# How Novice Programmers Interact with Programming Environments

2019 International Workshop on Computer Science Education (IWCSE 2019)

Brett A. Becker

University College Dublin & Beijing-Dublin International College @Beijing University of Technology

Slides, Abstract & References available at: www.brettbecker.com

brett.becker@ucd.ie

# Let's interact with a few programming environments

- Command-line / javac
  - often used with novices
- BlueJ 3 & 4
  - designed for novices
- EJE
  - basic, suitable for novices
- Netbeans
  - "Industry-strength"

# A very simple Java program

```
public class PrintNums {
    public static void main (String[] args){
        for(int x = 0; x < 10; x++)
            System.out.println(x);
    }
}
```

Output:
0
1
2
3
4
5
6
7
8
9

# Let's introduce a syntax error

```
public class PrintNums {
        public static void main (String[] args){
                for(int x = 0: x < 10; x++)
                        System.out.println(x);
        }
}
```

Regular colon :
instead of semicolon ;

# Command-line / javac

javac 1.8.0_131

# Command-line / javac (with error)



- Produces six "errors"
  - I'd rather call these *messages*
  - One first* message
  - Five subsequent* messages

\* Taxonomy of 'traditional' compiler error messages:
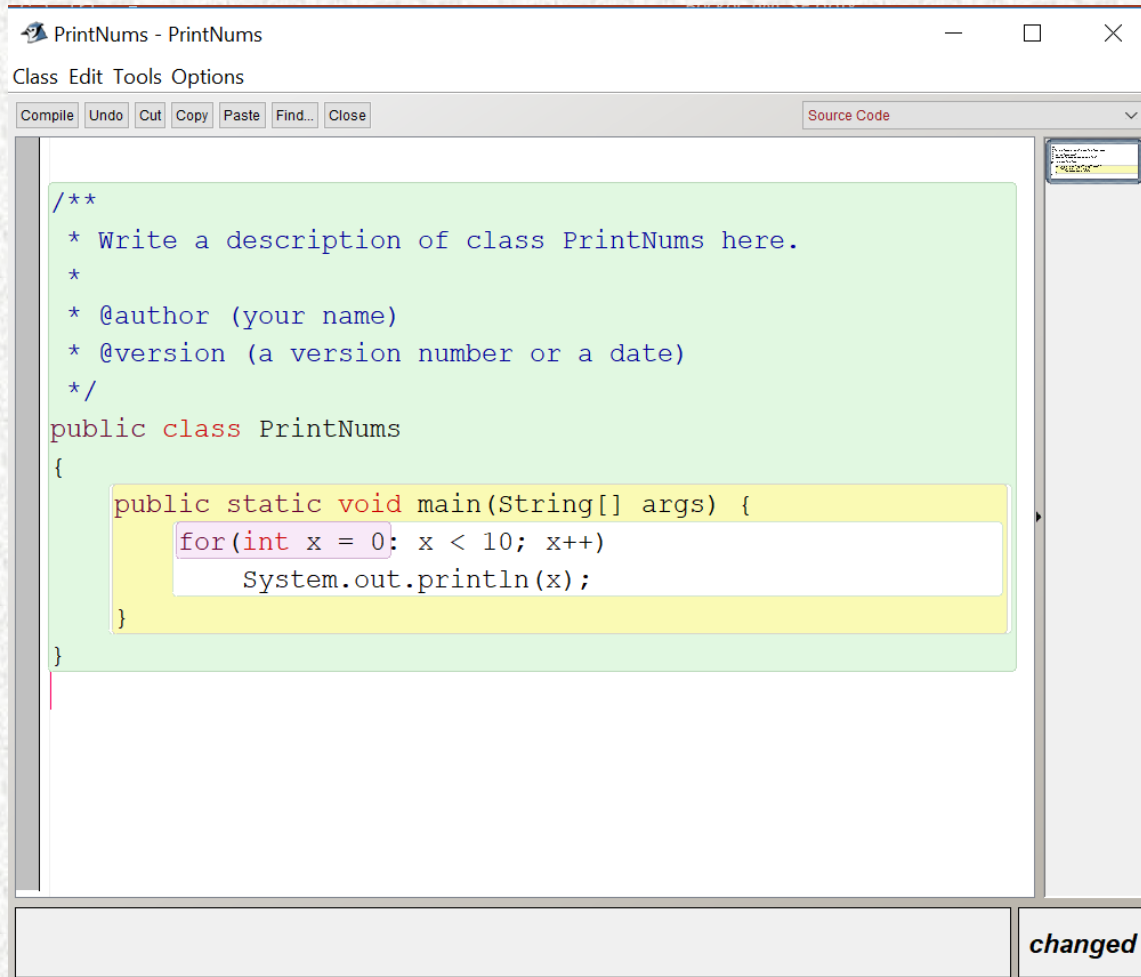Becker, et al. 2018: brettbecker.com/publications/#27

# Command-line / javac (error fixed)



- Produces nothing on success

# BlueJ 3

BlueJ 3.1.7

Kölling et al., 2003: www.bluej.org/papers/2003-12-CSEd-bluej.pdf

# BlueJ 3 (with error)



- Some red highlighting
- Red cursor at end

# BlueJ 3 (with error)
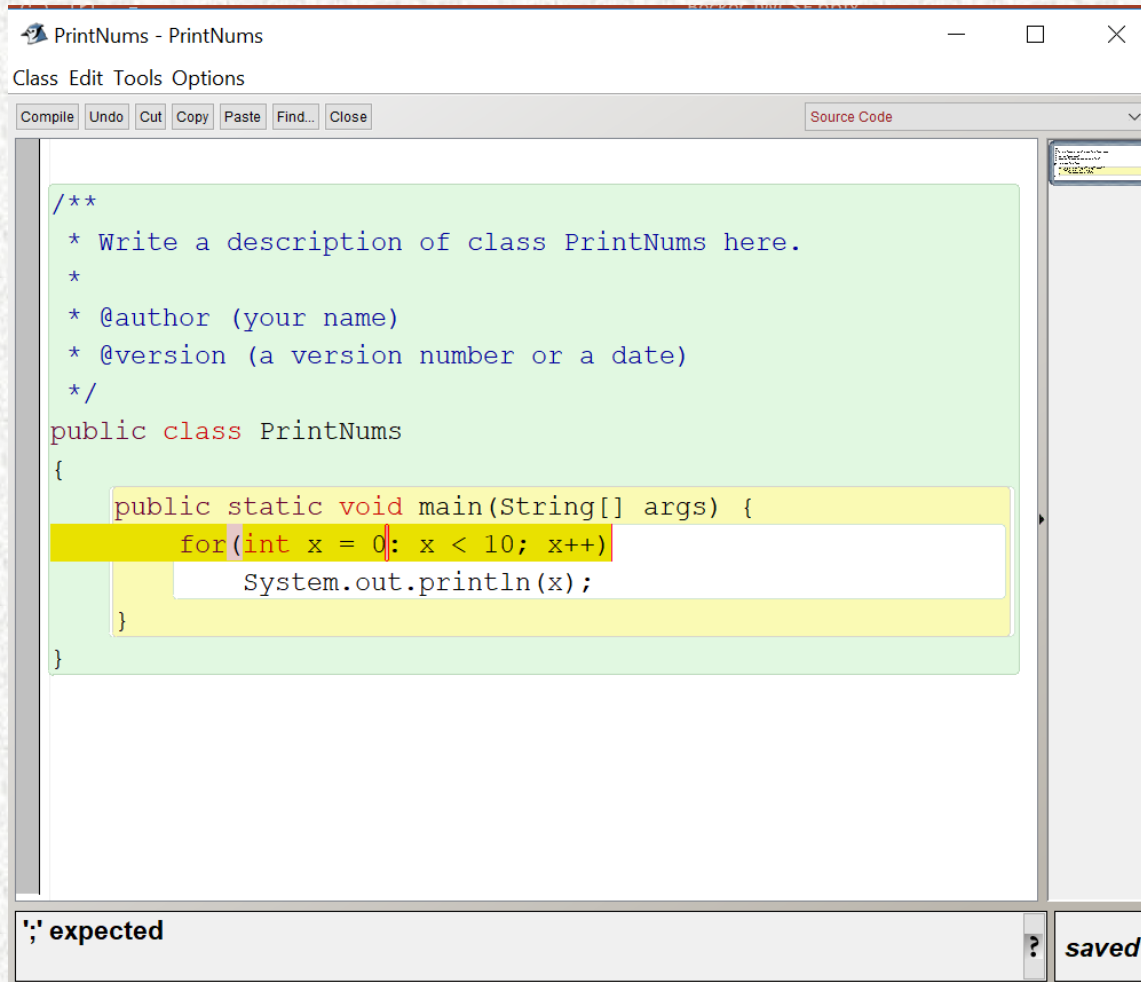


- Upon clicking compile:
  - Message appears in output window
    - This is a simplification of the first javac message
  - Highlighting changes
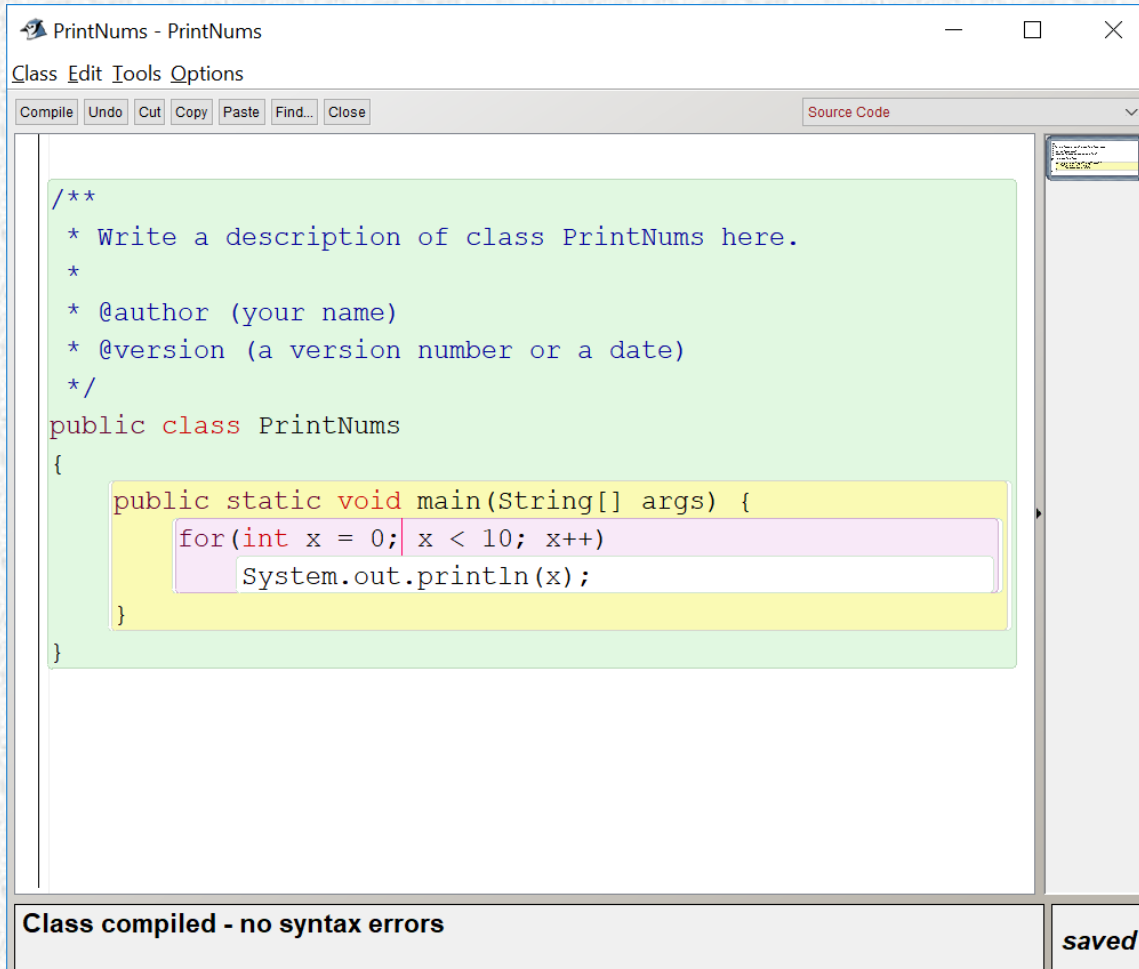  - Red cursor changes, moves

