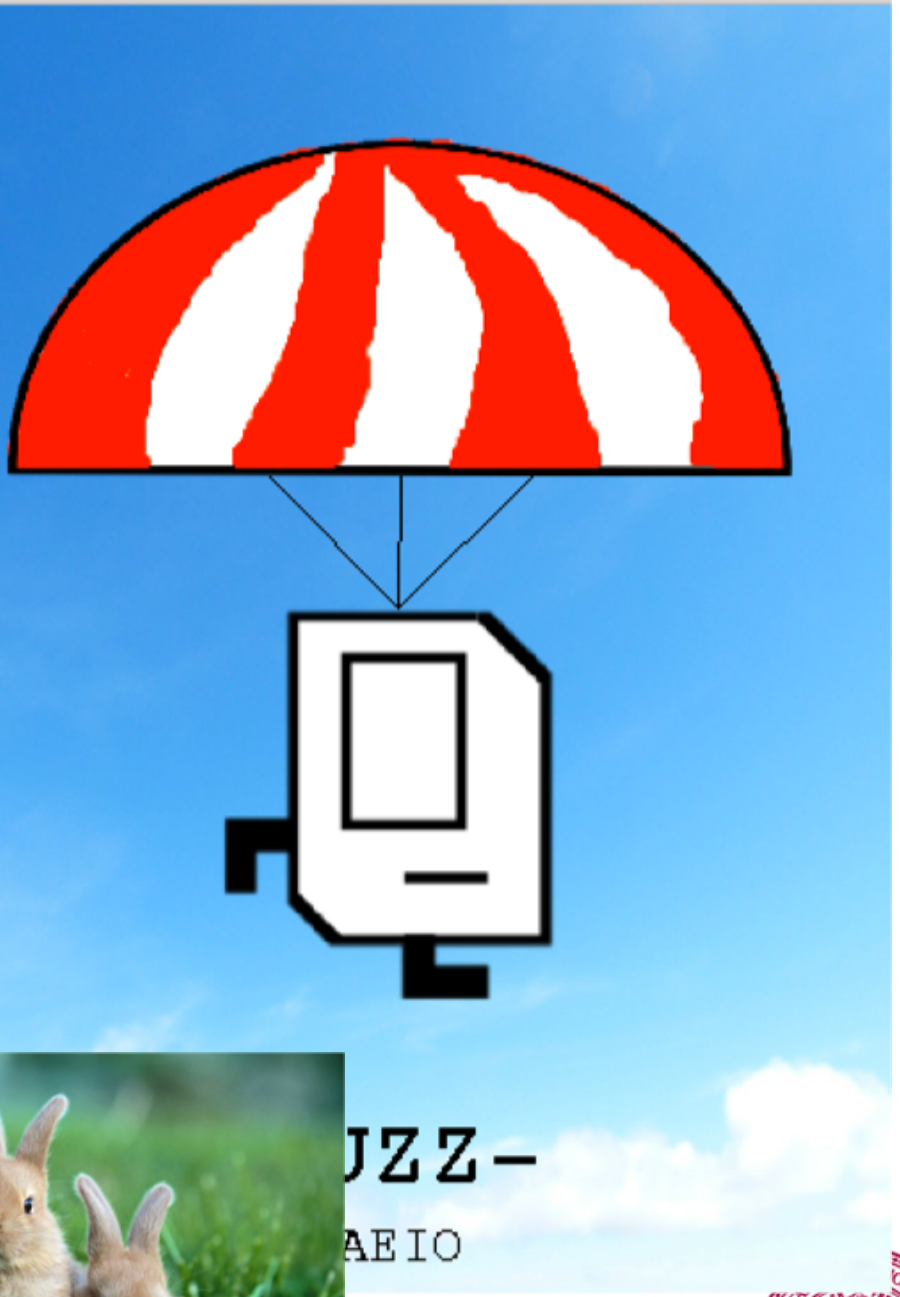
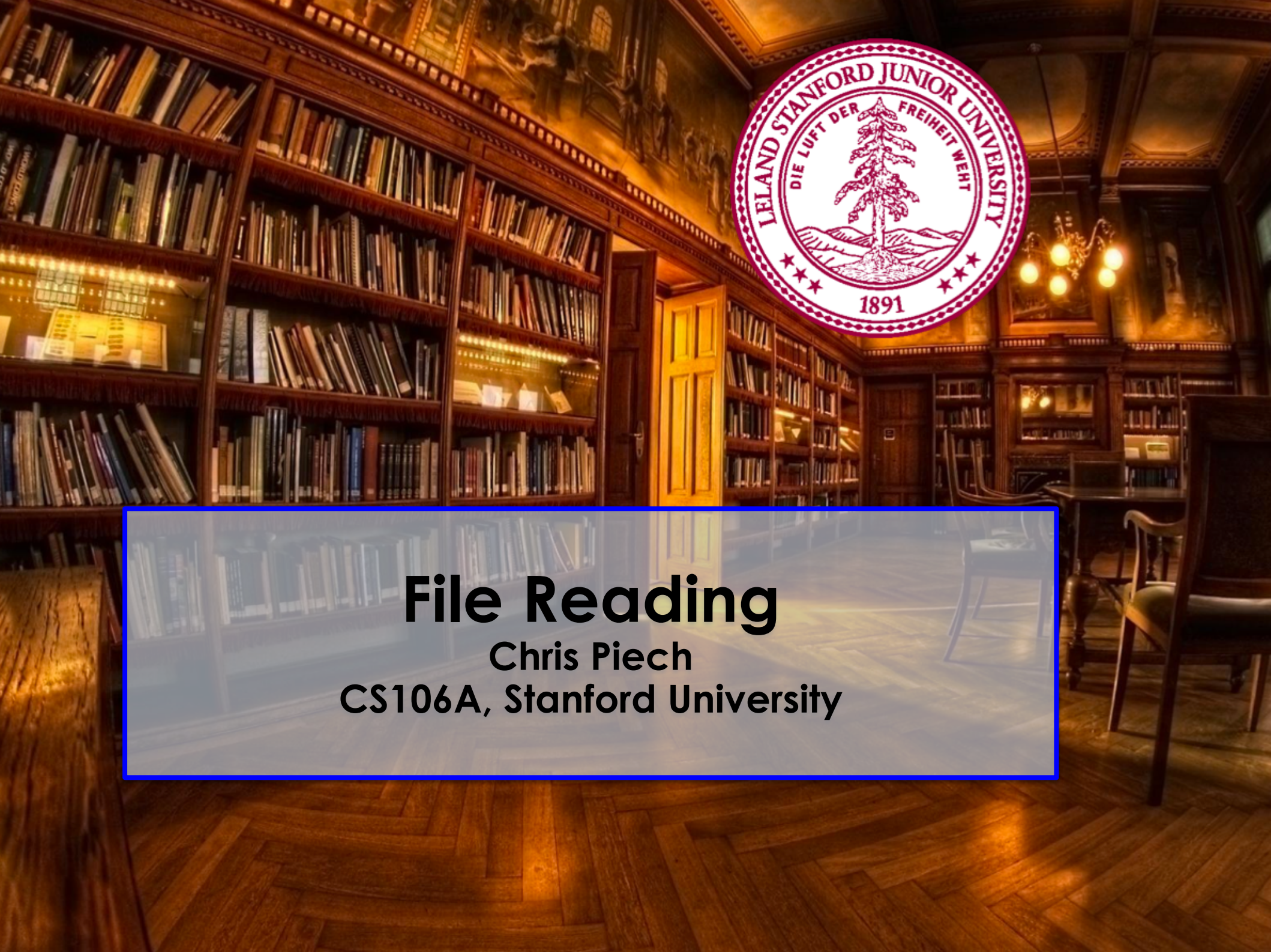
Hangman

```
Welcome to Hangman
Your word looks like this: -----
You have 7 guesses left
Your guess: a
There are no A's in the word.
Your word looks like this: -----
You have 6 guesses left
Your guess: e
There are no E's in the word.
Your word looks like this: -----
You have 5 guesses left
Your guess: i
There are no I's in the word.
Your word looks like this: -----
You have 4 guesses left
Your guess: o
There are no O's in the word.
Your word looks like this: -----
You have 3 guesses left
Your guess: u
That guess is correct.
Your word looks like this: -U---
You have 3 guesses left
Your guess: z
That guess is correct.
Your word looks like this: -UZZ-
You have 3 guesses left
Your guess:
```



