CS 106A, Lecture 10
File Reading

reading:
Art & Science of Java, 12.4

Plan For Today

- Practice: Caesar Cipher
- File Processing
- Try-Catch
- Practice: Election Results

Plan For Today

- Practice: Caesar Cipher
- File Processing
- Try-Catch
- Practice: Election Results

Exercise: Caesar Cipher

- Rotate alphabet by *n* letters (*n* = 3 in below)
  - *n* is called the **key**
- Wrap-around at the end
- Substitute letters based on this mapping

original	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
encrypt	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Exercise: Caesar Cipher

- Rotate alphabet by a certain key, with wrapping

original	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
encrypt	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

CaesarCipher [completed]

```
This program uses a Caesar cipher for encryption.
Enter encryption key: 5
Plaintext:  Shhh!  This is a secret message.
Ciphertext: XMMM!  YMNX NX F XJHWJY RJXXFLJ.
Decrypted text: SHHH!  THIS IS A SECRET MESSAGE.
```

Plan For Today

- Practice: Caesar Cipher
- File Processing
- Try-Catch
- Practice: Election Results

Why Files?

Files are cool! They provide us another place to read in text, besides prompting the user. They can store a lot of information that can easily be read in and processed.

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)

What Are 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.

What Are 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.

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, "Antigonish"
```

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, "Antigonish"
```

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, "Antigonish"
```

```
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, "Antigonish"
```

```
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, "Antigonish"
```

```
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, "Antigonish"
```

```
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, "Antigonish"
```

```
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, "Antigonish"
```

```
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, "Antigonish"
```

```
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, "Antigonish"
```

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

...
// He wasn't there again today
String line3 = 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, "Antigonish"
```

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

...
// He wasn't there again today
String line3 = 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, "Antigonish"
```

```
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

```
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, "Antigonish"
```

```
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

```
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, "Antigonish"
```

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

...
// - Hughes Mearns, "Antigonish"
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, "Antigonish"
```

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

...
// - Hughes Mearns, "Antigonish"
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, "Antigonish"
```

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

...
// - Hughes Mearns, "Antigonish"
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, "Antigonish"
```

```
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, "Antigonish"
```

```
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, "Antigonish"
```

```
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

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, "Antigonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints each word on its own line
while (input.hasNext()) {
    String word = input.next();
    println(word);
}

input.close();
```

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, "Antigonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints each word on its own line
while (input.hasNext()) {
    String word = input.next();
    println(word);
}

input.close();
```

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, "Antigonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints each word on its own line
while (input.hasNext()) {
    String word = input.next();
    println(word);
}

input.close();
```

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, "Antigonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints each word on its own line
while (input.hasNext()) {
    String word = input.next();
    println(word);
}

input.close();
```

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, "Antigonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints each word on its own line
while (input.hasNext()) {
    String word = input.next();
    println(word);
}

input.close();
```

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, "Antigonish"
```

```
Scanner input = new Scanner(new File("mydata.txt"));
...
// prints each word on its own line
while (input.hasNext()) {
    String word = input.next();
    println(word);
}

input.close();
```

Reading tokens

- Calling `nextDouble` etc. skips whitespace and reads one token.

```
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
^

double d1 = input.nextDouble();   // 16.2
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
       ^

double d2 = input.nextDouble();   // 23.2
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
            ^

String s1 = input.next();         // "19.2"
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
                    ^

String s2 = input.next();         // "7.7"
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
                       ^
```

Reading lines

- When you read a line, the cursor advances past the next \n marker.

```
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
^

String line = input.nextLine();  // "16.2  23.2"
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
               ^

String line = input.nextLine();  // "  19.2 7.7 22.9"
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
                             ^

String line = input.nextLine();  // ""  (empty)
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
                              ^

String line = input.nextLine();  // "18.4  -1.6 14.6 "
16.2   23.2\n   19.2 7.7  22.9\n\n18.4  -1.6 14.6 \n
                                   ^
```

Plan For Today

- Practice: Caesar Cipher
- File Processing
- Try-Catch
- Practice: Election Results

Sometimes Things Break

Programs sometimes encounter unexpected errors.

Sometimes these are bugs:

- Dividing by zero.

Sometimes these are due to external factors:

- Network errors.
- Missing files.

Exceptional Cases

- An ***exception*** occurs if Java encounters a case where it can't proceed as normal.
- 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.
- File processing exceptions: file not found, corrupted, etc.

Try your best…

```
try {
    // code that might throw an exception
    statements;
}
```

...we'll catch you if you fall!

```
try {
    // code that might throw an exception
    statements;
} catch (ExceptionType name) {
    // code to handle the error
    statements;
}
```

Try/Catch

```
try {
    statements;   // code that might throw an exception
} catch (ExceptionType name) {
    statements;   // code to handle the error
}
```

- To execute code that might throw an exception, you must enclose it in a `try/catch` statement.