# BlueJ 3 (with error)
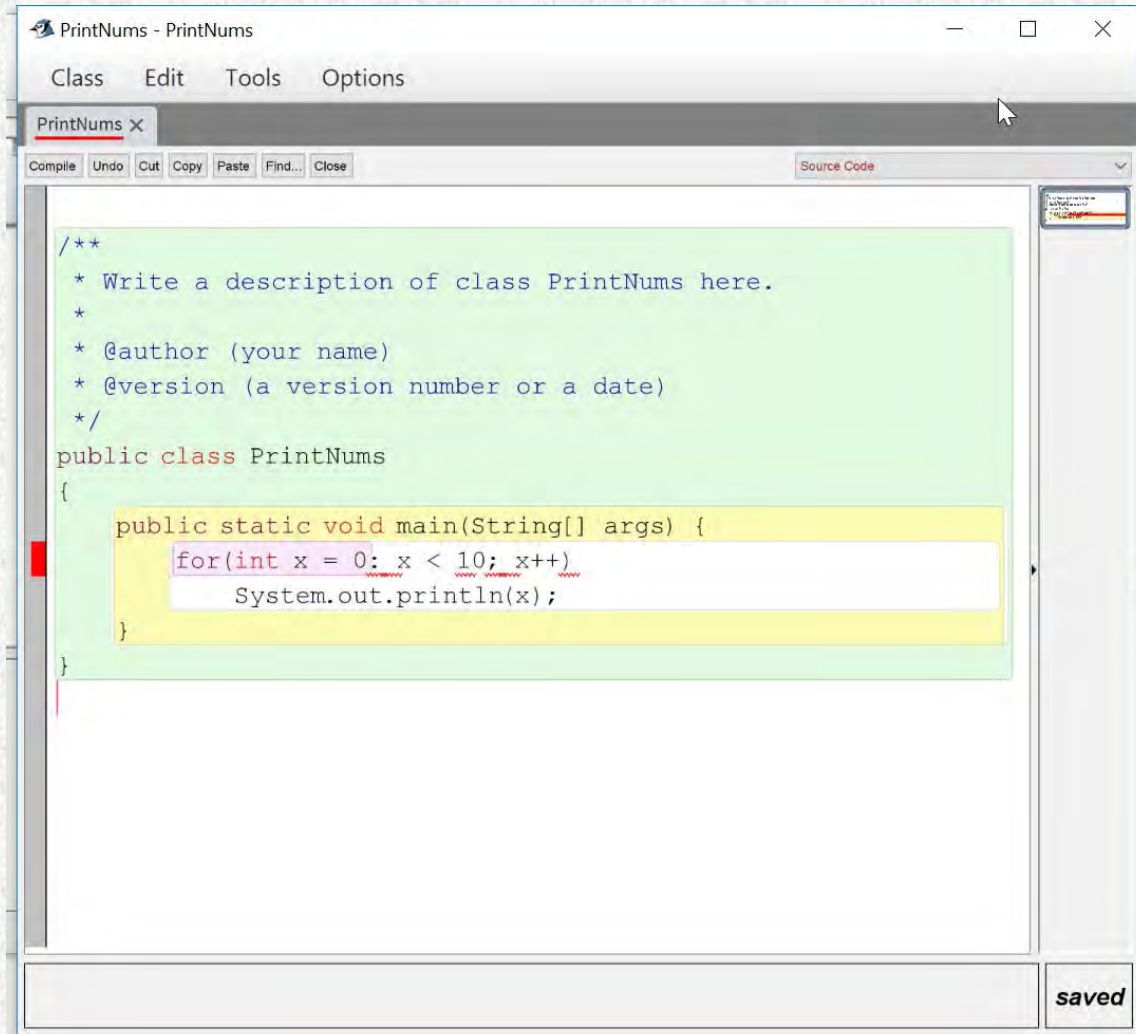


- Clicking compile again does nothing
  - Repeat forever...

# BlueJ 3 (error fixed)



- Upon fixing the error and clicking compile, we get:
  - Success message
  - Highlighting changes
  - Red cursor changes and moves

# BlueJ 4

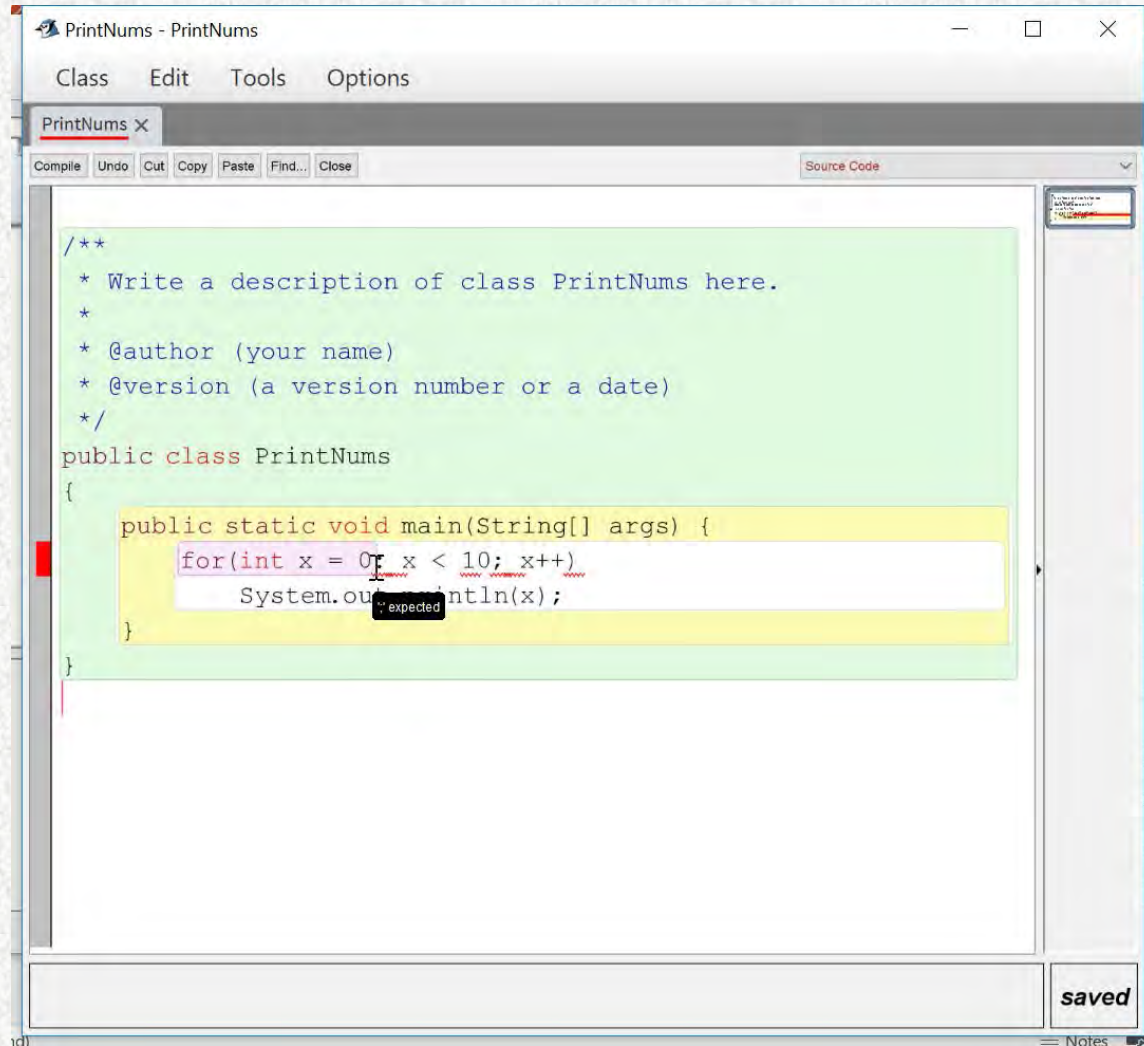BlueJ 4.0.0 preview 2

# BlueJ 4 (with error)



- Without clicking compile, BlueJ 4 has:
  - Similar highlighting to BlueJ 3
  - A red bar in the left margin
  - Lots of underlining (red squiggles)
  - Similar to BlueJ 3, mysterious red cursor at end

