Week 07 Laboratory Sample Solutions

Objectives

- Getting used to Python programing.
- Practice text processing in Python.
- Practice using regex in Python.

Preparation

Before the lab you should re-read the relevant lecture slides and their accompanying examples.

Getting Started

Set up for the lab by creating a new directory called `lab07` and changing to this directory.

```
$ mkdir lab07
$ cd lab07
```

There are some provided files for this lab which you can fetch with this command:

```
$ 2041 fetch lab07
```

If you're not working at CSE, you can download the provided files as a zip file or a tar file.

EXERCISE:

How many Orcas in that File?

A citizen science project monitoring whale populations has files of containing large numbers of whale observations. Each line in these files contains:

- the date the observation was made
- the number of whales in the pod ("pod" is the collective number for a group of whales)
- the species of whale

```
$ head whales0.txt
18/01/18  9 Pygmy right whale
01/09/17 21 Southern right whale
16/02/18  4 Striped dolphin
02/07/17  4 Common dolphin
19/02/18  4 Blue whale
21/02/18 38 Dwarf sperm whale
14/06/17 29 Southern right whale
20/06/17  3 Spinner dolphin
22/07/17 34 Indo-Pacific humpbacked dolphin
20/03/18  7 Long-finned pilot whale
```

Write a Python program **orca.py** which prints a count of the number of observations of Orcas in the files.

Your program should take 1 or more filenames as command line arguments.

```
$ ./orca.py whales0.txt
1245 Orcas reported
$ ./orca.py whales0.txt whales1.txt whales2.txt
3150 Orcas reported
```

Note: each line records one pod of whales
Your program should print the number of individual Orca observed - it needs to sum each pod of Orcas.

> NOTE:
>
> Your answer must be Python only. You can not use other languages such as Shell, Perl or C.
>
> Make the first line of your script `#!/usr/bin/env python3`
>
> You may not run external programs.

When you think your program is working, you can use `autotest` to run some simple automated tests:

```
$ 2041 autotest orca
```

When you are finished working on this exercise, you must submit your work by running `give` :

```
$ give cs2041 lab07_orca orca.py
```

before **Monday 17 July 12:00 (midday)** (2023-07-17 12:00:00) to obtain the marks for this lab exercise.

SOLUTION:

Sample solution for `orca.py`

```
#! /usr/bin/env python3

# Count Orca observations in 1 or more files
# d.brotherston@unsw.edu.au for COMP2041/9044

import sys

n_orcas = 0
for file in sys.argv[1:]:
    try:
        with open(file, encoding="utf-8") as f:
            for line in f:
                try:
                    # remove any leading & trailing white-space
                    line = line.strip()
                    date, count, species = line.split(maxsplit=2)
                    if species == "Orca":
                        n_orcas += int(count)
                except ValueError:
                    pass
    except OSError as e:
        print(
            f"{sys.argv[0]}: error: couldn't open {e.filename}: {e.strerror}",
            file=sys.stderr,
        )
        sys.exit(1)

print(n_orcas, "Orcas reported")
```

SOLUTION:

Alternative solution for `orca.py`

```python
#! /usr/bin/env python3

# Count Orca observations in 1 or more files
# d.brotherston@unsw.edu.au for COMP2041/9044

import fileinput

n_orcas = 0
for line in fileinput.input():
    try:
        # remove newline & any leading & trailing white-space
        line = line.strip()
        date, count, species = line.split(maxsplit=2)
        if species.casefold() == "Orca".casefold():
            n_orcas += int(count)
    except ValueError:
        pass

print(n_orcas, "Orcas reported")
```

EXERCISE:

Printing A summary of All Whale Species

Write a Python program **whale_summary.py** which take one or more files on the command line.

whale_summary.py should print for, every whale species in the file, how many pods were seen and how many total whales were seen of that species.

Whale species should be listed in alphabetic order.

All whale names should be converted to lower case.

All whale names should be converted to singular form - assume this can be done safely by deleting a trailing 's' if it is present.

```
$ ./whale_summary.py whales0.txt
blue whale observations: 51 pods, 1171 individuals
bryde's whale observations: 47 pods, 1023 individuals
coastal bottlenose dolphin observations: 53 pods, 1169 individuals
common dolphin observations: 62 pods, 1232 individuals
dwarf minke whale observations: 49 pods, 1080 individuals
dwarf sperm whale observations: 58 pods, 1272 individuals
fin whale observations: 65 pods, 1251 individuals
humpback whale observations: 55 pods, 1071 individuals
indo-pacific humpbacked dolphin observations: 43 pods, 897 individuals
long-finned pilot whale observations: 50 pods, 996 individuals
orca observations: 53 pods, 1245 individuals
pygmy right whale observations: 59 pods, 1260 individuals
pygmy sperm whale observations: 57 pods, 1346 individuals
sei whale observations: 55 pods, 929 individuals
short-finned pilot whale observations: 68 pods, 1344 individuals
southern right whale observations: 55 pods, 1252 individuals
spinner dolphin observations: 52 pods, 1118 individuals
striped dolphin observations: 68 pods, 1337 individuals
$ ./whale_summary.py messy_whales0.txt
blue whale observations: 2 pods, 27 individuals
dwarf minke whale observations: 6 pods, 132 individuals
orca observations: 2 pods, 48 individuals
```

NOTE:

Your answer must be Python only. You can not use other languages such as Shell, Perl or C.