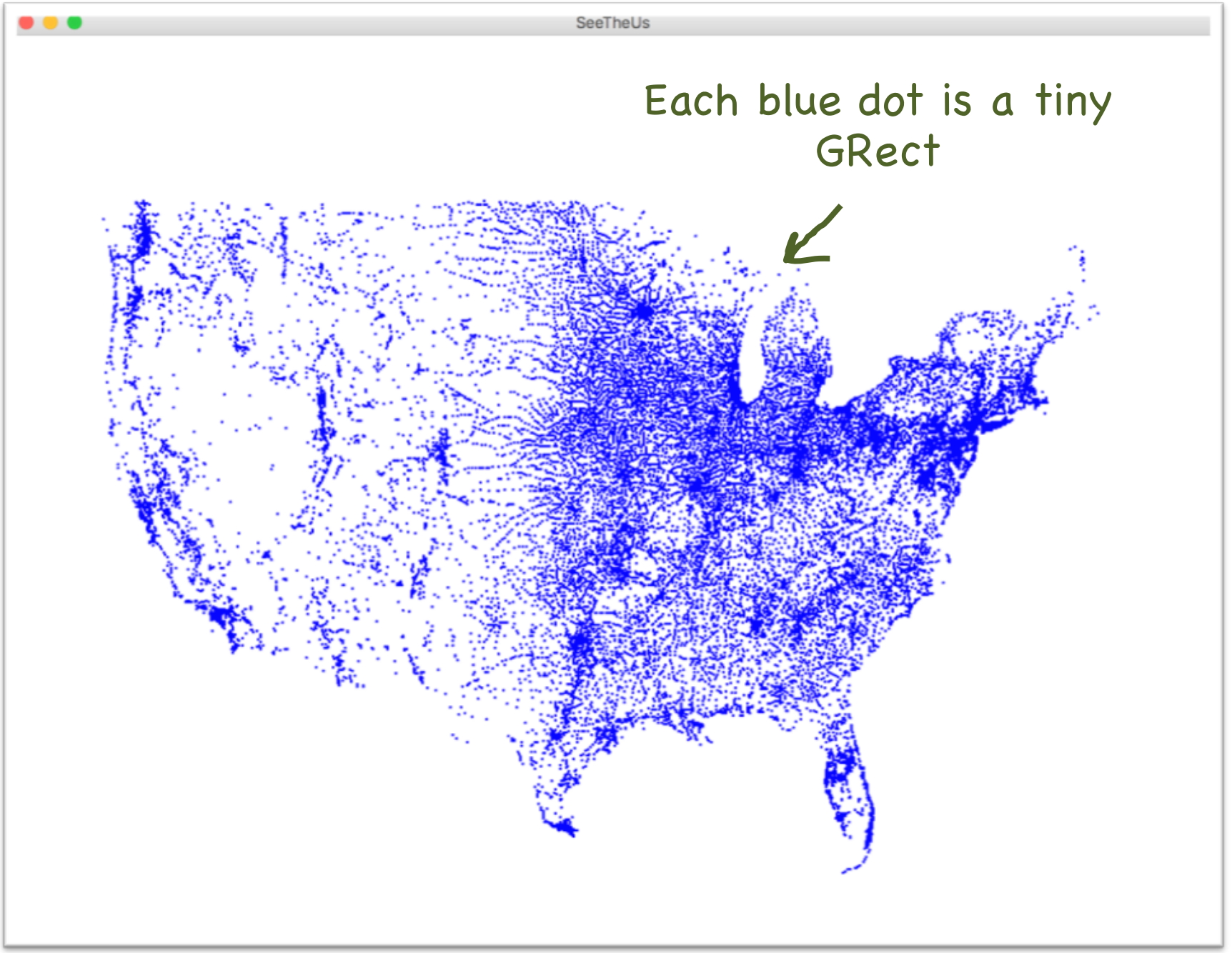


File Reading

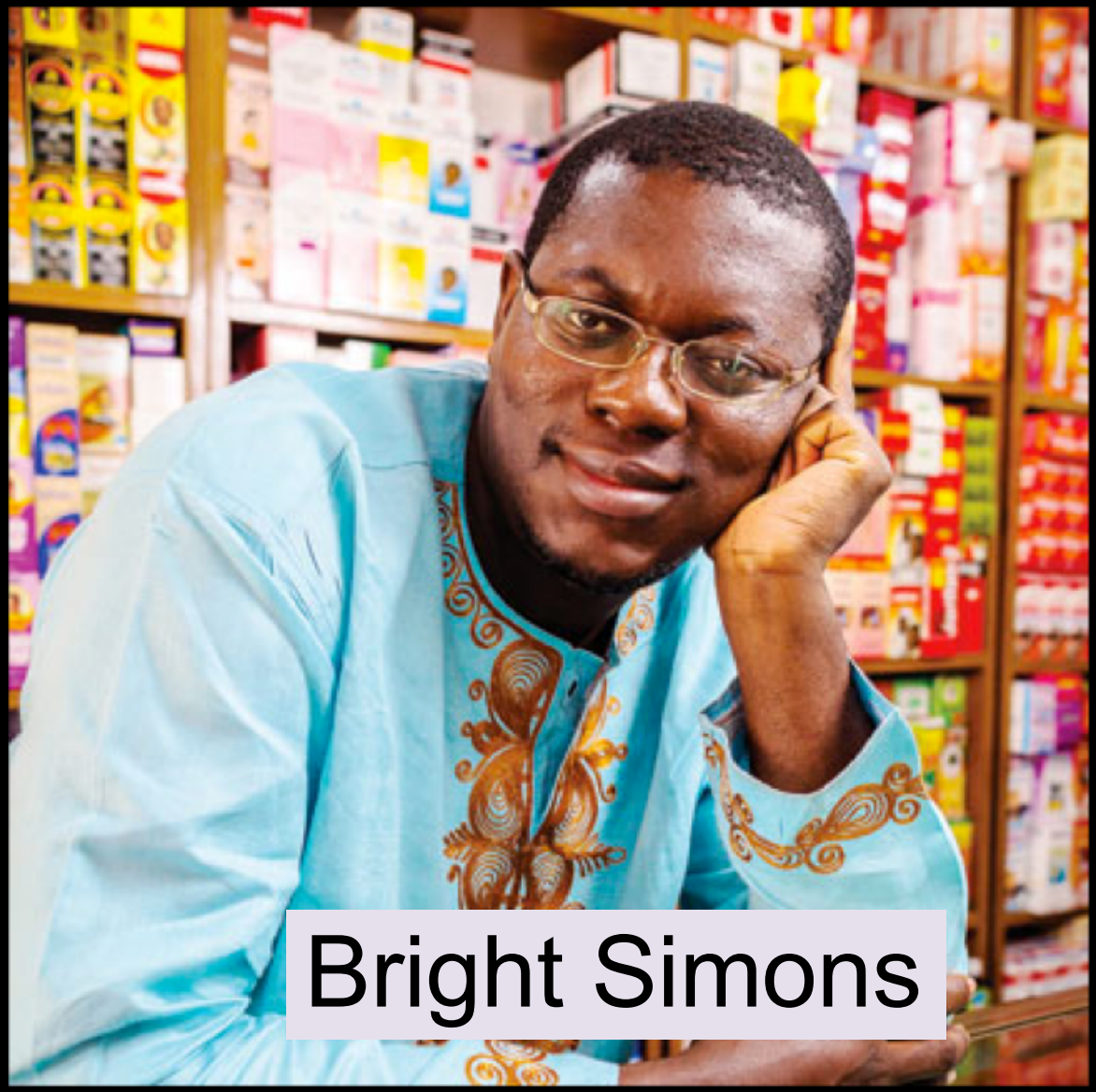
Chris Piech
CS106A, Stanford University

Learning Goals

1. Know how to read a file line by line.


SeeTheUs
Each blue dot is a tiny GRect

Chris' Favorite Program



Underlying Puzzle