# BlueJ 4 (with error)
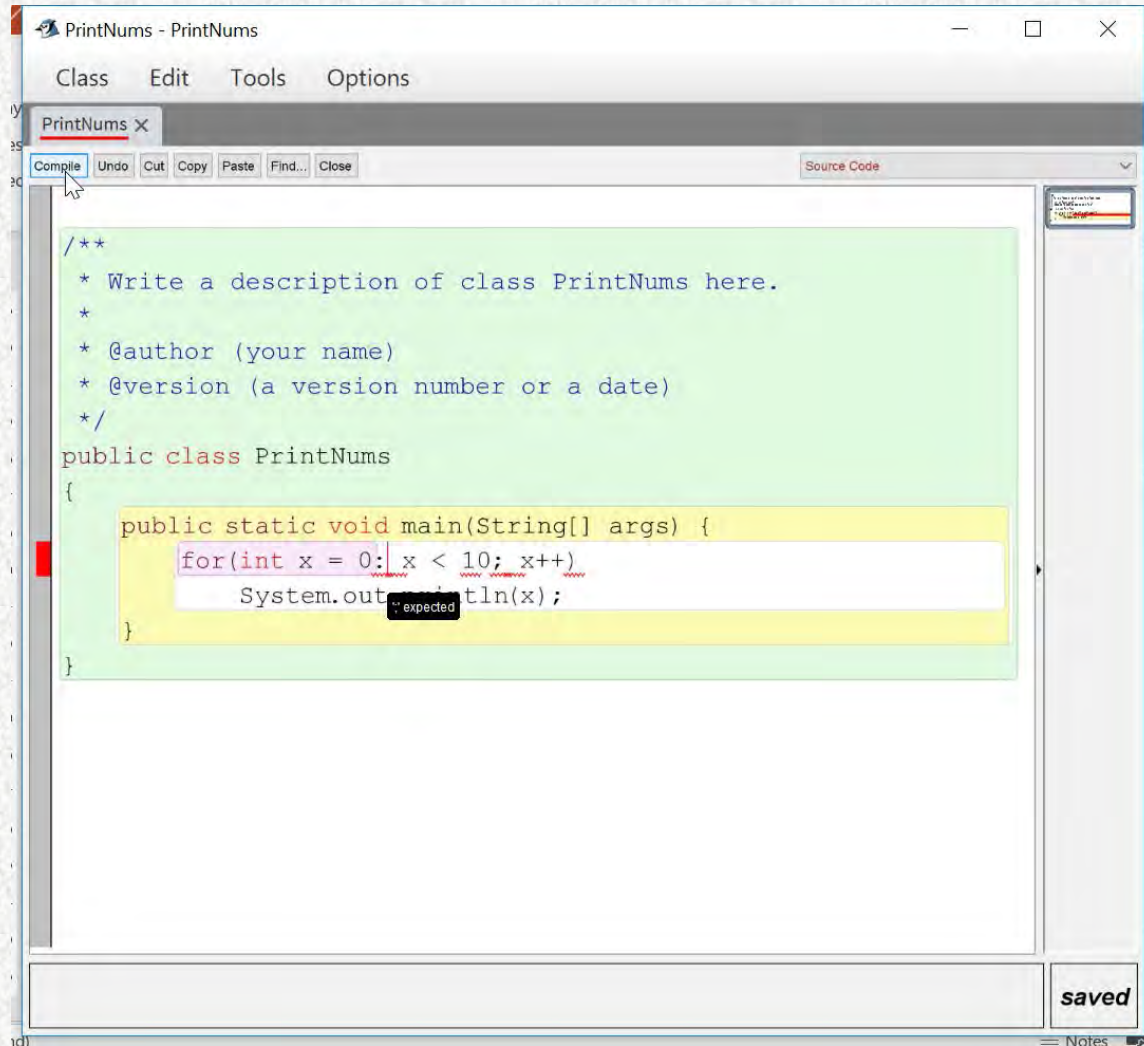


- Clicking compile generates no messages in the output window
- Generates pop-up with abbreviated message that matches first javac message
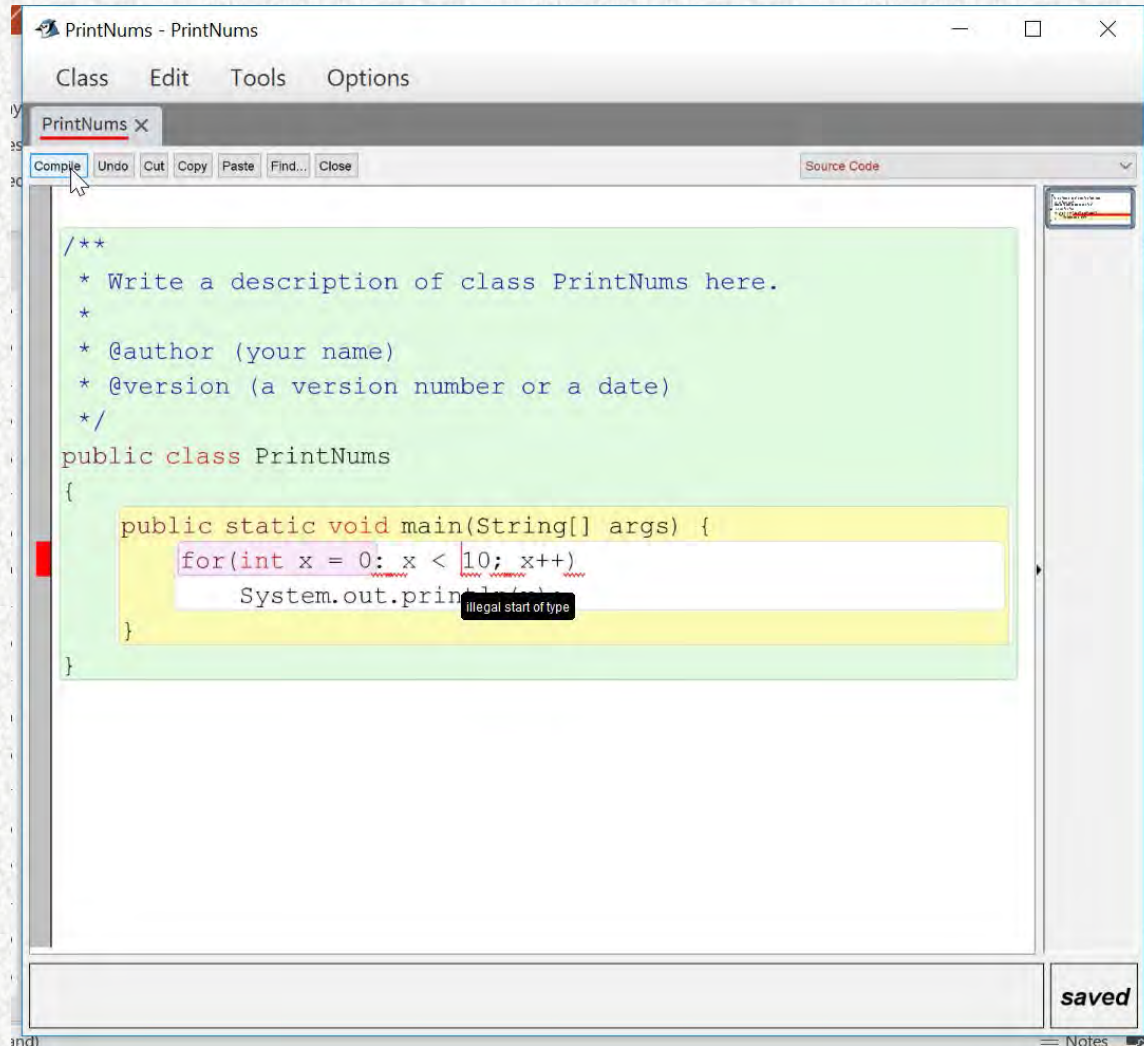
# BlueJ 4 (with error)



- Clicking compile again generates no messages in the output window
- Generates new pop-up with abbreviated message that matches second javac message

# BlueJ 4 (with error)
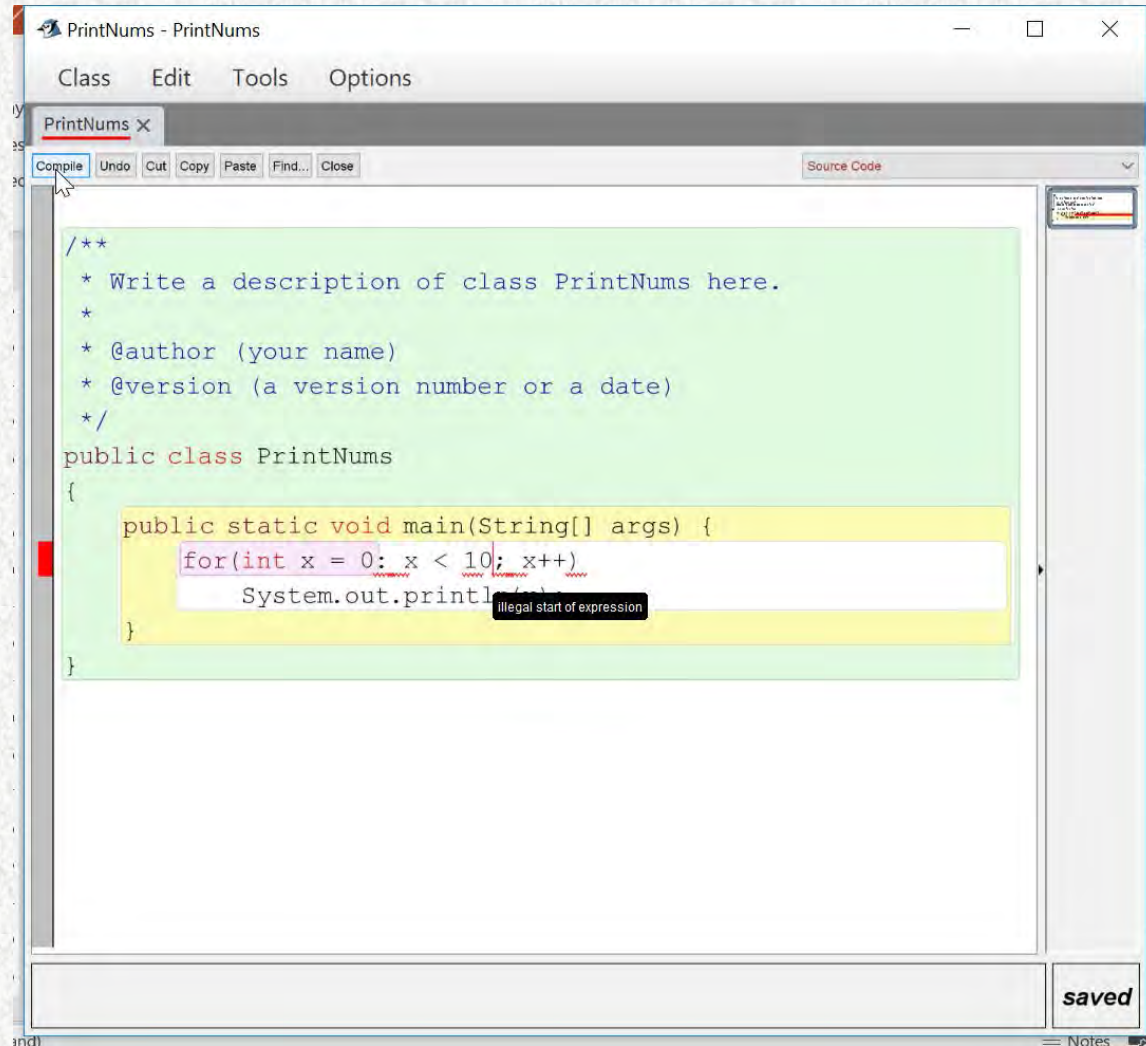


- Clicking compile again generates no messages in the output window
- Generates new pop-up with abbreviated message that matches third javac message

# BlueJ 4 (with error)



- Clicking compile again generates no messages in the output window
- Generates new pop-up with abbreviated message that matches fourth javac message

