Reading From Files:

- You've done this in section, but here is a quick reminder:
- When instantiating a Scanner object you can give it a File object.
- To use a file object you must import **`java.io.File`**

```
import java.io.*:
public class FileExample {
  public static void main(String[] args) {
    File inFile = new File("testFile");
```

- The above code creates a **`File`** object named **`inFile`** which has stored the filename **`testFile`**.
- This code will run without error even if the file **`testFile`** does not exist.

Reading From Files:

- However if you try to add a line to instantiate a Scanner using this File object:

```
import java.io.*:
import java.util.*;
public class FileExample {
  public static void main(String[] args) {
    File inFile = new File("testFile");
    Scanner in = new Scanner(inFile);
```

- The code will not compile. This is because when the Scanner is created it might throw an error that you must either acknowledge or handle.

Reading From Files:

- This code will compile, but it does not handle the error (not good)

```
import java.io.*:
import java.util.*;
public class FileExample {
  public static void main(String[] args) throws IOException {
    File inFile = new File("testFile");
    Scanner in = new Scanner(inFile);
```

- It would be better to catch the exception and deal with it.

Reading From Files:

- Java has a try/catch that works very similar to Python

```
import java.io.*:
import java.util.*;
public class FileExample {
  public static void main(String[] args) {
    File inFile = new File("testFile");
    try {
      Scanner in = new Scanner(inFile);
    }
    catch (IOException e){
      // Do something here to handle the error
    }
```

- We will talk more about try/catch later, but for now you can use it to catch IOExceptions when trying to open files.