Counterfeiter

User

You (Distributor)

Underlying Puzzle

Counterfeiter

User

You (Distributor)

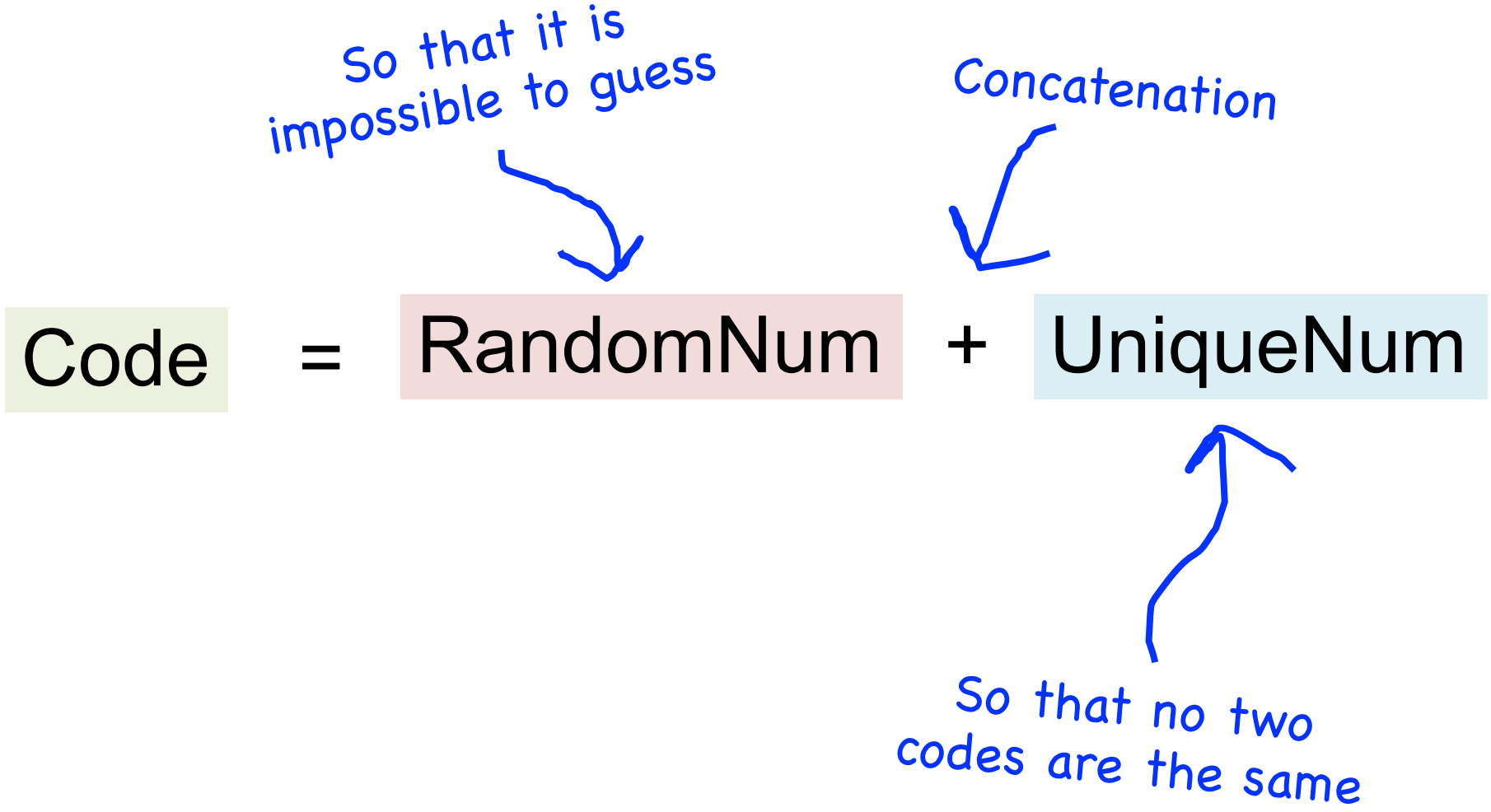
Make a code to
put on every box


Artelum®
Artemether Lumefantrin
40 /240 mg
12 Tablets
Anti-malarial agent
M&B
342302033
BlackBerry
To: 1393
To:
342302033