# BlueJ 4 (with error)



- Clicking compile again generates no messages in the output window
- Generates new pop-up with abbreviated message that matches fifth javac message
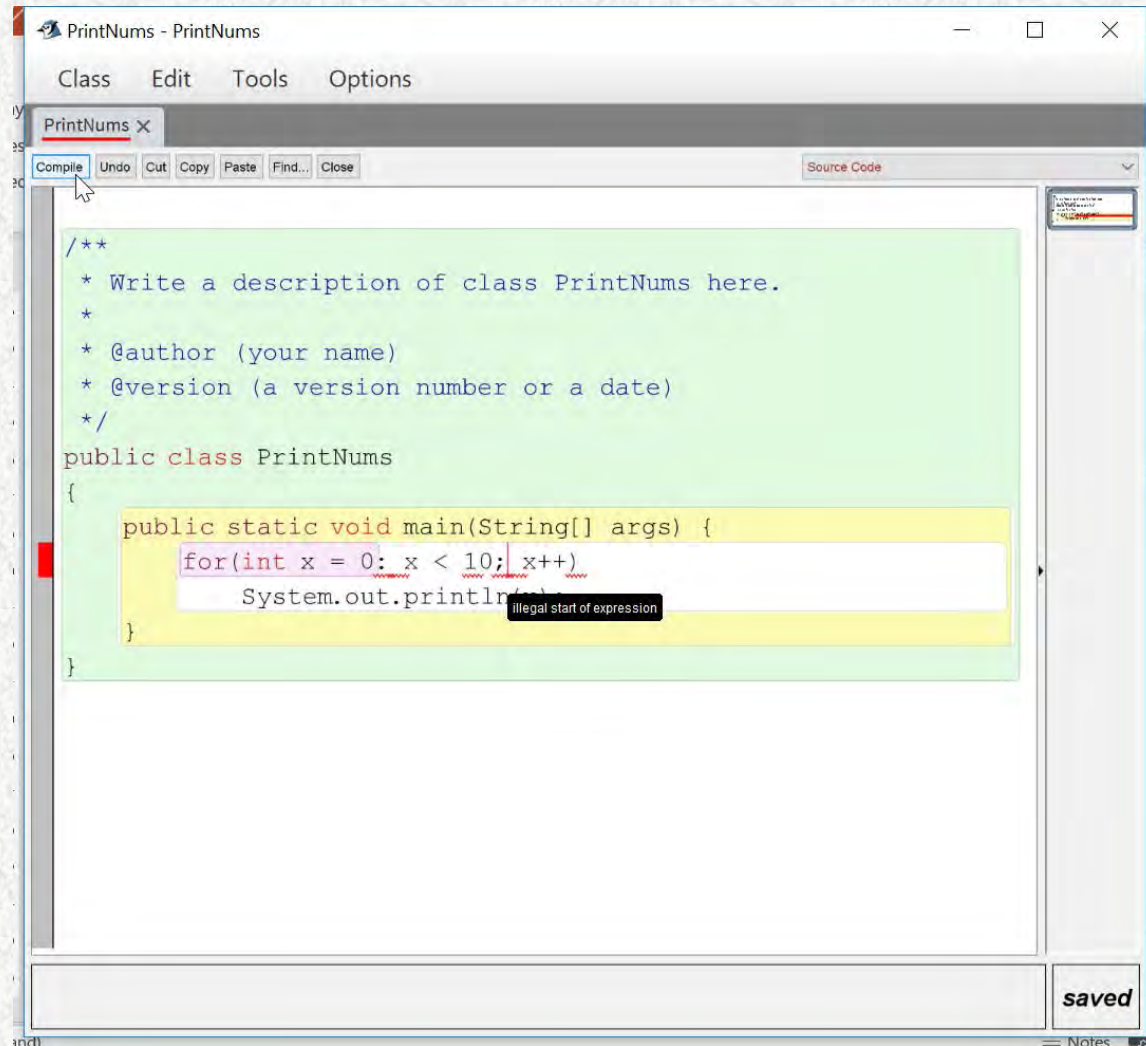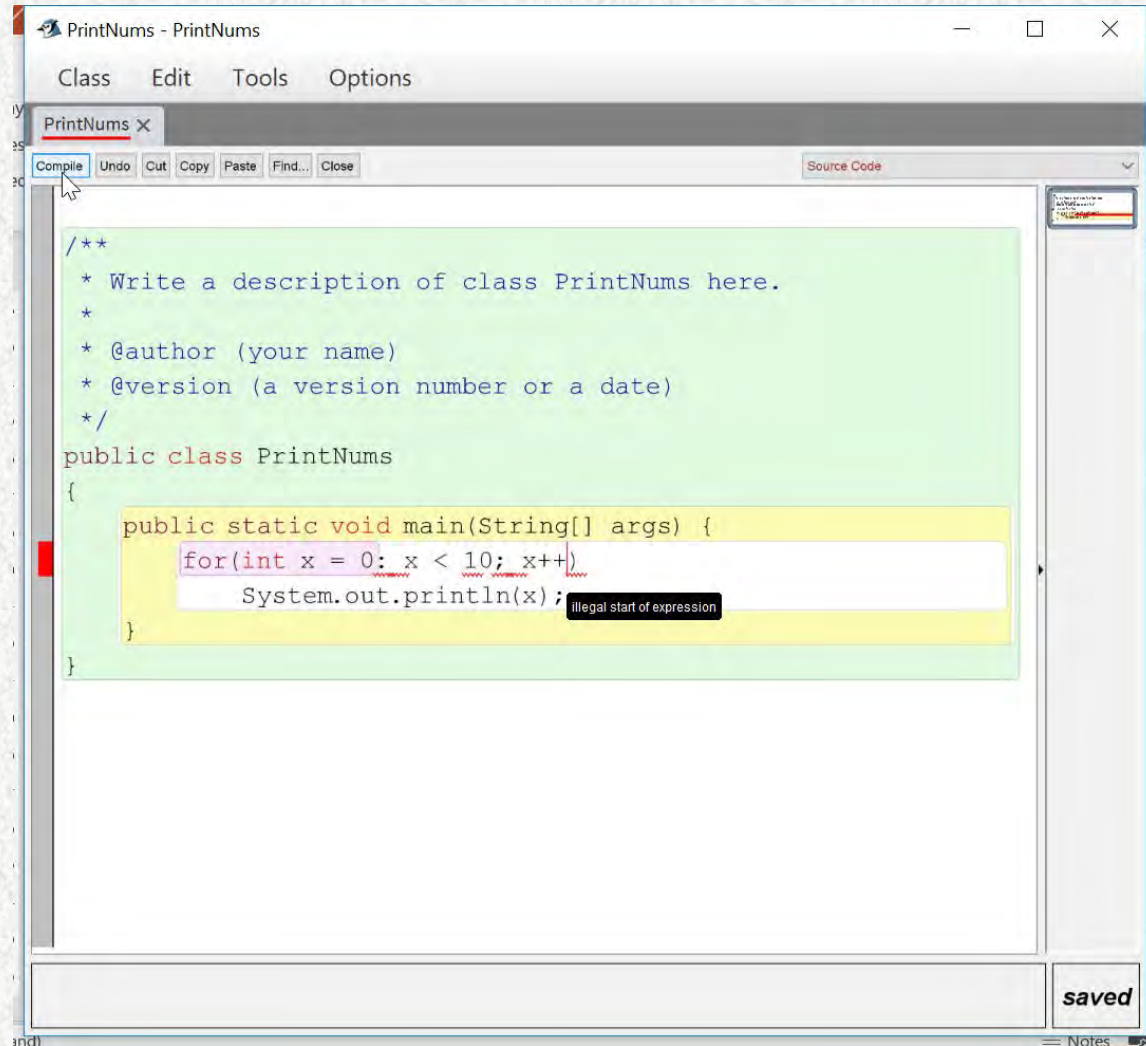
# BlueJ 4 (with error)



