DS2000
Fall 2021
Handout: File Processing

All programming languages must be able to:

1. remember things,
2. repeat things,
3. communicate, and
4. make decisions.

File processing is all about #3: **communicating**.

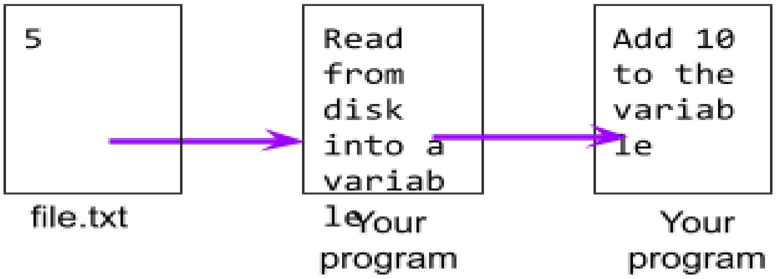
We started out the semester by getting data from the user, with the function *input*. There's another, better way to get data though, and that's from a file. Typically, in any data science project, you'll download data files from a trusted source and then apply your data science knowledge and skill to make sense of it.

For most homeworks this semester, data will come from files instead of from the user.

Memory vs. Disk

When you write a Python program that either reads from or writes to files, there's a relationship between memory and disk. If a file contains information that we want to process, we first need to get data from the file (disk) into a variable (memory) so that your program can operate on it. There's no way for your program to *directly* process data on disk.

For example, suppose you have a file called file.txt and it contains just one number. You want to write a program that adds 10 to this number. Your program needs to do this:



Rules of Thumb

A couple of things to bear in mind when processing files:

1. Files are processed sequentially
2. Save your file in the same directory as your Python program.

The first rule of thumb means we can't jump around within a file. We read the first line, then the second, and so on. Top-to-bottom, left-to-right, just the same way you'd read a book. If you want to get to some data in the middle of a file, you need to work your way through the file to get there.

The second rule of thumb is actually just to make our lives easier. Whatever file you're processing in a Python program, save it in the same directory (aka, folder) as the Python program itself. If Python can't find the file you're looking for, it will throw an error. It's easier and minimizes errors if we keep Python programs and files all together in the same place.

Make sure you also set your Spyder *working directory* to the same location as your code and your text file. Otherwise, Spyder gets confused about where to look for the file you're reading. Look for the "open" icon in the upper-right of Spyder, and select the correct directory.

Reading Text Files

Python has a lot of file-reading capabilities and tons of stuff it can do. Later in the semester we'll see how to work with comma-separated value (CSV) files, messy text files, and more. For now, our first use of files, we'll keep it nice and simple.

Right now, our files will look like this, with one piece of data per line, like this:

```
90
97
93
```

grades.txt

This file is named *grades.txt* and I've saved it in the same directory as my code. I want to read in all the numbers so I can calculate their average.

Steps for Reading a File

1. *Open the file (for reading)*

```
infile = open('grades.txt', 'r')
```

Variable name is **infile**, which is a file object. It's a variable, so it can be named anything. I like to use **infile** when reading and **outfile** when writing.

'`grades.txt`' is the name of the file we're reading. Note that I write the entire filename, including the extension, .txt.

`'r'` specifies the mode, which here indicates reading.

2. *Read the File*

Python gives us a few options here. I'm going to use the version of reading called *readline*, which processes one line at a time. Each line in the file becomes one variable in my program.

```
infile = open('grades.txt', 'r')
grade_one = int(infile.readline())
grade_two = int(infile.readline())
grade_three = int(infile.readline())
```

3. *Process the data*

After executing this little code snippet, the variables *grade_one, grade_two,* and *grade_three* each contain a single number, and I can treat them like any other variables. So I’m done with the file! I put the rest of my code outside of the *with/open/as* structure:

```
infile = open('grades.txt', 'r')
grade_one = int(infile.readline())
grade_two = int(infile.readline())
grade_three = int(infile.readline())
grade_sum = grade_one + grade_two + grade_three
avg = grade_sum / 3
```