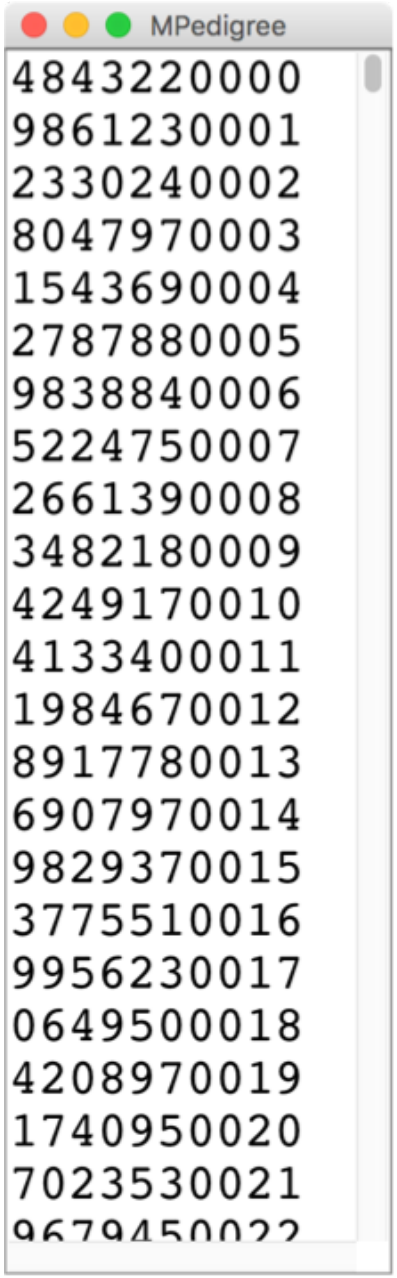
1. Unique

2. Impossible to guess

Insight

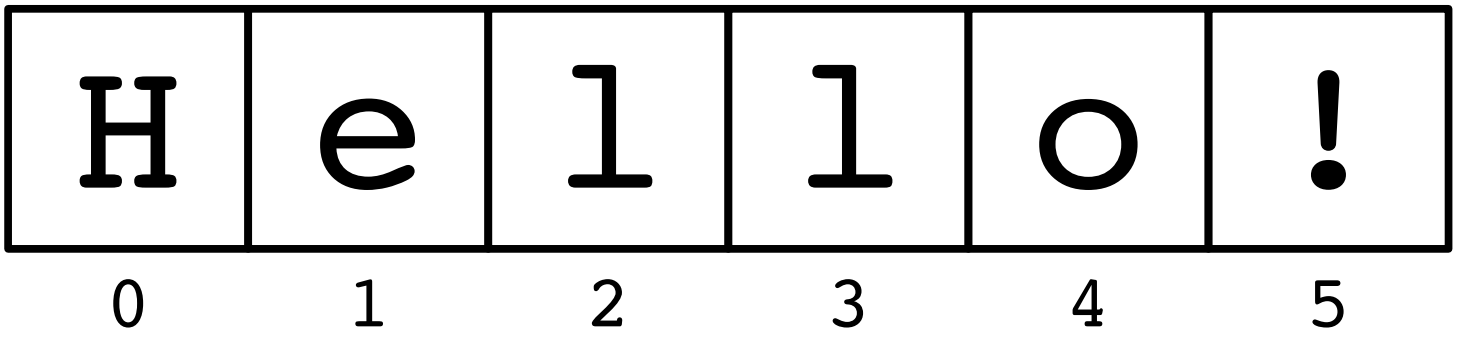

So that it is impossible to guess
Concatenation
Code = RandomNum + UniqueNum
So that no two codes are the same

M-Pedigree


MPedigree
4843220000
9861230001
2330240002
8047970003
1543690004
2787880005
9838840006
5224750007
2661390008
3482180009
4249170010
4133400011
1984670012
8917780013
6907970014
9829370015
3775510016
9956230017
0649500018
4208970019
1740950020
7023530021
9679450022