- Clicking compile again generates no messages in the output window
- Generates new pop-up with abbreviated message that matches sixth javac message
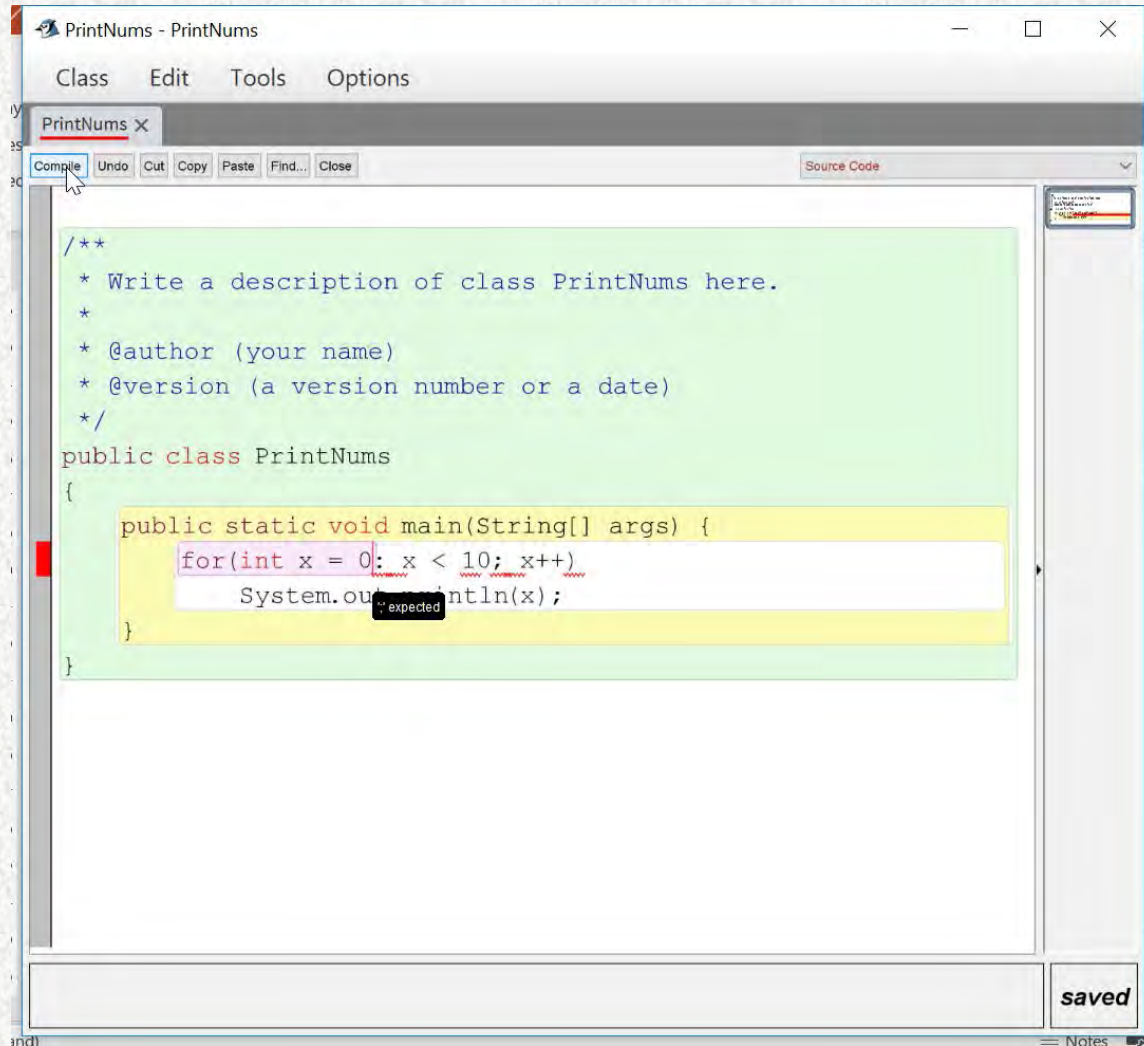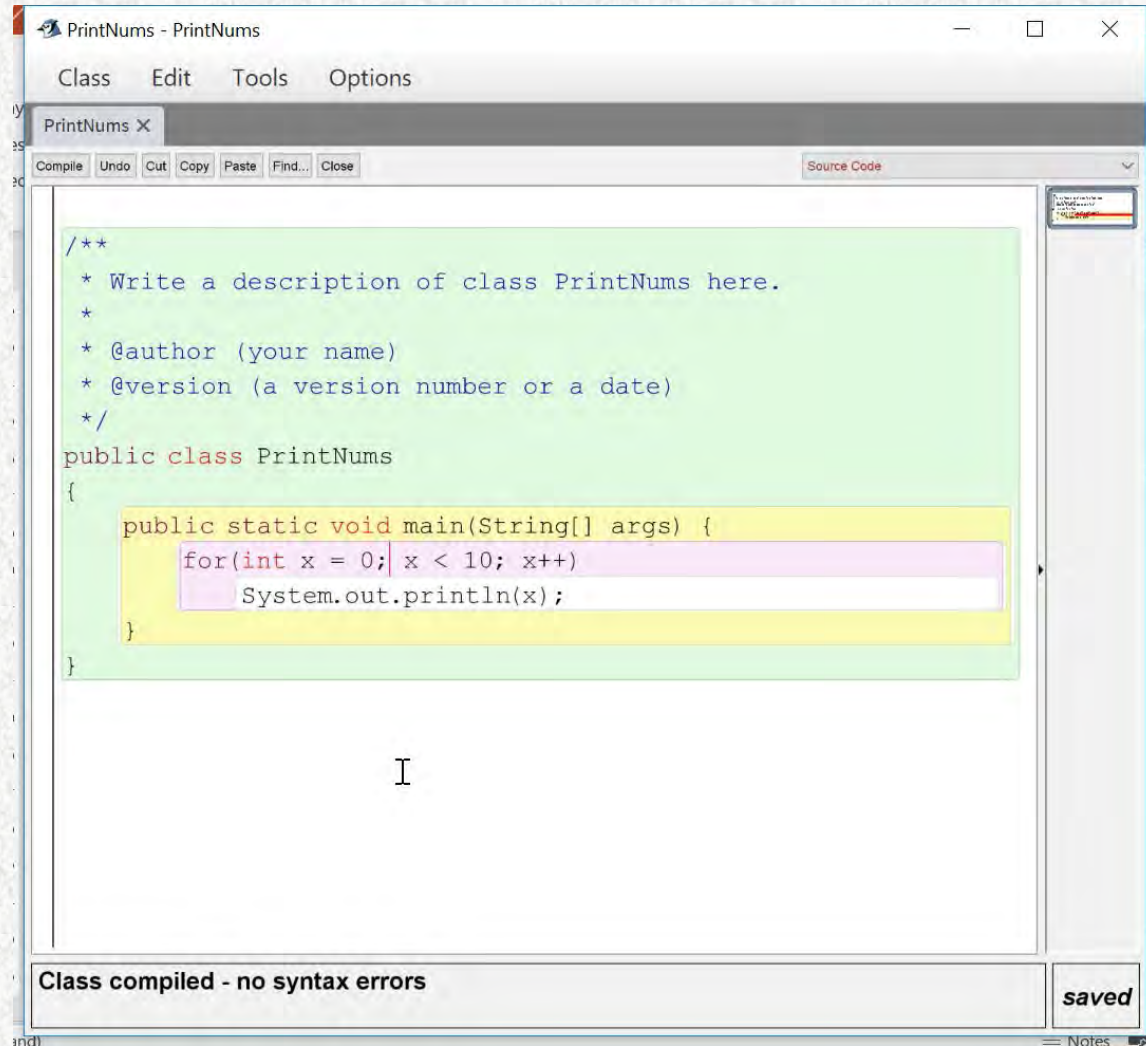
# BlueJ 4 (with error)



- Clicking compile again generates no messages in output window

- Generates new pop-up with abbreviated message that matches **first** javac message (**again**)

- Errors are being cycled in round-robin fashion

# BlueJ 4 (error fixed)



- Upon fixing the error, we receive:
  - Success message in output window
  - Highlighting changes
  - Red bar in margin disappears
  - Pop-up disappears
  - Red cursor moves but remains

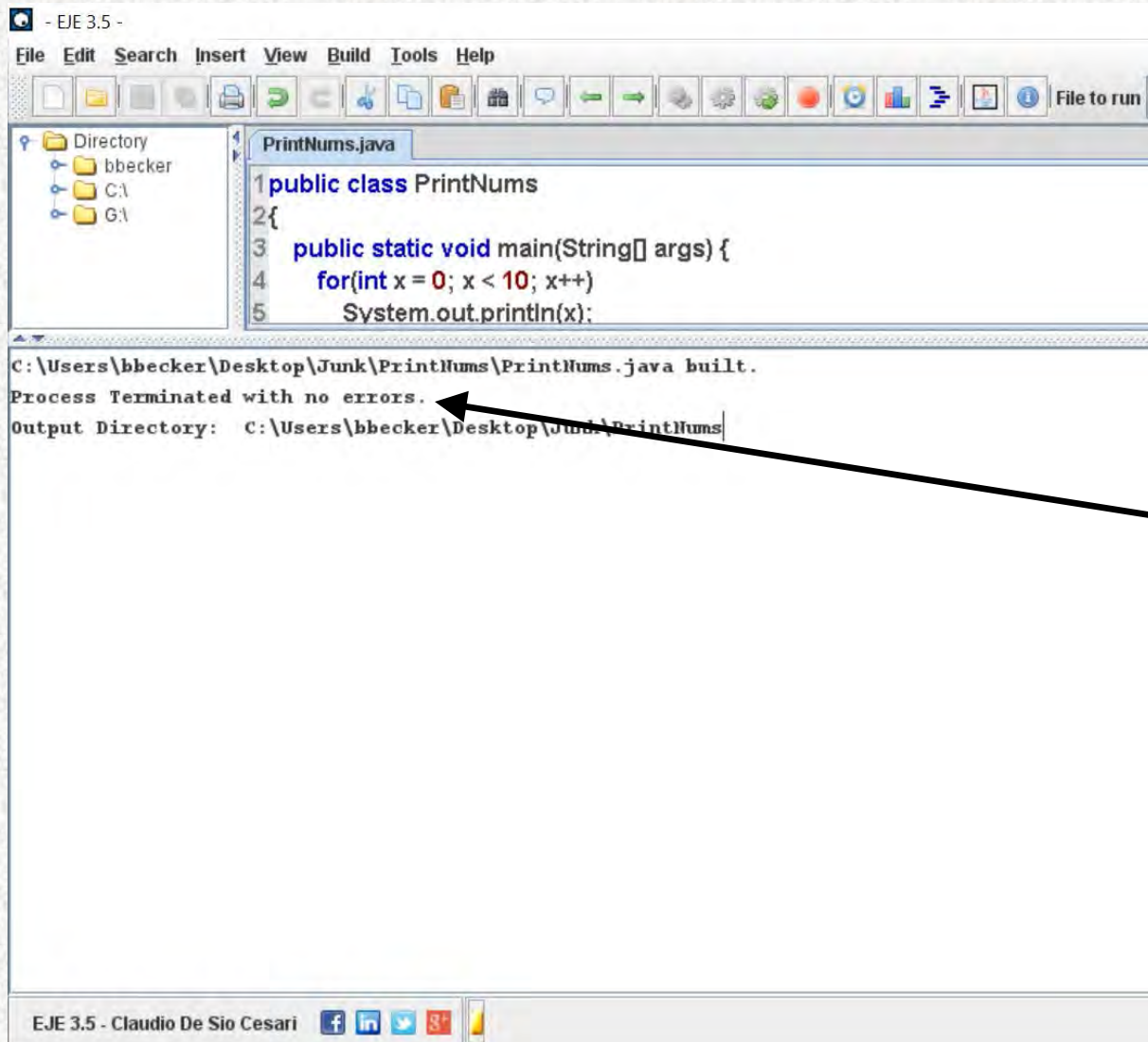# EJE (Everyone's Java Editor)

EJE 3.5

http://sourceforge.net/projects/eje/

# EJE (with error)



- Upon clicking compile, EJE output is almost identical to javac
- No change in edit window

# EJE (error fixed)



- Upon fixing error and clicking compile we get a message in the output window
  - Unlike javac
    - Same javac version though
- Somewhat oddly it uses "Terminated" in a success message

This might not be what a novice pictures when seeing this message

Process Terminated with
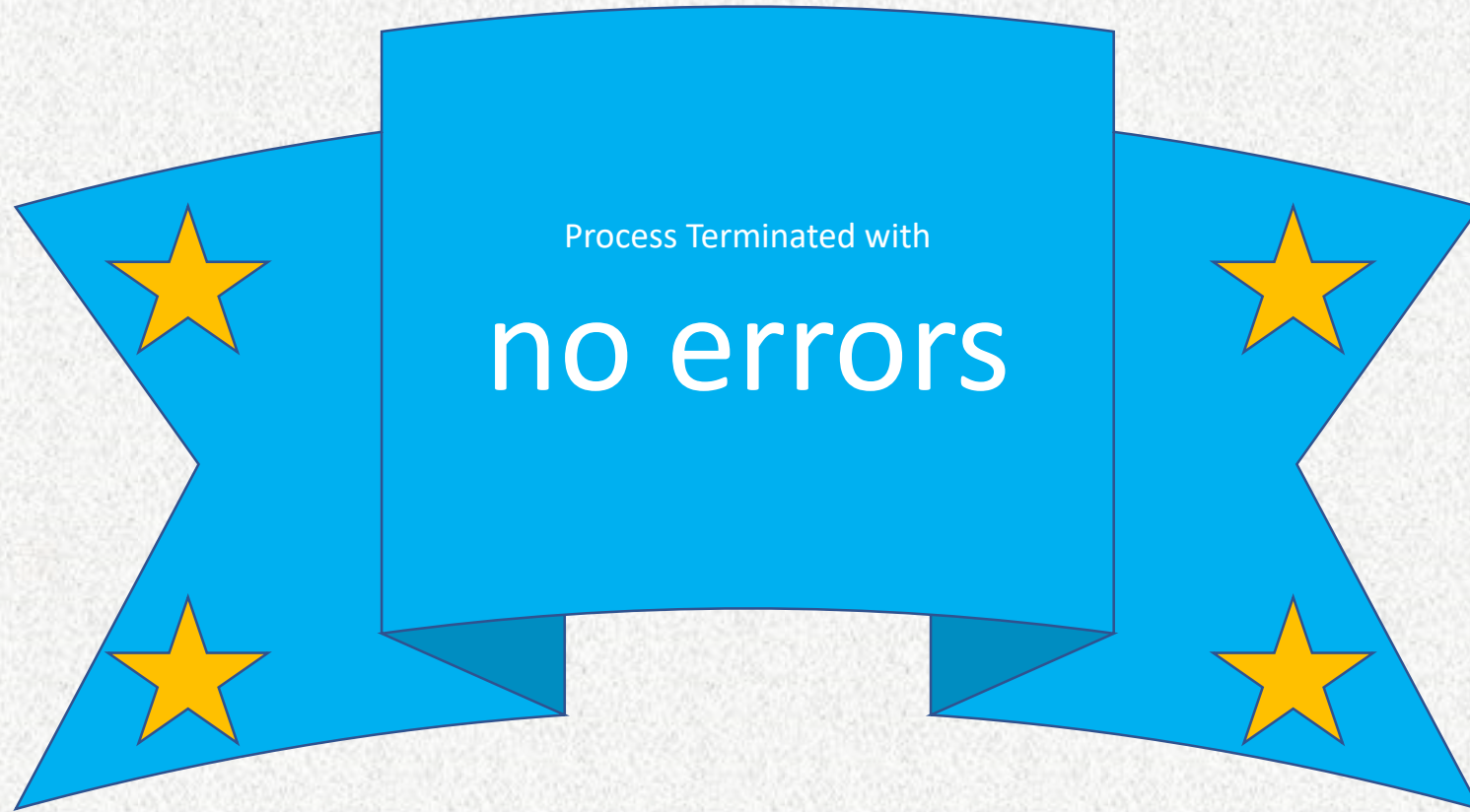
no errors

This might be what a novice pictures when seeing this message

X                                                    X

Process **Terminated**

with no errors

X                                                    X

# Netbeans

Netbeans 8.1

# Netbeans (with error)



Without clicking compile we see:

- One big red underline
- Red lines and box in right margin
- Lightbulbs in left margin

# Netbeans (with error)
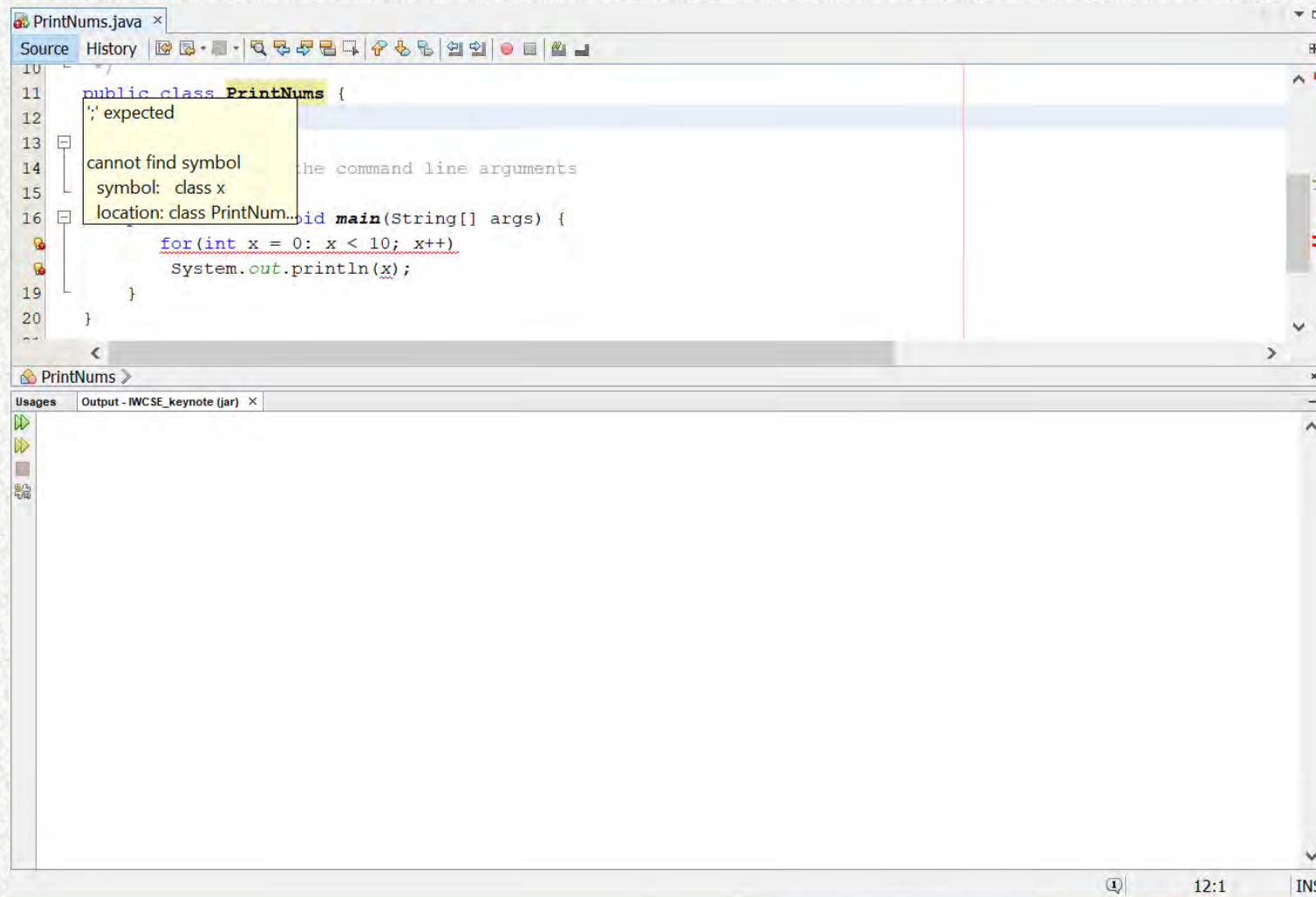


**Hovering** over a lightbulb gives us:

- pop-up with what seems to be one short message, and another truncated message, but there is no way to see the missing text

