UCSD CSE 15L

Java Debugging and the Scientific Method

Week 2
October 5, 2011

Quiz today

1. Keep your quiz sheet upside down.
2. At the announcement, turn it over and start.
3. Write on the front and, if necessary, on the back.
4. You have ***seven*** minutes.
5. Turn the quiz over again, at the announcement.

Today’s quiz question

When you are debugging a Java program, often it is a good idea to insert a print command to see how the value of a variable changes. For example, you might use the command

```
    System.out.println("the value of x is " + x);
```

However, it is often a better idea to put the print command inside an if command like the following:

```
if (flag)
    { System.out.println("the value of x is " + x); }
```

[6 points] Explain why the latter is a better idea. What should the type of flag be? Should flag be a constant or a variable? [You may ignore any possible mistakes in Java syntax.]

Lab 2

The procedure for the lab this week is similar to last week.

In lecture now, I will talk about debugging strategies. These will be useful in today's lab.

The strategies should sound sensible to you, based on your experience in the first lab.

Hint: This week's program is similar in some ways to last week's. Learn from experience!

Four debugging strategies

1. Remember basic facts about Java.
2. Look for patterns in symptoms.
3. Search the web to find an answer to your problem.
4. Understand the concepts of Java, e.g. the difference between a class and an object.

A fact about Java (example)

The operator == tests for identity, not for equal contents.

So, don’t use it to test whether strings are equal.

Instead, use the right Java library method:

```
if (arg.equals("--help"))
        usage();
```

Also don’t use == for assignment!

Look for patterns in symptoms

Suppose you get many error messages like this:

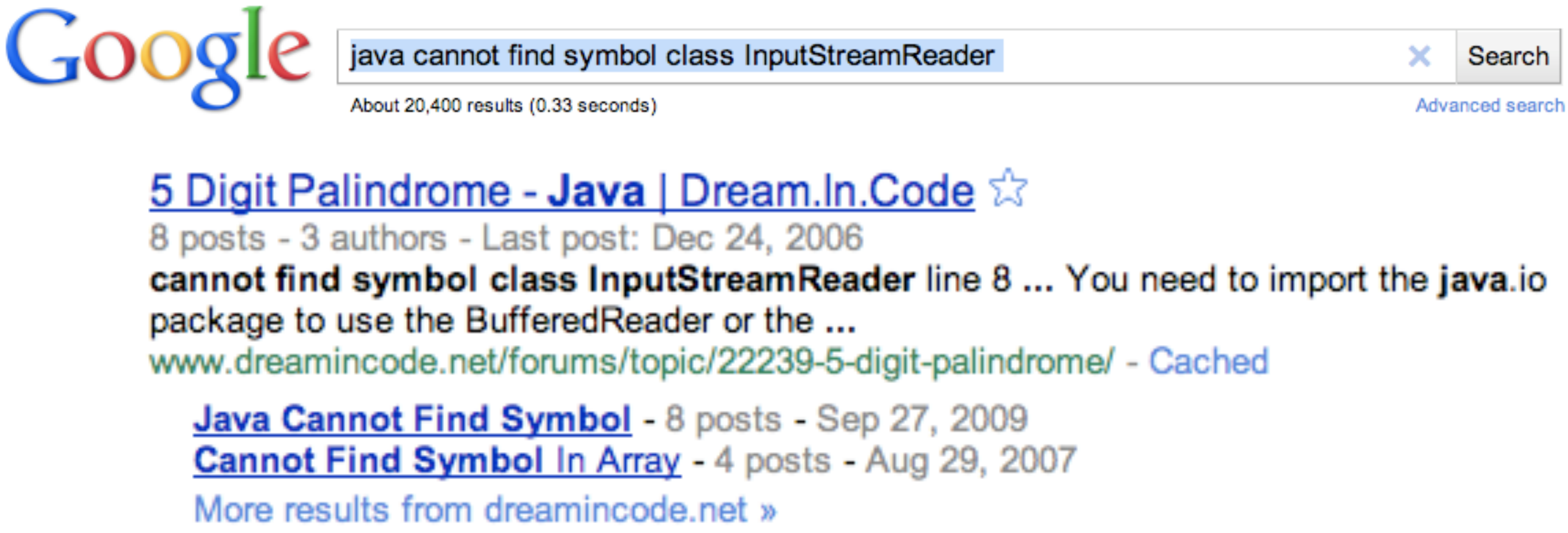
```
cannot find symbol
symbol  : class InputStreamReader
location: class Lab1
       in = new BufferedReader(new InputStreamReader(System.in));
```

Most likely they share a ***single*** common cause.