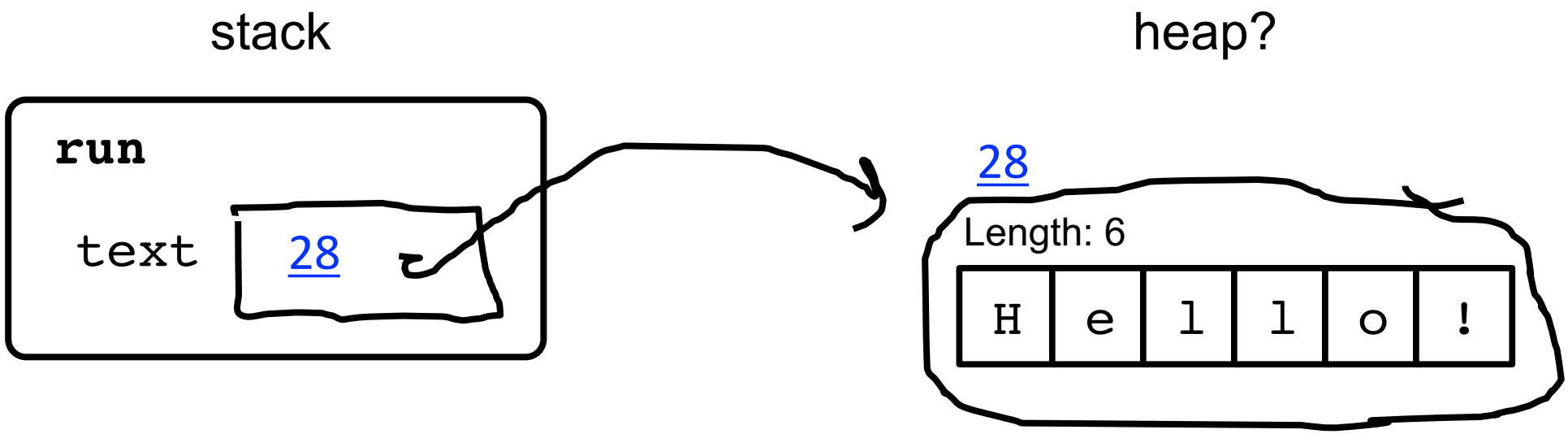
How strings are represented

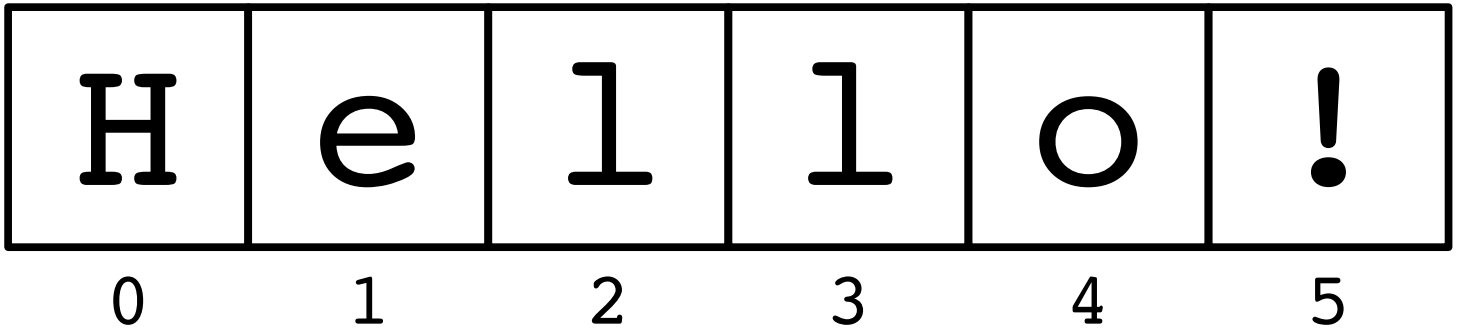
```
public void run() {
    String text = “hello!”;
}
```



How it is actually stored

```
public void run() {
    String text = "hello!";
}
```





***text*.charAt(*index*)**

Recall Palindromes

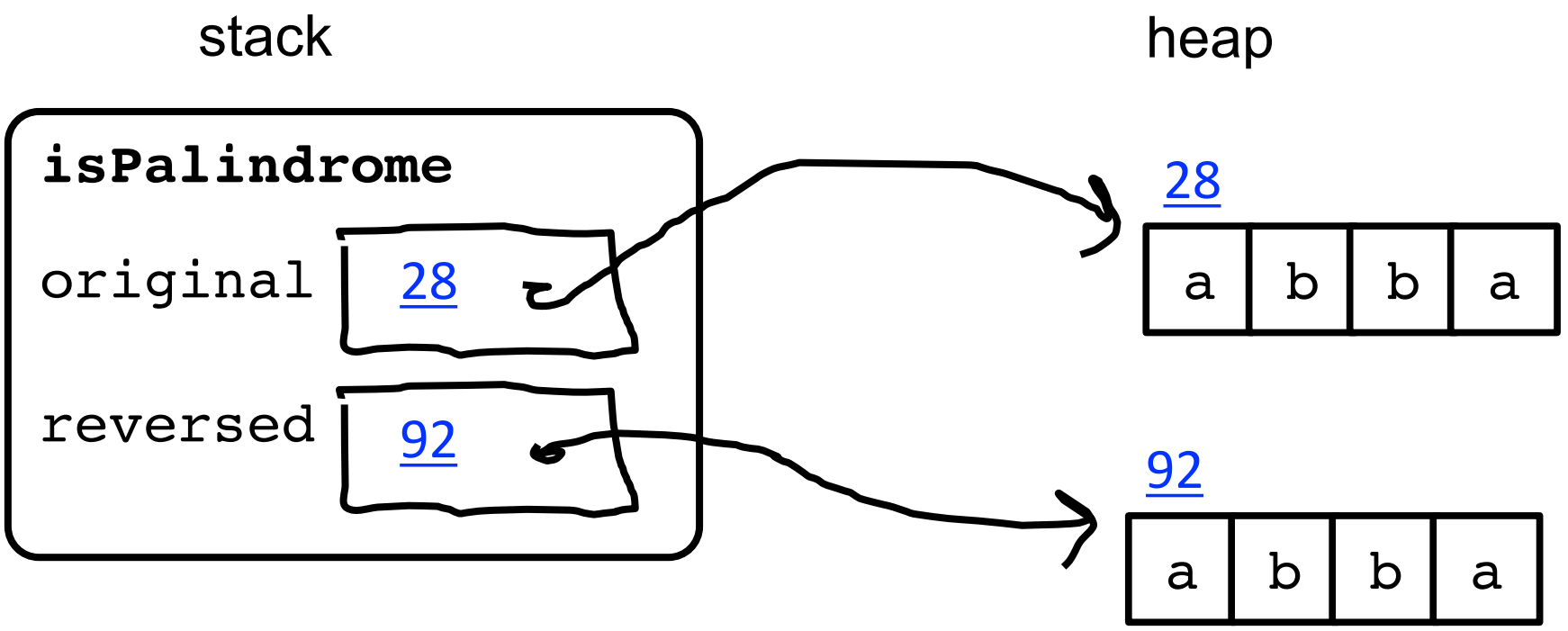
Here are some palindromes in other languages:

- بلح تعلق تحت قلعة حلب (Dates hang underneath a castle in Halab)
- 여보, 안경 안보여 (Honey, I can't see my glasses)
- कड़क (a loud thunderous sound)
- 上海自來水來自海上 (Shanghai tap water originates from "above" the ocean)

The comedian Dmitri Martin also has a routine about palindromes; check it out at https://www.youtube.com/watch?v=0hUHDIOazIU

First try
original == reverse
What went wrong?

```
private boolean isPalindrome(String original) {
    String reversed = reverse(original);
    return reversed == original;
}
```



Use .equals to compare strings, not ==

</ Review>


ATGCTTAAACC..

Human Genome Project

"ATGCCAGGAC"

"GGACTTACATTTTT"

"ATTTTTGGCCGGCC"

The human genome has 3 billion base pairs

Compose Problem

```
strand1  "ATGCCAGGAG"

strand2  "GGAGTTACATTTTT"
```

```
result   "ATGCCAGGAGTTACATTTTT"
```

The human genome has 3 billion base pairs

Ha. Gene was working on
The Genome project ☺

Did Gene Myers define all those little pieces as constants?

File Processing!

Thanks Keith Schwarz for some great slides to build off!

Getting Data into Programs

- Put it directly in the program:
  - Define constants holding your values.
- Get it from the user:
  - Mouse events, nextLine, etc.
- Generate it randomly:
  - Use a RandomGenerator.
- Get it from an external source.
  - Store it in a file and read it later.

Reading Fiels

- Virtually all programs that you've used at some point read files from disk:
  - Word processing (documents)
  - Web browser (cookies)
  - Games (saved progress)
  - Eclipse (Java files)
  - Music player (songs)

The structure of files

- A file is just a series of ***bits*** (ones and zeros).
- Those bits can have structure:
  - Plain-text: Bits represent characters.
  - JPEG: Bits encode information about the structure of an image.
  - MP3: Bits encode frequency information about music.
  - etc.

The structure of files

A file is just a series of ***bits*** (ones and zeros).

Those bits can have structure:

- Plain-text: Bits represent characters.

JPEG: Bits encode information about the structure of an image.

MP3: Bits encode frequency information about music.

etc.