- When you move the cursor, the window disappears

# Netbeans (with error)
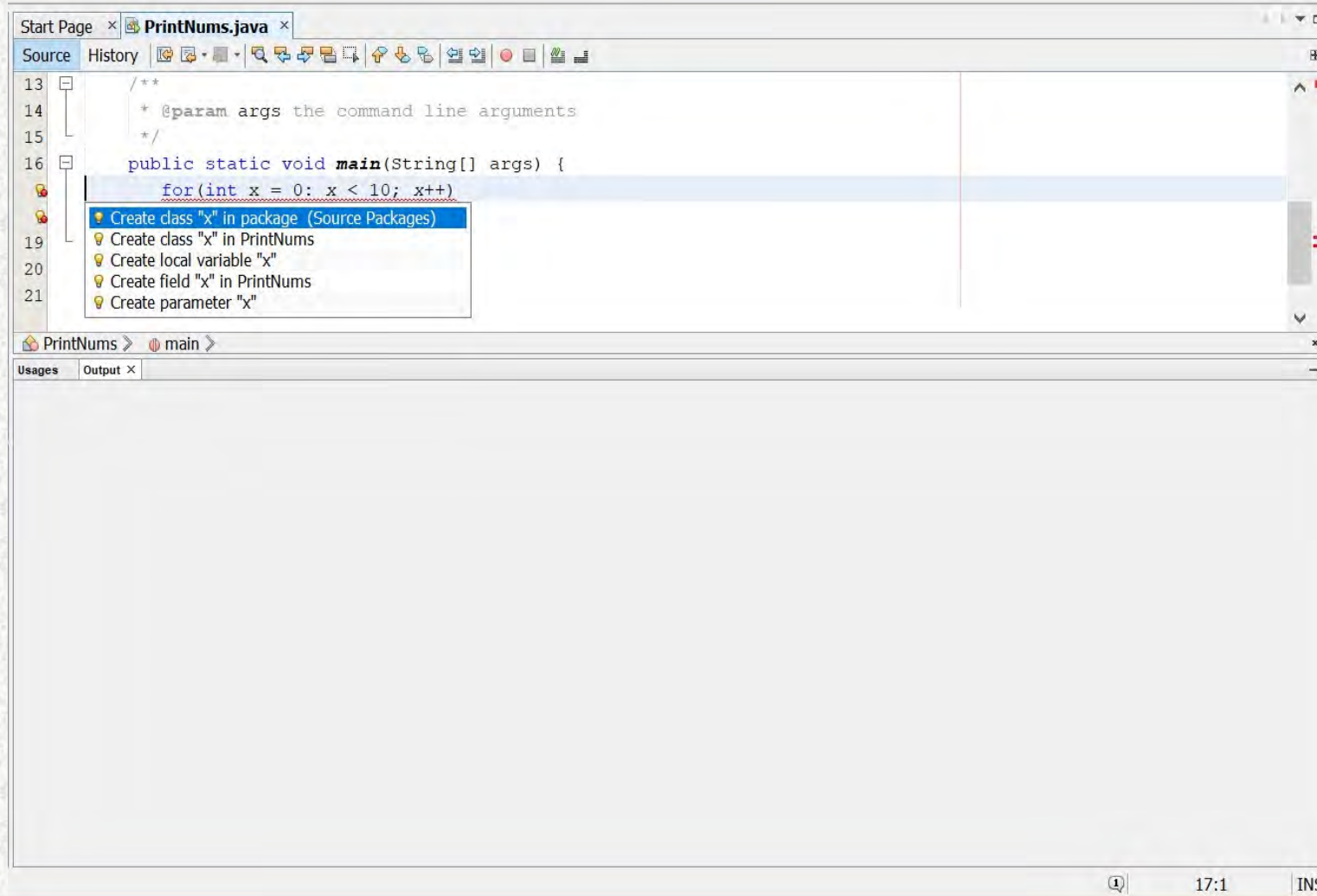


**Clicking** the first lightbulb we get:

- A list of what seem to be suggestions
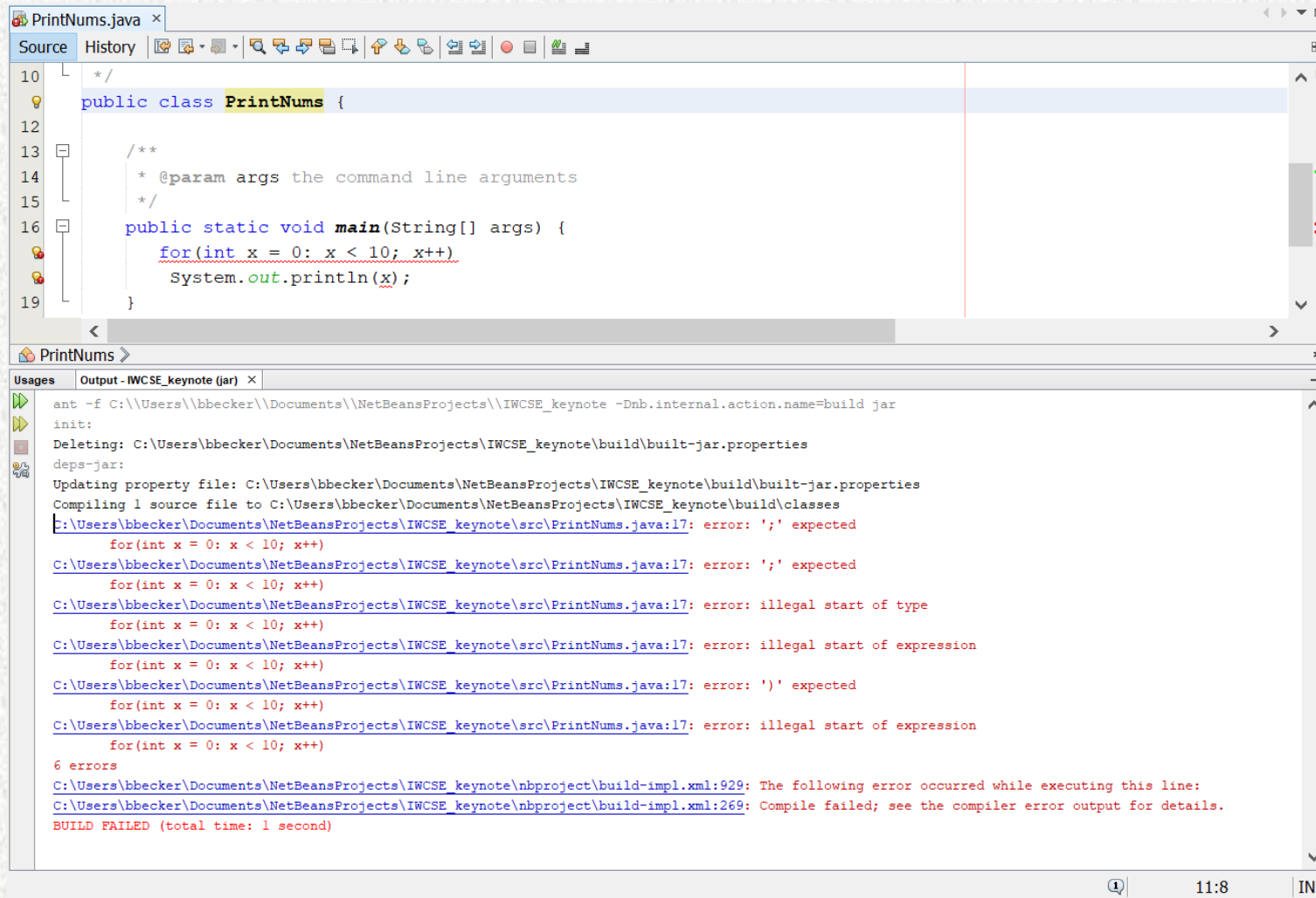- I clicked one of these and what happened was too complicated to present
- **It *was not* a fix**

# Netbeans (with error)



Clicking compile gives us this in the output window:

- Similar to javac but more complicated

- Red text

- Black text

- Grey text

- What appear to be links

# Netbeans (with error)



Clicking what looks to be a link highlights the "link" in blue and moves the cursor in the edit window

So, not really a link…

# Netbeans (with error)

What about the links at the bottom?

# Netbeans (with error)



Oh.

That did not help.

I guess I'll close that window…

# Netbeans (with error)
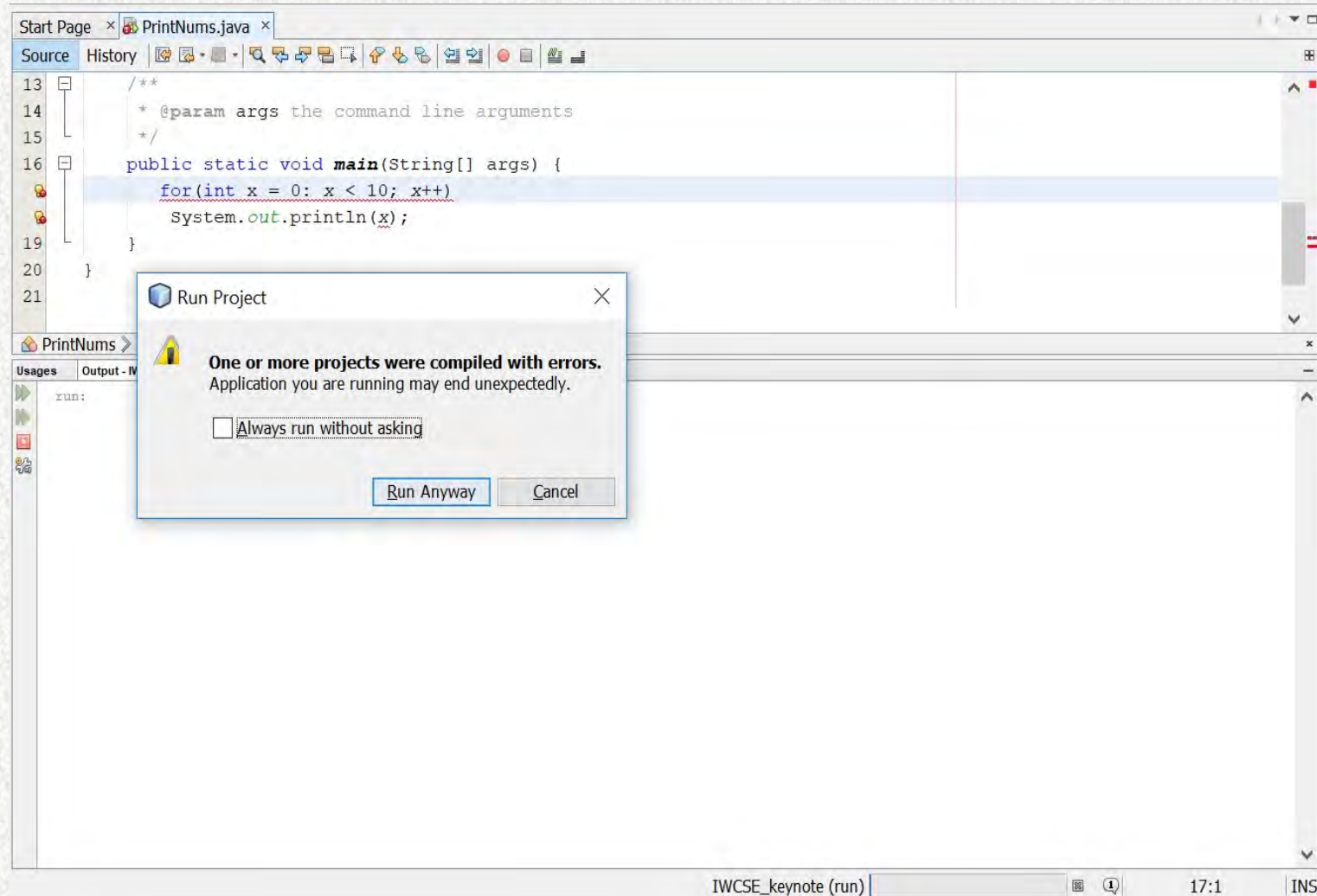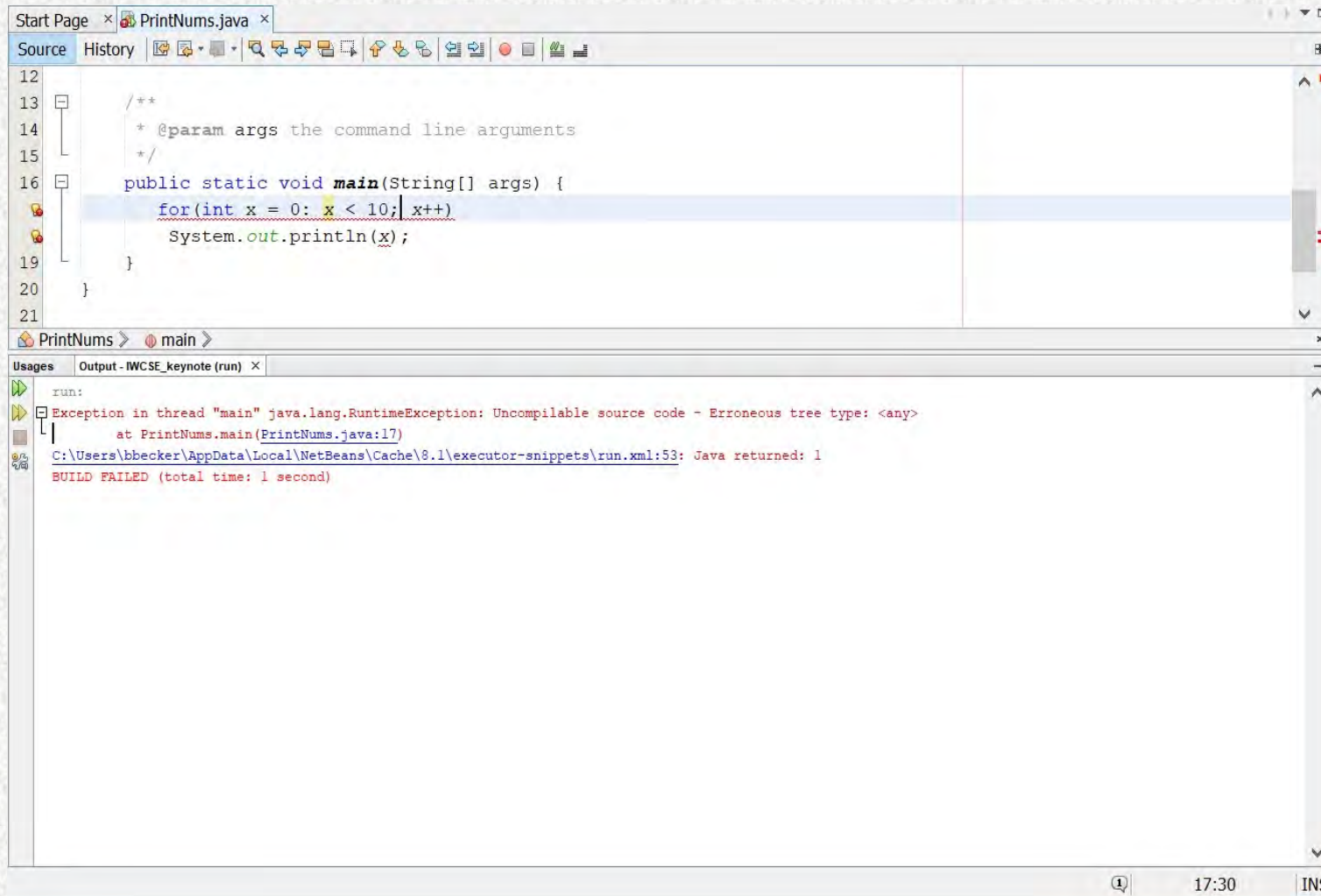


Clicking <u>run</u> gives us a warning window

Maybe then, the error doesn't matter?

Maybe my code will work anyway…?

