

Introduction To Files In **Python**

In this section of notes you will learn how to read from and write to files in your programs.

James Tam

Why Bother With Files?

- Many reasons:
 - Too much information to input all at once
 - The information must be persistent (RAM is volatile)
 - Data entry of information is easier via a text editor rather than through the computer program that you write.
 - Etc.

James Tam

What You Need In Order To Read Information From A File

1. Open the file and associate the file with a file variable
2. A command to read the information

James Tam

1. Opening Files

Prepares the file for reading:

- A. Links the file variable with the physical file (references to the file variable are references to the physical file).
- B. Positions the file pointer at the start of the file.

Format:¹

`<file variable> = open (<file name>, "r")`

Example:

(Constant file name)

```
inputFile = open ("data.txt ", "r")
```

OR

(Variable file name: entered by user at runtime)

```
filename = raw_input ("Enter name of input file: ")  
inputFile = open (filename, "r")
```

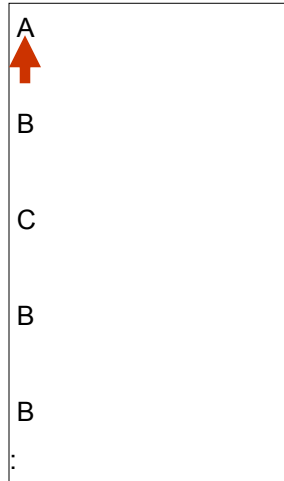
¹ Assumes that the file is in the same directory/folder as the Python program.

James Tam

B. Positioning The File Pointer

letters.txt

```
A
B
C
B
B
:
```



James Tam

2. Reading Information From Files

Typically reading is done within the body of a loop

Format:

```
for <variable to store a string> in <name of file variable>:
    <Do something with the string read from a file>
```

Example:

```
for line in inputFile:
    print line
```

James Tam

Reading From Files: Putting It All Together

The complete online version of the program can be found in UNIX under: `/home/231/examples/files/grades.py`

```
inputFileName = raw_input ("Enter name of input file: ")
inputFile = open (inputFileName, "r")
print "Opening file", inputFileName, " for reading."

# While we haven't read past the end of the file continue reading from
# it.
for line in inputFile:
    print line

inputFile.close()
print "Completed reading of file", inputFileName,
```

James Tam

An Alternate Method Of Getting Input

- Command line arguments: inputs given to a program as it's run.
- The complete online version of the program can be found in UNIX under:
`/home/231/examples/files/grades.py`
- Example execution: “python command_line.py first_input 2ndInput”

<< **Filename: command_line1.py** >>

```
import sys
def main ():
    first = sys.argv[0]      # Name of program (command_line.py)
    second = sys.argv[1]    # First input (first_input)
    third = sys.argv[2]     # Second input (2ndInput)
```

James Tam

An Alternate Method Of Getting Input (2)

- The complete online version of the program can be found in UNIX under:
/home/231/examples/files/command_line2.py

<< **Filename: command_line2.py** >>

```
import sys
def main ():
    arguments = sys.argv[0:]
    for argument in arguments:
        print argument
main ()
```

James Tam

What You Need To Write Information To A File

1. Open the file and associate the file with a file variable
2. A command to write the information

James Tam

1. Opening The File

Format:

```
<name of file variable> = open (<file name>, "w")
```

Example:

(Constant file name)

```
outputFile = open ("gpa.txt", "w")
```

(Variable file name: entered by user at runtime)

```
outputFileName = raw_input ("Enter the name of the output file to record  
the GPA's to: ")
```

```
outputFile = open (outputFileName, "w")
```

James Tam

3. Writing To A File

Format:

```
outputFile.write (temp)
```

Example:

Assume that temp contains a string of characters.

```
outputFile.write (temp)
```

James Tam

Writing To A File: Putting It All Together

- The complete online version of the program can be found in UNIX under:

/home/231/examples/files/grades2.py

```
inputFileName = raw_input ("Enter the name of input file to read the grades from: ")
```

```
outputFileName = raw_input ("Enter the name of the output file to record the GPA's to: ")
```

```
# Open file for reading
```

```
inputFile = open (inputFileName, "r")
```

```
outputFile = open (outputFileName, "w")
```

```
# Update user on what is happening.
```

James Tam

Writing To A File: Putting It All Together (2)

```
gpa = 0
```

```
for line in inputFile:
```

```
    if (line[0] == "A"):
```

```
        gpa = 4
```

```
    elif (line[0] == "B"):
```

```
        gpa = 3
```

```
    elif (line[0] == "C"):
```

```
        gpa = 2
```

```
    elif (line[0] == "D"):
```

```
        gpa = 1
```

```
    elif (line[0] == "F"):
```

```
        gpa = 0
```

```
    else:
```

```
        gpa = -1
```

James Tam

Writing To A File: Putting It All Together (3)

```
temp = str (gpa)
temp = temp + '\n'
print letter[0], '\t', gpa
outputFile.write (temp)
```

```
inputFile.close ()
outputFile.close ()
print "Completed reading of file", inputFile.name,
print "Completed writing to file", outputFile.name,
```

James Tam

Another Example Reading From A File Into A String

- The complete online version of the program can be found in UNIX under:
`/home/231/examples/files/file_list.py`

```
inputFile = open ("input.txt", "r")
for line in inputFile:
    i = 0
    for ch in line:
        print i, ch,
        i = i + 1
    print
```

James Tam

Building An Arbitrary Sized List By Reading From File

- The complete online version of the program can be found in UNIX under:

/home/231/examples/files/file_list2.py

```
inputFile = open ("input2.txt", "r")
```

```
myList = []
```

```
for line in inputFile:
```

```
    myList.append(line)
```

```
inputFile.close()
```

James Tam

Building An Arbitrary Sized List By Reading From File (2)

```
row = 0
```

```
for line in myList:
```

```
    if (row < 10):
```

```
        print row, line,
```

```
    else:
```

```
        temp = row + 55
```

```
        ch = chr(temp)
```

```
        print ch, line,
```

```
    row = row + 1
```

James Tam

Error Handling With Exceptions

- Exceptions are used to deal with extraordinary errors.
- Typically these are fatal runtime errors.
- Example: trying to open a non-existent file

James Tam

Exceptions: Example

- An executable version can be found in UNIX under:
/home/231/examples/files/file_exception.py

```
fileOpened = False
while (fileOpened == False):
    inputFilename = raw_input ("Enter the name of the input file: ")
    try:
        inputFile = open (inputFilename, "r")
        fileOpened = True
    except IOError:
        print "Could not open file", inputFilename
```

James Tam

Exceptions: Example (2)

```
for line in inputFile:  
    print line,  
  
inputFile.close()
```

James Tam

You Should Now Know

- How to open a file for reading
- How to open a file a file for writing
- The details of how information is read from and written to a file
- How to close a file and why it is good practice to do this explicitly
- What is a command line argument
- How to read and use command line arguments
- How to read from a file of arbitrary size
- How to build an arbitrary sized list by reading the information from a file
- How exceptions can be used in conjunction with file input

James Tam