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```


```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

Step one:

Open the file for reading.

Reading in a File

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

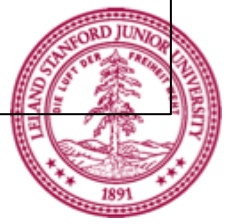
```
Scanner input = new Scanner(new File("mydata.txt"));
```


Reading in a File

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
```

```
import java.util.*;    // for Scanner
import java.io.*;      // for File
```



Reading in a File

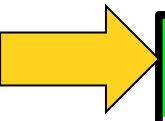
```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
```

Step Two:

Read the file, one line at a time.

Reading in a File

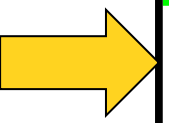


```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));

// Yesterday, upon the stair
String line1 = input.nextLine();
```

Reading in a File

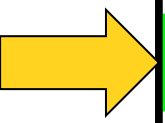


```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));

// Yesterday, upon the stair
String line1 = input.nextLine();
```

Reading in a File



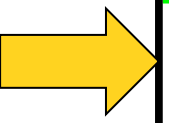
```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));

// Yesterday, upon the stair
String line1 = input.nextLine();

// I met a man who wasn't there
String line2 = input.nextLine();
```

Reading in a File



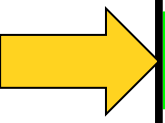
```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));

// "Yesterday, upon the stair"
String line1 = input.nextLine();

// I met a man who wasn't there
String line2 = input.nextLine();
```


Reading in a File

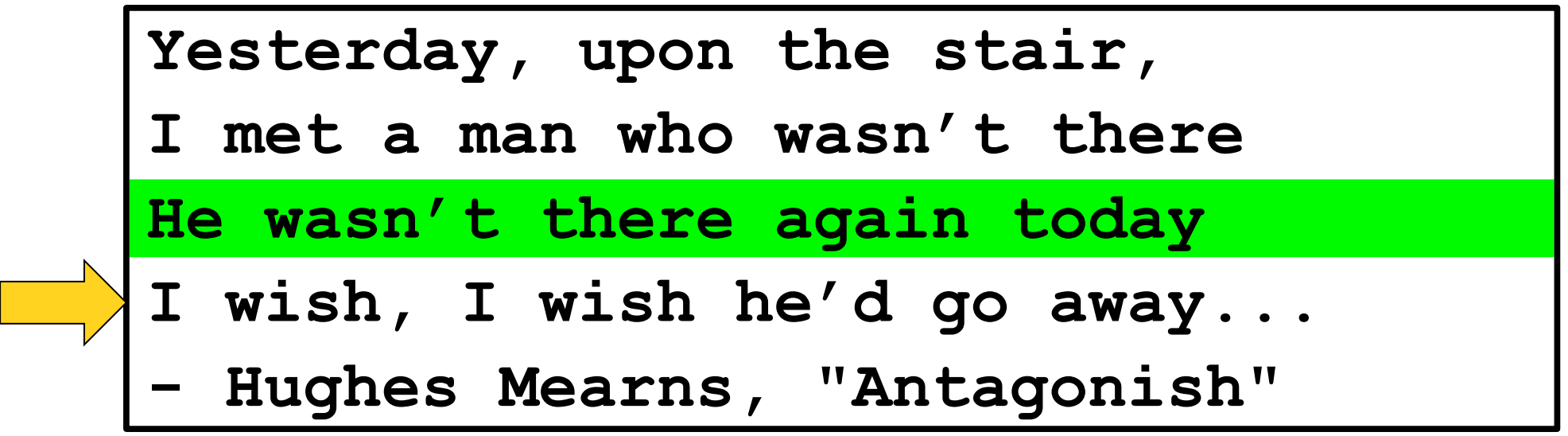


```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));

...
// He wasn't there again today
String line3 = input.nextLine();
```

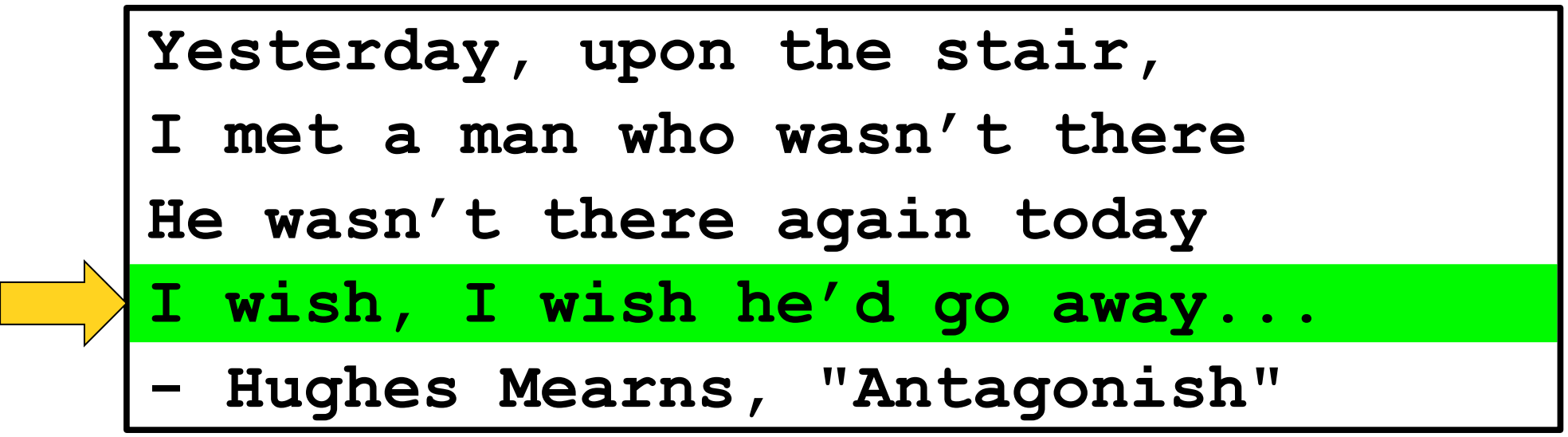
Reading in a File




```
Scanner input = new Scanner(new File("mydata.txt"));

...
// He wasn't there again today
String line3 = input.nextLine();
```