```
try {
    Scanner input = new Scanner(new File("data.txt"));
    ...
} catch (FileNotFoundException ex) {
    println("Error reading the file: " + ex);
}
```

Try/Catch

To execute code that might throw an exception, you must enclose it in a `try/catch` statement.

If something fails up here...

```
try {
    Scanner input = new Scanner(new File("data.txt"));
    while (input.hasNextLine()) {
          String line = input.nextLine();
          println(line);
    }
} catch (FileNotFoundException ex) {
    println("Error reading the file: " + ex);
}
```

Try/Catch

To execute code that might throw an exception, you must enclose it in a `try/catch` statement.

If something fails up here...

```
try {
    Scanner input = new Scanner(new File("data.txt"));
    while (input.hasNextLine()) {
        String line = input.nextLine();
        println(line);
    }
} catch (FileNotFoundException ex) {
    println("Error reading the file: " + ex);
}
```

... we immediately jump down here.

File Reading Overview

1. Make a `Scanner` to open a file to read
2. Use `Scanner` methods such as `nextLine` or `next` to read in the file, usually in a loop
3. `Scanner` operations on files are "dangerous", so we need to use a try/catch block
4. Close the `Scanner` when you are done

Uncaught Scanner exceptions

- **`NoSuchElementException`**
  - You read past the end of the input.

- **`InputMismatchException`**
  - You read the wrong type of token (e.g. read `"hi"` as an `int`).

- Finding and fixing these exceptions:
  - Read the exception text for line numbers in your code (the first line that mentions your file):

```
Exception in thread "main" java.util.InputMismatchException
    at java.util.Scanner.throwFor(Scanner.java:838)
    at java.util.Scanner.next(Scanner.java:1347)
    at MyProgram.readFile(MyProgram.java:39)
    at MyProgram.run(MyProgram.java:15)
```

Scanners on Strings

- A `Scanner` can tokenize the contents of a `String` :

```
Scanner name = new Scanner(string);
```

  – Example:

```
String text = "15  3.2 hello   9  27.5";
Scanner scan = new Scanner(text);

int num = scan.nextInt();
println(num);                        // 15

double num2 = scan.nextDouble();
println(num2);                       // 3.2

String word = scan.next();
println(word);                       // hello
```

Scanners on Strings

- A `Scanner` can tokenize the contents of a `String` :

```
Scanner name = new Scanner(string);
```

You do not need a try/catch block, since this is not reading a file!

– Example:

```
String text = "15  3.2 hello   9  27.5";
Scanner scan = new Scanner(text);

int num = scan.nextInt();
println(num);                    // 15

double num2 = scan.nextDouble();
println(num2);                   // 3.2

String word = scan.next();
println(word);                   // hello
```

Mixing lines and tokens

Input file `input.txt` :	Output to console:
`The quick brown fox jumps over` `the lazy dog.`	`Line has 6 words` `Line has 3 words`

```
// Counts the words on each line of a file
Scanner input = new Scanner(new File("input.txt"));
while (input.hasNextLine()) {
    Scanner tokens = new Scanner(input.nextLine());

    // process the contents of this line
    int count = 0;
    while (tokens.hasNext()) {
        String word = tokens.next();
        count++;
    }
    println("Line has " + count + " words");
}
...
```

Prompting for file name

```
// prompt for a file name in the res/ folder
String filename = readLine("Input file name? ");
File inputFile = new File("res", filename);
// make sure inputFile exists, else re-prompt
Scanner input = new Scanner(inputFile);
```

- To ensure that the file exists, you may want to re-prompt…
- Or the method **`promptUserForFile`** handles all of this:

```
// re-prompt for a file name in the res/ folder
String filename = promptUserForFile("Input? ", "res");
Scanner input = new Scanner(new File(filename));
```

Plan For Today

- Practice: Caesar Cipher
- File Processing
- Try-Catch
- Practice: Election Results

Throwing It All Together

- Write a program **`Election`** that reads a file of poll data.

  Format: *State Candidate1% Candidate2% ElectoralVotes Pollster*

```
CT 56 31 7 Oct U. of Connecticut
NE 37 56 5 Sep Rasmussen
AZ 41 49 10 Oct Northern Arizona U.
...
```

- The program should print how many electoral votes each candidate has earned.
  - If they tie in a given region, don't give anybody those votes.

```
Input file? polls.txt
Candidate 1: 325 votes
Candidate 2: 213 votes
```

Election pseudocode

- Get data filename from user
- Open Scanner in a try-catch block
- Read input line-by-line. For each line in the input:
  - Parse the line (separate into tokens)
  - Compare vote percentages and award electoral votes to winner
- Close Scanner

Recap

- Practice: Caesar Cipher
- File Processing
- Try-Catch
- Practice: Election Results

Next time: Graphics programs