

Strings and Files

David H Smith IV

University of Illinois Urbana-Champaign

Wed, July 13 2021

Reminders

Reminders

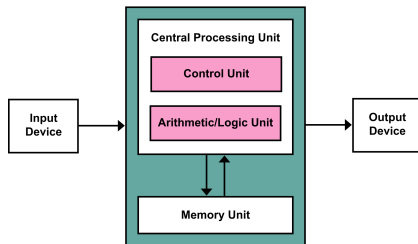
The following are due Friday

- Homework 9p1
- Participation 9p2
- Post-reading 9p2

Files

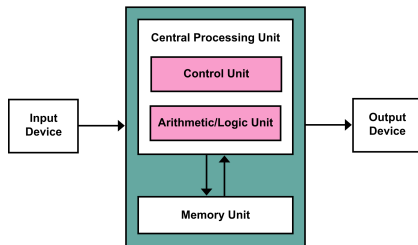
Why do we use files?

- Files are how data are stored on external storage
- Managed by O/S
- Disk is slow
- Opening files moves them to memory for the program
- Data is buffered (temporarily stored) in memory



Why do we use files?

- Files are how data are stored on external storage
- Managed by O/S
- Disk is slow
- Opening files moves them to memory for the program
- Data is buffered (temporarily stored) in memory



Lets got over what a path is.

The OS Module: Path

- `os.stat()` → Returns all information on a file.
- `os.remove()` → Removes a file at a given path.
- `os.path.isfile()` → Test whether a path is a regular file
- `os.path.getsize()` → Return the size of a file, reported by `os.stat()`.

The OS Module

The OS Module: Paths

- `os.path.split()` → Split a pathname. Returns tuple "(head, tail)" where "tail" is everything after the final slash. Either part may be empty.
- `os.path.exists()` → Test whether a path exists.
- `os.path.isfile()` → Test whether a path is a regular file
- `os.path.getsize()` → Return the size of a file, reported by `os.stat()`.
- `os.path.sep` → Gives you the separator character that is used on your system.

The OS Module: Paths

- `os.path.split()` → Split a pathname. Returns tuple "(head, tail)" where "tail" is everything after the final slash. Either part may be empty.
- `os.path.exists()` → Test whether a path exists.
- `os.path.isfile()` → Test whether a path is a regular file
- `os.path.getsize()` → Return the size of a file, reported by `os.stat()`.
- `os.path.sep` → Gives you the separator character that is used on your system.

Why do we need these?

Poll Question: Opening a File

How many times will this function call iterate if placed in a for loop:

`os.walk("dev_website")?`

```
1 dev_website
2 |---index.html
3 |___elements
4     |--- about.html
5     |--- cv.html
6     |___ projects.html
7
```

- ☐ A 1
- ☐ B 2
- ☐ C 6
- ☐ D 0

Reading Files

Poll Question: Opening a File

By default what does the `open()` function have it's file access permissions set as?

- ☐ A Reading
- ☐ B Writing
- ☐ C Appending
- ☐ D Reading and Writing
- ☐ E Writing and Appending
- ☐ F Reading, Writing, and Appending

Files

Which of the following reads all the contents of a file into a list of strings?

- ☐ A readlines
- ☐ B readall
- ☐ C read
- ☐ D readline

Poll Question: Read Characters

Given a variable named `file_object` that contains a file object which of the following will read the next 15 character into a variable named `title`.

- ☐ A `title = file_object.read(15)`
- ☐ B `title = file_object.read(14)`
- ☐ C `title = file_object.reads(15)`
- ☐ D `title = read(file_object, 15)`

Reading from Files

```
1 file_object = open('filename')
2 lines = file_object.readlines()
3 for line in lines:
4     print(line)
5 file_object.close()
```


Write Files

Poll Question: Files

Continue writing to existing files without overwriting the contents of the original file?

- ☐ A `outf = open('filename', 'r')`
- ☐ B `outf = open('filename', 'x')`
- ☐ C `outf = open('filename', 'a')`
- ☐ D `outf = open('filename', 'w')`
- ☐ E `outf = open('filename', 'w+')`

Poll Question: Files

Read from a file that may not exist?

- ☐ A `outf = open('filename', 'r+w')`
- ☐ B `outf = open('filename', 'rw')`
- ☐ C `outf = open('filename', 'r')`
- ☐ D `outf = open('filename', 'r+')`
- ☐ E `outf = open('filename', 'w+')`

Writing to Files

```
1 file_object = open('filename', 'w')
2 file_object.write('thing to write')
3 file_object.close() #automatic at program end
4 file_object.flush() #optional
```

Patterns Continued

Finding (single thing) in a Collection

```
1 def find_thing(collection):  
2     for thing in collection:  
3         if <thing meets condition>:  
4             return thing
```

```
5 def find_thing(collection):  
6     found = None  
7     for thing in collection:  
8         if <thing meets condition>:  
9             found = thing  
10            break  
11    return found
```

Filtering a collection

```
12 def filter(collection):  
13     new_list = []  
14  
15     for thing in collection:  
16         if <thing meets criteria>:  
17             newlist.append(thing)  
18  
19     return new_list
```

Patterns and Files

Usual Sum/Total:

```
1 def foo(some_list):  
2     total = 0  
3     for item in  
4         some_list  
5         total += item  
6     return total
```

Sum/Total Pattern w/ File:

```
1 def foo(filename):  
2     file_object = open(filename)  
3     lines = file_object.readlines()  
4     total = 0  
5     for line in lines:  
6         total += int(line)  
7     return total
```