Here variable 1 is i, and variable 2 is unnamed
```