Construct a Bing query that captures the pattern, e.g.

```
       cannot find symbol class InputStreamReader
```

Web search results



java cannot find symbol class InputStreamReader

About 20,400 results (0.33 seconds)

5 Digit Palindrome - **Java** | Dream.In.Code ☆
8 posts - 3 authors - Last post: Dec 24, 2006
cannot find symbol class InputStreamReader line 8 ... You need to import the **java**.io package to use the BufferedReader or the ...
www.dreamincode.net/forums/topic/22239-5-digit-palindrome/ - Cached

Java Cannot Find Symbol - 8 posts - Sep 27, 2009
Cannot Find Symbol In Array - 4 posts - Aug 29, 2007
More results from dreamincode.net »

Of course, you must use good judgment to figure out if an answer is relevant and correct!

Principles of Java (example)

Understand the ***concepts*** of Java, e.g. the difference between a class and an object.

Important question to consider for every instance variable:

- Should it belong to the class, or to each object?

Feedback from last week

Overall, most students did well on reasoning, but less well on clear communication.

Many students need to do more testing to show that the fixed program works correctly.

In reports, always explain the ***reasoning*** behind changes. Provide concrete details to support explanations.

Grading rubrics

Each box in a rubric has a letter code: the graders have written corresponding letters on submissions.

The rubrics are ***unchanged*** from last week, so you can learn from feedback.

Important: For last week, the graders were generous.

The directions above the rubrics have been clarified slightly.

Common mistakes in notes

Copy and paste inputs, javac and java commands, and outputs into notes.

Use a readable format for code changes. For example: “Change line 18 from `if (args.length()==0)` to `if (args.length==0)`”

Use clear organization to show what you are doing. For example:

BUG 1:

error:

change:

output:

After each change, show the output of the program. Move on to the next bug ***only after*** showing that the change fixed the bug.

Common mistakes in reports

Explain ***how*** you find the location in the code of the bug you write about. Example: You suspect that the value of variable X is incorrect, so you insert `println` commands to look.

Write clearly and completely. Example: If you add a new variable, explain its type and name, ***exactly*** where it is declared and used, etc.

Write in full English sentences and paragraphs. Mention fragments of code ***only*** if needed. Example: "we change the while loop condition from `while (in != null)` to `while (in.readLine() != null)`".

Have a proper ***Conclusion*** section; see instructions.

Formatting mistakes in submissions

Subject line should be like `Notes 2 emakiuch jil022`

Include the lab number. Use spaces, not symbols like _ and <.

Use ***both*** your UCSD email ids, in alphabetical order.

Do **not** use full email addresses or non-UCSD addresses.

Cc ***both*** of yourselves.

Example of multiple mistakes:

```
Lab_1 k2chen jc022@gmail.com
```

About today's lab

The program named `Uniq.java` should be similar to the Unix `uniq` command.

It should copy an input file to an output file, while removing ***duplicate*** lines that are ***adjacent***.

The definition of duplicate may ***or may not*** ignore UPPER versus lower case.

In the notes for today’s lab

There are quite a few bugs in today’s program, most small.

Write down step by step ***all*** your actions while finding and fixing all these bugs.

Include steps that succeed ***and*** steps that fail.

Don’t write long sentences, but do write notes in English when needed to be clear.

In the report for today

Write about only ***one*** bug: the bug involving incorrect comparison of lines that are next to each other.

Let’s go!

The code for this week’s lab is at

`http://cseweb.ucsd.edu/~elkan/15L/Uniq.java`

Please move to the labs now: B230, 240, 250 and 260.

Pick a partner and get started.