COMP10001 Foundations of Computing
Semester 1, 2021

Tutorial Questions: Week 10

— VERSION: 1475, DATE: MAY 10, 2021 —

Discussion

1. What is a "library"? How do we access them?
2. What is a "`defaultdict`"? How do we initialise and use it?
3. What is a "list comprehension"? How do we write one and how do they make our code simpler?
4. What is an "iterator"? What are some helpful methods in the `itertools` library?

   Now try Exercises 1–3
5. Why do we use "files"? Could we use computers without them?
6. What are the steps to reading and writing files?
7. What is a "csv" file and why is it useful for storing and manipulating data?

   Now try Exercises 4 & 5

Exercises

1. Rewrite the following with a default dictionary

```
my_dict = {}
for i in range(20):
    if i % 3 in my_dict:
        my_dict[i % 3].append(i)
    else:
        my_dict[i % 3] = [i]
```

2. Evaluate the following list comprehensions. For each one, also write some python code to generate the same list without using a comprehension.

   (a) `[(name, 0) for name in ("evelyn", "alex", "sam")]`

   (b) `[i**2 for i in range(5) if i % 2 == 1]`

   (c) `"".join([letter.upper() for letter in "python"])`

3. What output does the following code print?

```
import itertools
beatboxer = itertools.cycle(['boots', 'and', 'cats', 'and'])

for count in range(39):
    print(next(beatboxer))
```

4. Fill in the blanks in the program below which reads from `in.txt` and writes to `out.txt`.

```
outfile = [   1   ]("out.txt", "w")
with open("in.txt", [   2   ]) as infile:
    line_no = 1
    for line in [   3   ]:
        outfile.[   4   ](f"line:␣{line_no},␣length:␣{len(line)}\n")
        line_no += 1
outfile.write("The␣End")
[   5   ]
```

5. "travel.csv" is a csv file containing data on how people get to work in different cities in Australia. "process.py" is a python program which processes this data. What information does the "process.py" attempt to find and print? How could we edit it to find different statistics?

`travel.csv`

```
City,Train,Tram,Bus,Ferry,Car,Total
Melbourne,242969,55169,31937,783,1282997,1613855
Sydney,368572,3210,138340,9007,1206350,1725482
Adelaide,13715,4137,33673,211,390360,442102
Brisbane,62069,229,58228,3761,663353,787650
Perth,56417,223,37899,373,594571,689489
```

Data Source: Census of Population and Housing, 2016, TableBuilder

`process.py`

```
import csv

fp = open("travel.csv")
city = ''
curr_max = 0.0
for row in csv.DictReader(fp):
    ferry = int(row["Ferry"])
    total = int(row["Total"])
    if ferry / total > curr_max:
        city = row["City"]
        curr_max = ferry / total
print(city)
```

Problems

1. Let's head back to worksheet 8 and solve "For a While ..." again! **Using a list comprehension**, rewrite the function `allnum` that takes a list of strings, and returns a list of those that exclusively contain digits. `allnum(['3', '-4', '5', '3.1416', '0xfff', 'blerg!'])` should return `['3', '5']`.

2. Write a function which takes a string as input and returns a sorted list of the words which occur only once in the string. Try using a defaultdict and list comprehensions in your solution.

3. Write a function which takes two strings as input and uses an `itertools` iterator to find whether the first word is an anagram of the second word. This might not be a very efficient way to find an anagram but it will help us work with iterators!