Reading in a File

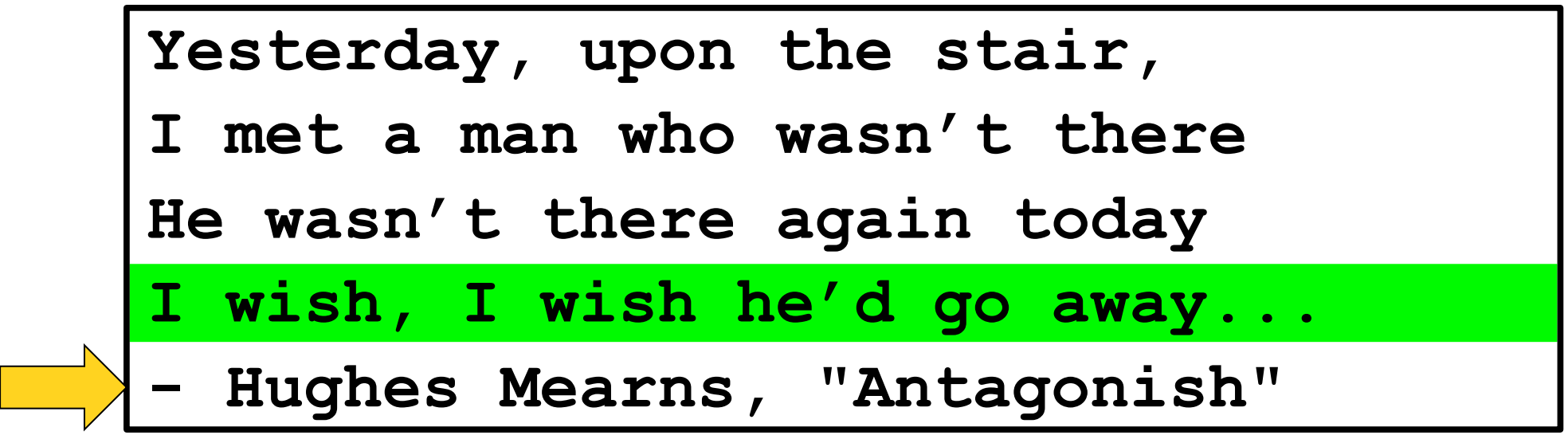



```
Scanner input = new Scanner(new File("mydata.txt"));

...
// He wasn't there again today
String line3 = input.nextLine();

// I wish, I wish he'd go away
String line4 = input.nextLine();
```

Reading in a File

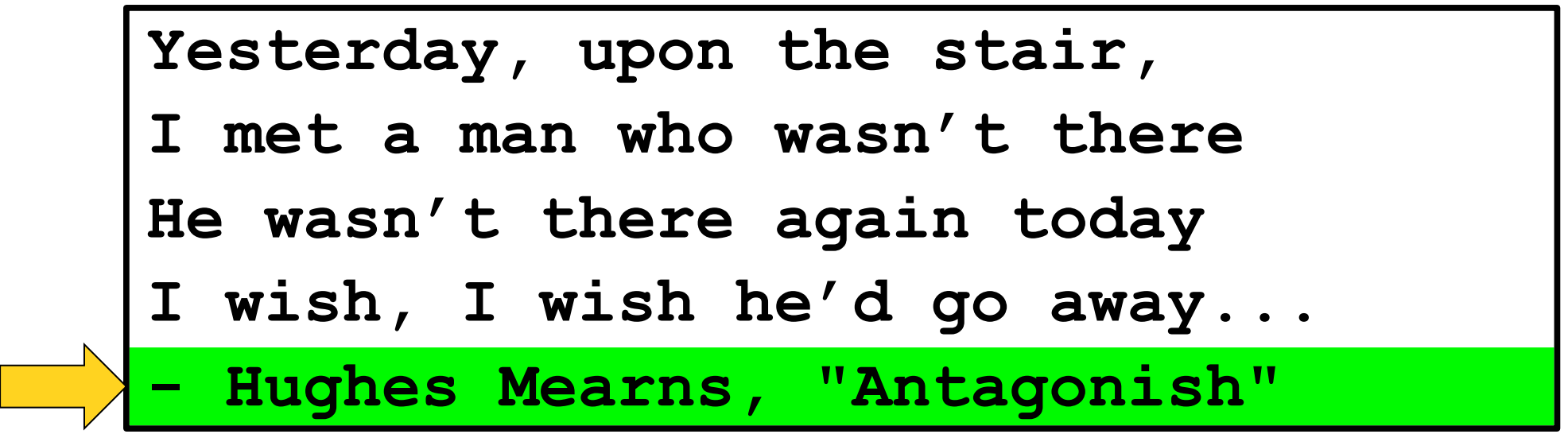



```
Scanner input = new Scanner(new File("mydata.txt"));

...
// He wasn't there again today
String line3 = input.nextLine();

// I wish, I wish he'd go away
String line4 = input.nextLine();
```

Reading in a File




```
Scanner input = new Scanner(new File("mydata.txt"));

...
// - Hughes Mearns, "Antagonish"
String line5 = input.nextLine();
```

Reading in a File

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));

...
// - Hughes Mearns, "Antagonish"
String line5 = input.nextLine();
```

Reading in a File

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));

...
// - Hughes Mearns, "Antagonish"
String line5 = input.nextLine();
```

Reading in a File

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));


...
// prints all lines in the file
while (input.hasNextLine()) {
    String line = input.nextLine();
    println(line);
}
```

Reading in a File

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints all lines in the file
while (input.hasNextLine()) {
    String line = input.nextLine();
    println(line);
}
```

Step Three: close the file.

Reading in a File

```
Yesterday, upon the stair,
I met a man who wasn't there
He wasn't there again today
I wish, I wish he'd go away...
- Hughes Mearns, "Antagonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints all lines in the file
while (input.hasNextLine()) {
    String line = input.nextLine();
    println(line);
}

input.close();
```

Scanner Methods

Method	Description
`sc.nextLine()`	reads and returns a one-*line* `String` from the file
`sc.next()`	reads and returns a one-word `String` from the file
`sc.nextInt()`	reads and returns an `int` from the file
`sc.nextDouble()`	reads and returns a `double` from the file
`sc.hasNextLine()`	returns `true` if there are any more lines
`sc.hasNext()`	returns `true` if there are any more tokens
`sc.hasNextInt()`	returns `true` if there is a next token and it's an `int`
`sc.hasNextDouble()`	returns `true` if there is a next token and it's a `double`
`sc.close();`	should be called when done reading the file

Let “try” it out!

Thanks Keith Schwarz for some great slides to build off!

There's a "catch"

Thanks Keith Schwarz for some great slides to build off!

Sometimes things break

- Programs sometimes encounter unexpected errors.
- Sometimes these are bugs:
  - Dividing by zero.
  - Sending a message to a `null` object.
- Sometimes these are due to external factors:
  - Network errors.
  - Missing files.

Exceptional cases

- If Java encounters a case where it can't proceed as normal, it will cause an ***exception***.
- Java requires that your program handle certain types of exceptions.
- Think of exceptions as rerouting control in an emergency:
  - If all goes well, program continues as usual.
  - If something goes wrong, handle the emergency.

Let “try” it out!

Thanks Keith Schwarz for some great slides to build off!

Let try it out!

Thanks Keith Schwarz for some great slides to build off!

