15-110: Principles of Computing, Spring 2018

Lab 12 – Thursday, April 26

Goals

In this lab, you will experiment with reading from and writing to simple text files.

When you are done you should be able to do the following:

1. Open a file for reading or writing.
2. Read from a text file to initialize a list of strings.
3. Write to a text file to store a list of strings, one per line.
4. Read from a text file to initialize a list of integers.
5. Write to a text file to store a list of integers, one per line.

Part 1: TA Demonstrations (Quick review of files)

Open up the Primer on Files available on the course website. Remember that all data read from files are strings, and all data written to files must be strings. Review how to open a file for reading or writing. Review that when reading from files, there is an extra newline that must be removed.

If the string read in from a file is called `text`, then to remove the newline, we would use `text[0:-1]`. `text[0:-1]` is equivalent to saying `text[0:len(text)-1]`. Make sure you understand why the range is what it is.

If the string read in from a file needs to be treated as an integer `x`, we use the `int()` function: `x = int(text[0:-1])` to convert the string without a newline to an integer. To take an integer `x` and convert it to a string to write into a file, we use the `str()` function: `str(x)+'\n'` to convert the integer to a string and then append a newline for writing into the file.

Most loops for processing files require a `for` loop with an iterator.

For example:

```
inputfile = open("months.txt", 'r')
monthlist = []
for line in inputfile:
    monthlist.append(line[0:-1])
```

(For each `line` in the `inputfile`, append the line without the newline character on to `monthlist`.)

Activities

Create a `lab12` folder for this lab. Download the data (.txt) files from Autolab into your `lab12` folder. You will use them in the following activities.

1. Write a function `read_buildings(filename)` in the file `cmu.py` that opens up the given filename. It is assumed that the file contains the names of buildings on the CMU campus in Pittsburgh. Your function should append all of the buildings to a single list and return the list as the final result. Your buildings should not have extra newlines.

Sample usage:

```
python3 –i cmu.py
>>> cmu_bldgs = read_buildings("buildings.txt")
>>> cmu_bldgs
['Baker - Porter Hall', 'Cohon University Center', 'Collaborative
Innovation Center', 'College of Fine Arts', 'Cyert Hall', 'Doherty Hall',
'Elliott Dunlap Smith Hall', 'Gates / Hillman Centers', 'Hamburg Hall',
'Hamerschlag Hall', 'Hunt Library', 'Margaret Morrison Carnegie Hall',
'Margaret Morrison 5121 (Solar)', 'Mellon Institute', 'National Robotics
Engr Institute', 'Newell Simon Hall', 'Posner Center', 'Purnell Center
for the Arts', 'Roberts Engineering Hall', 'Scaife Hall', 'Scott Hall',
'Skibo Gymnasium', 'Software Engineering Institute', 'Warner Hall',
'Wean Hall', 'Whitfield Hall']
```

(What happens when you read a text file that has blank lines in it? Create a simple text file with this condition and run the function to see what your list looks like.)

2. Write a function `write_buildings(filename, bldg_list)` in `cmu.py` that takes a list of building names and a file name to write to, and writes the names of the buildings, one per line, into the given file in reverse order without changing the list of buildings. Your function should return None.

Sample usage:

```
python3 –i cmu.py
>>> cmu_bldgs = read_buildings("buildings.txt")
>>> write_buildings("buildings_reversed.txt", cmu_bldgs)
>>>
```

File contents of `buildings_reversed.txt`:

```
Whitfield Hall
Wean Hall
Warner Hall
...
Baker - Porter Hall
```

3. Write a function `max_in_file(filename)` in the file `maxnumber.py` that opens up the file with the given name. The file is assumed to contain a list of positive integers, one per line. The function should return the maximum integer stored in the file. You may assume the file has at least one integer in it.

Sample usage:

```
python3 -i maxnumber.py
>>> max_in_file("numbers.txt")
42
>>>
```

(Optional: Modify your function so that it can handle multiple integers on one line separated by commas. The file `numbers_v2.txt` has a set of data you can use to test this function.)

4. Write a function `factorial(filename)` in the file `factorial.py` that opens up the file given by the name and writes the first 50 factorial values: 1!, 2!, 3!, …, 50!, one per line. You should write the actual values, not the explanation points. HINT: Set up a variable to hold the factorial value (initially set to 1, why not 0?). Then set up a loop with a counter i that goes from 1 to 50. For each iteration, multiply the factorial value by i to get the next value to write to the file.

Sample usage:

```
python3 –i factorial.py
>>> factorial("factorial_numbers.txt")
>>>
```

File contents of `factorial_numbers.txt`:

```
1
2
6
24
120
720
...
30414093201713378043612608166064768844377641568960512000000000000
```

Submission

When you finish the lab, you should be inside the `lab12` folder, which is inside the `private/15110` directory. When you type `ls` and press the Enter key, you should see one or more of the following files: `cmu.py`, `maxnumber.py`, and `factorial.py` along with the text files. Type `cd ..` to move up one folder and press the Enter key. Then, zip your `lab12` folder by typing `zip –r lab12.zip lab12` and you should see a `lab12.zip` file in the current folder if you type `ls`. Please submit the zipped file `lab12.zip` on Autolab under 'lab 12'.