# Netbeans (with error)
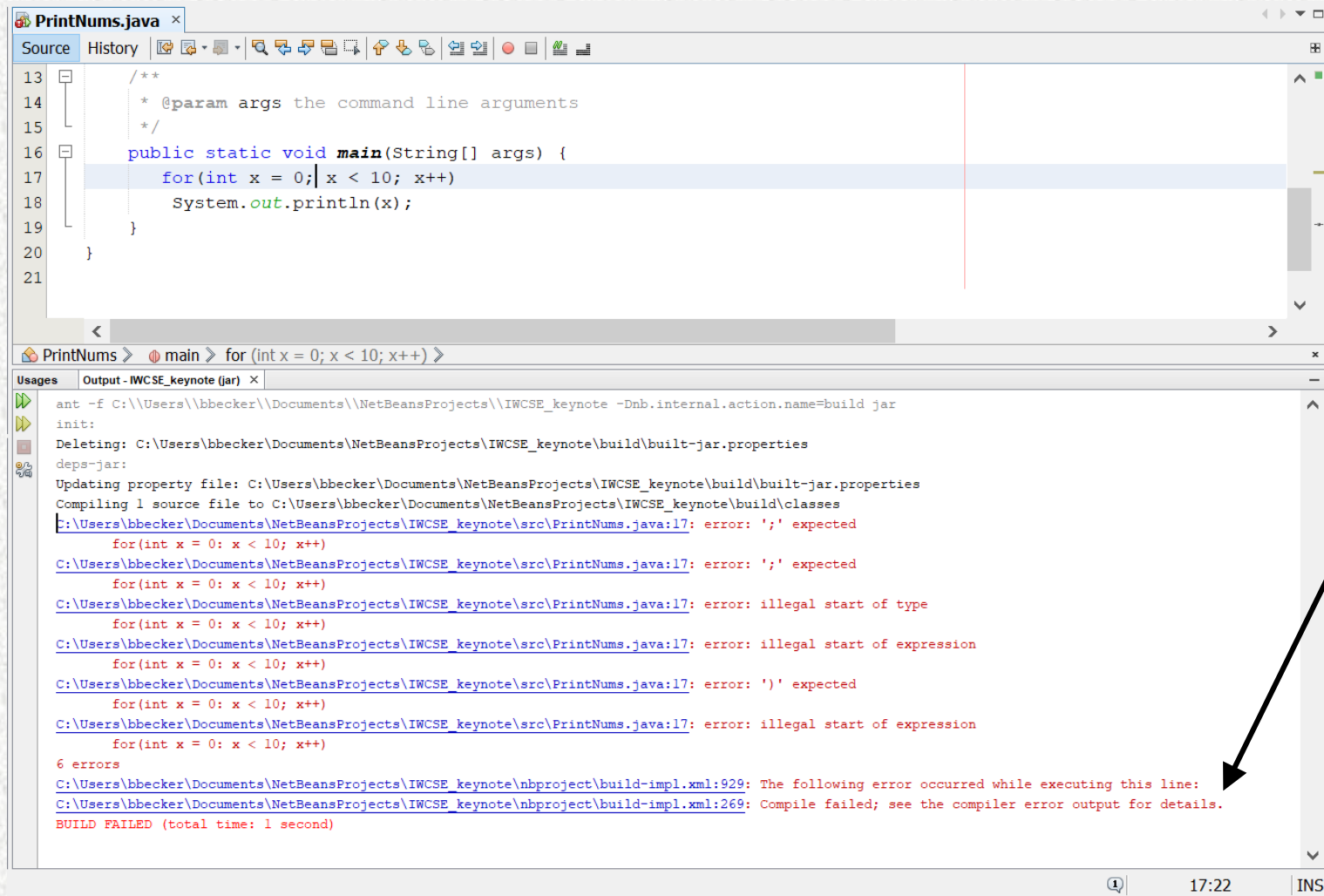


I guess not

Clicking *Run Anyway* gives us this

Clicking compile again brings us back to where we were

# Netbeans (error fixed)



When error is fixed:
- Lightbulbs disappear
- Underlining disappears
- Red margin lines disappear

- Messages in output window persist
- This says "**compile failed; see the compiler error output for details**"
- Uh. What is this then? I thought **this was** the compiler error output
- And, I thought my code was fixed…

# Netbeans (error fixed)



Clicking compile finally brings me to something that looks good!

# Recap

- One short program, one syntax error, 5 environments
- We observed
  - Up to six error messages
    - but, sometimes one, and sometimes none (even when there were errors)
  - Artefacts directly related to this error
  - Behaviours directly related to this error
  - Other artifacts & behaviours, some less related to the error / more general

# Artefacts related to the error (I think)

**We saw:**

- Lone messages
- Long lists of messages
- Nothing
- Text-based 'output' messages (shown in separate window in IDEs)
  - "First" messages ("Shown", in BlueJ/Blackbox terminology)
  - Several "subsequent" messages
- Underlining (red-squiggles)
- Mysterious red lines in the middle of lines of code
- Whole lines of code highlighted in red
- Partial lines of code highlighted in red
- Whole lines of code highlighted in yellow
- Whole lines of text highlighted in blue

- **|** Vertical red markers in left margin
- Horizontal and square markers in right margin
- Lightbulbs
- Windows that appear on hover and disappear without allowing us to see what seems to be missing text
- Windows with suggestions that would not fix the problem
- A red cursor|
- Red text
- Black text
- Green text
- Grey text
- Blue text links
- Blue text that looks like a link but isn't

# Behaviours related to the error

- Presentation of long lists of errors
  - Some of which are not *really* errors (at least not from the student's view)
- Message filtering - presentation of the first text-based message only
- At least three styles of pop-up windows with text
  - 'Static' windows that come and go with clicks
    - Some of that text seemingly wrong (and not in the javac output)
  - 'Semi-Transient' windows with text that came and went when 'compile' was clicked
    - These rotated in round-robin fashion
  - Transient windows that disappear when you try to interact with them
- Hyperlinks that pointed out errors in the code
  - Many that weren't real errors
- Hyperlinks that took me to XML code
- Suggestions that broke my code more than it was already
- The ability to execute code that wouldn't compile
- More…

# Other artefacts & behaviours

- Syntax highlighting
- Bracket matching (which actually had me confused at one point)
- 'Background' compilation and 'click-to-compile'
- The word "error" on multiple occasions which in many cases wasn't a real error
- The words "FAILED", "illegal" and "Terminated"*
- All kinds of skeleton code that a novice might not understand
  - In both the professional and pedagogical IDEs
- Mixing of automatic behaviour and behaviour requiring user input
- More…

*which, apparently, was part of a GOOD message

# Four ways of indicating success with text

1. BUILD SUCCESSFUL (total time: 1 second)
2. Class compiled – no syntax errors
3. Process Terminated with no errors
4.

Number 4 is literally, nothing. That is what javac (at the command line) told me when the compilation was successful:

```
C:\Users\bbecker\Desktop\Junk>javac PrintNums.java

C:\Users\bbecker\Desktop\Junk>_
```

# Would you be confused if you were a beginner?

?

# What are the effects of these artefacts and behaviours?

- How are students learning to program, possibly for the first time, supposed to navigate this, on top of learning:
  - the language
  - the environment (hopefully with the goal of adapting to other environments)
  - how to adhere to good practice / convention (indenting, variable naming, commenting, etc.)
  - problem solving
  - etc.
- …and, maintaining momentum, desire, and the attitude to persevere

# What I did not discuss and could be even more fundamental

**1. The readability of error messages is not well studied**



```
PrintNums.java:3: error: ';' expected
                for(int x = 0: x < 10; x++)
                             ^
PrintNums.java:3: error: ';' expected
                for(int x = 0: x < 10; x++)
                              ^
PrintNums.java:3: error: illegal start of type
                for(int x = 0: x < 10; x++)
                           ^
PrintNums.java:3: error: illegal start of expression
                for(int x = 0: x < 10; x++)
                              ^
PrintNums.java:3: error: ')' expected
                for(int x = 0: x < 10; x++)
                               ^
PrintNums.java:3: error: illegal start of expression
                for(int x = 0: x < 10; x++)
                                   ^
```

Barik, 2018: https://repository.lib.ncsu.edu/bitstream/handle/1840.20/35439/etd.pdf?sequence=1
Becker et al. 2019 ITiCSE working group: https://cszero.wordpress.com/2019/02/11/1826/

# What I did not discuss and could be even more fundamental

2. **What happens (psychologically and metacognitively) with these negative words?** Especially when coupled with instantaneous feedback from a machine that a student might think can never be wrong, because they were probably told that, or given that impression...

```
Process Terminated
with no errors
```

BUILD
FAILED

De Lira, 2017: https://dl.acm.org/citation.cfm?id=3105726.3105742
Prather et al.: https://dl.acm.org/citation.cfm?id=3106169

# What I did not discuss and could be even more fundamental

**3. How are non-native English speakers, and non-English speakers impacted by this?**

```
Illegal start of expression
```

- We already know that problem presentation can affect students, even when they are native speakers of English, and in particular if they are not

- Why would messages from an interface be any different?

Bouvier et al., 2016:https://dl.acm.org/citation.cfm?id=3024912

# What I did not discuss and could be even more fundamental

**4. How do students that learned to program with block-based languages transition to these environments?**



Altadmri et al., 2016: http://www.twistedsquare.com/Syntax-Cost.pdf
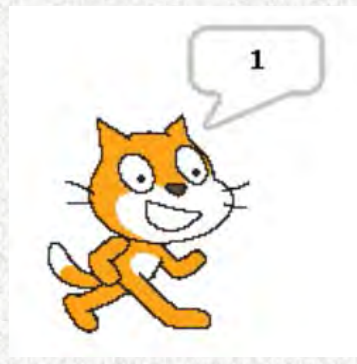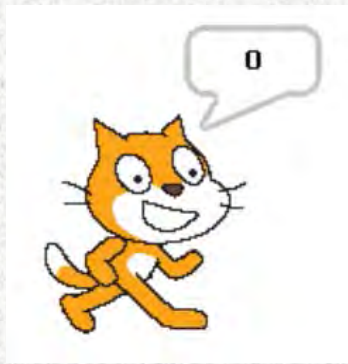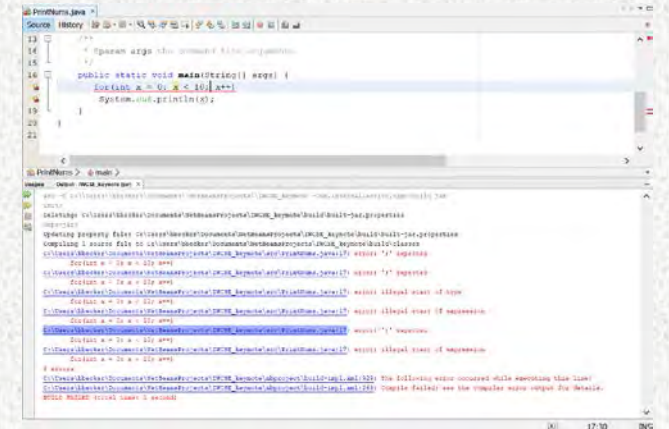
# What I did not discuss and could be even more fundamental

5. **What about pedagogy?**

- We take feedback seriously in the classroom
- Shouldn't the feedback that environments provide be backed by sound pedagogical principles?

Luxton-Reilly, Becker, et al., 2017: https://dl.acm.org/citation.cfm?id=3174784

# A few resources

Available at [www.brettbecker.com/talks-panels-etc/](www.brettbecker.com/talks-panels-etc/)
  - Slides
  - References mentioned here, and others related to this topic

# Questions?