`try`-ing your best

- To use a method or class that might cause an exception, you need to tell Java to **try** its best, knowing that it might fail.

try-ing your best

- To use a method or class that might cause an exception, you need to tell Java to **try** its best, knowing that it might fail.

```
Scanner input =
   new Scanner(new File("poem.txt"));

String line1 = input.nextLine(); // Yesterday, upon the stair,
String line2 = input.nextLine(); // I met a man who wasn't there
String line3 = input.nextLine(); // He wasn't there again today
String line4 = input.nextLine(); // I wish, I wish he'd go away
String line5 = input.nextLine(); // - Hughes Mearns, "Antagonish
String line6 = input.nextLine(); // *Returns null*

input.close();
```


try-ing your best

- To use a method or class that might cause an exception, you need to tell Java to **try** its best, knowing that it might fail.

```
try {
    Scanner input =
       new Scanner(new File("poem.txt"));

    String line1 = input.nextLine(); // Yesterday, upon the stair,
    String line2 = input.nextLine(); // I met a man who wasn't there
    String line3 = input.nextLine(); // He wasn't there again today
    String line4 = input.nextLine(); // I wish, I wish he'd go away
    String line5 = input.nextLine(); // - Hughes Mearns, "Antagonis
    String line6 = input.nextLine(); // *Returns null*

    input.close();
}
```


There's a "catch"

Thanks Keith Schwarz for some great slides to build off!

There's a **catch**

Thanks Keith Schwarz for some great slides to build off!

`try` and `catch` me

- If an exception occurs, you may need to tell Java to **catch** that exception.

try and catch me

- If an exception occurs, you may need to tell Java to **catch** that exception.

```
try {
    Scanner input =
       new Scanner(new File("poem.txt"));

    String line1 = input.nextLine(); // Yesterday, upon the stair,
    String line2 = input.nextLine(); // I met a man who wasn't there
    String line3 = input.nextLine(); // He wasn't there again today
    String line4 = input.nextLine(); // I wish, I wish he'd go away
    String line5 = input.nextLine(); // - Hughes Mearns, "Antagonisl
    String line6 = input.nextLine(); // *Returns null*

    input.close();
}
```


try and catch me

- If an exception occurs, you may need to tell Java to **catch** that exception.

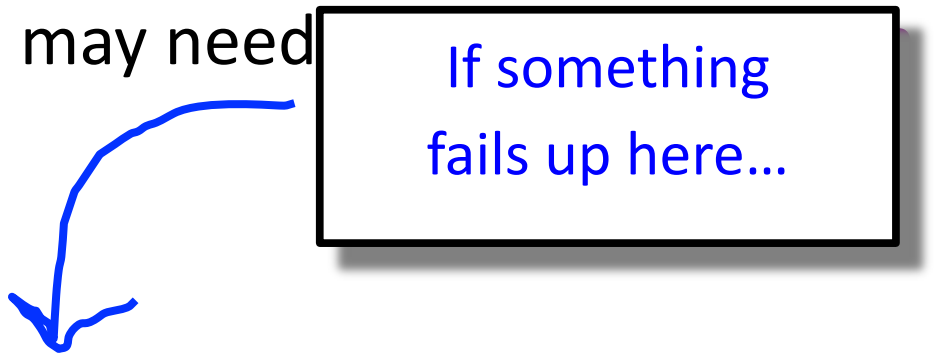
```
try {
    Scanner input =
       new Scanner(new File("poem.txt"));

    String line1 = input.nextLine(); // Yesterday, upon the stair,
    String line2 = input.nextLine(); // I met a man who wasn't there
    String line3 = input.nextLine(); // He wasn't there again today
    String line4 = input.nextLine(); // I wish, I wish he'd go away
    String line5 = input.nextLine(); // - Hughes Mearns, "Antagonisl
    String line6 = input.nextLine(); // *Returns null*

    input.close();
} catch (IOException e) {
    println("An error occurred: " + e);
}
```

try and catch me

- If an exception occurs, you may need that exception.




```
try {
    Scanner input =
       new Scanner(new File("poem.txt"));

    String line1 = input.nextLine(); // Yesterday, upon the stair,
    String line2 = input.nextLine(); // I met a man who wasn't there
    String line3 = input.nextLine(); // He wasn't there again today
    String line4 = input.nextLine(); // I wish, I wish he'd go away
    String line5 = input.nextLine(); // - Hughes Mearns, "Antagonis
    String line6 = input.nextLine(); // *Returns null*

    input.close();
} catch (IOException e) {
    println("An error occurred: " + e);
}
```

try and catch me

- If an exception occurs, you may need that exception.

If something fails up here…

```
try {
    Scanner input =
        new Scanner(new File("poem.txt"));

    String line1 = input.nextLine(); // Yesterday, upon the stair,
    String line2 = input.nextLine(); // I met a man who wasn't there
    String line3 = input.nextLine(); // He wasn't there again today
    String line4 = input.nextLine(); // I wish, I wish he'd go away
    String line5 = input.nextLine(); // - Hughes Mearns, "Antagonis
    String line6 = input.nextLine(); // *Returns null*

    input.close();
} catch (IOException e) {
    println("An error occurred: " + e);
}
```

… we immediately jump down here.

try and catch me

- If an exception occurs, you may need to catch that exception.

If something fails up here…

```
try {
    Scanner input =
        new Scanner(new File("poem.txt"));

    String line1 = input.nextLine(); // Yesterday, upon the stair,
    String line2 = input.nextLine(); // I met a man who wasn't there
    String line3 = input.nextLine(); // He wasn't there again today
    String line4 = input.nextLine(); // I wish, I wish he'd go away
    String line5 = input.nextLine(); // - Hughes Mearns, "Antagonis
    String line6 = input.nextLine(); // *Returns null*

    input.close();
} catch (IOException e) {
    throw new RuntimeException(e);
}
```

… we immediately jump down here.

File concepts in one slide

1. Make a Scanner (lets call it input) to open a file for reading

```
Scanner input = new Scanner(new File("poem.txt"));
```

2. Use scanner.nextLine to get one line from the file

```
input.nextLine(); // returns the next line
```

3. Both the above operations are “dangerous” so we need to use a try/catch loop

```
try{
   // live dangerously
} catch (Exception e){
   // have heath insurance
}
```

4. You can either handle the problem or throw a runtime exception

```
throw new RuntimeException(“AHHHH!”);
```

lets **throw** it all together.

Thanks Keith Schwarz for some great slides to build off!

The classic file reading program.

- The idiomatic "read all the lines of a file" code is shown here:

```
try {
    Scanner input = /*…open the file… */
    while (input.hasNextLine()) {
        String line = input.nextLine();
        /* … process current line … */
    }
    input.close();
} catch (IOException e) {
    throw new RuntimeException(e);
}
```

Understanding this code is about 95% of what we want you to know for files in CS106A

US Census Data

Thanks Keith for the cool dataset