Make the first line of your script `#!/usr/bin/env python3`

You may not run external programs.

When you think your program is working, you can use `autotest` to run some simple automated tests:

```
$ 2041 autotest whale_summary
```

When you are finished working on this exercise, you must submit your work by running `give`:

```
$ give cs2041 lab07_whale_summary whale_summary.py
```

before **Monday 17 July 12:00 (midday)** (2023-07-17 12:00:00) to obtain the marks for this lab exercise.

SOLUTION:

Sample solution for `whale_summary.py`

```python
#! /usr/bin/env python3

# Count Orca observations in 1 or more files
# d.brotherston@unsw.edu.au for COMP2041/9044
import re
import sys

whales = {}

for file in sys.argv[1:]:
    try:
        with open(file, encoding="utf-8") as f:
            for line in f:
                try:
                    date, count, species = line.split(maxsplit=2)
                    # convert to lower case
                    species = species.lower()
                    # remove leading & trailing white space
                    species = species.strip()
                    # convert multiple space to a single space
                    species = re.sub(r"\s+", " ", species)
                    # remove trailing 's'
                    if species[-1] == "s":
                        species = species[:-1]
                    # if this is the first time we have seen this species, add it to the dictionary
                    if species not in whales:
                        whales[species] = [0, 0]
                    whales[species][0] += 1
                    whales[species][1] += int(count)
                except ValueError:
                    print(f"{sys.argv[0]}: error reading line: {line}")
    except OSError as e:
        print(
            f"{sys.argv[0]}: error: couldn't open {e.filename}: {e.strerror}",
            file=sys.stderr,
        )
        sys.exit(1)

for species, (n_pods, n_individuals) in sorted(whales.items()):
    print(f"{species} observations: {n_pods} pods, {n_individuals} individuals")
```

SOLUTION:

Alternative solution for `whale_summary.py`

```python
#! /usr/bin/env python3

# Count Orca observations in 1 or more files
# d.brotherston@unsw.edu.au for COMP2041/9044
import re
import sys
import fileinput
from collections import defaultdict

whales = defaultdict(lambda: [0, 0])
for line in fileinput.input():
    try:
        date, count, species = map(lambda x: x.lower().strip(), line.split(maxsplit=2))
        # convert multiple space to a single space
        species = re.sub(r"\s+", " ", species)
        # remove trailing 's'
        if species[-1] == "s":
            species = species[:-1]
        whales[species][0] += 1
        whales[species][1] += int(count)
    except ValueError:
        print(f"{sys.argv[0]}: error reading line: {line}")

for species, (n_pods, n_individuals) in sorted(whales.items()):
    print(f"{species} observations: {n_pods} pods, {n_individuals} individuals")
```

EXERCISE:

Find the sum of all numbers in a file

Write a Python program **summing_numbers.py** which given a file as a command line argument, prints the sum of all the numbers in the file.

A *number* for the purposes of this exercise is any sequence of consecutive digits.

Note only the digits 0-9 are considered parts of numbers. The characters **'.'** and **'-'** are not considered part of numbers.

For example, the string the **"It was -42.5C outside"** should be treated as containing the numbers **42** and **5**.

```
$ ./summing_numbers.py A-Tale-of-Two-Cities.txt
7104
$ ./summing_numbers.py Pride-and-Prejudice.txt
46520
```

NOTE:

Your answer must be Python only. You can not use other languages such as Shell, Perl or C.

Make the first line of your script `#!/usr/bin/env python3`

You may not run external programs.

When you think your program is working, you can use `autotest` to run some simple automated tests:

```
$ 2041 autotest summing_numbers
```

When you are finished working on this exercise, you must submit your work by running `give` :

```
$ give cs2041 lab07_summing_numbers summing_numbers.py
```

before **Monday 17 July 12:00 (midday)** (2023-07-17 12:00:00) to obtain the marks for this lab exercise.

SOLUTION:

Sample solution for `summing_numbers.py`

```python
#! /usr/bin/env python3

import fileinput
import re

total = 0
for line in fileinput.input():
    for number in re.findall(r"\d+", line):
        total += int(number)

print(total)
```

CHALLENGE EXERCISE:

A Python Program that Prints Python

Write a Python program **python_print.py** which is given a single argument. It should output a Python program which when run, prints this string. For example:

```
$ ./python_print.py "hello world, how are you?" | python3
hello world, how are you?
$ ./python_print.py "I'm So Meta, Even This Acronym" > new_program.py
$ chmod +x new_program.py
$ ./new_program.py
I'm So Meta, Even This Acronym
```

NOTE:

You can assume the string contains only ASCII characters.

You can not make other assumptions about the characters in the string.

Your answer must be Python only. You can not use other languages such as Shell, Perl or C.

Make the first line of your script `#!/usr/bin/env python3`

You may not run external programs.

When you think your program is working, you can use `autotest` to run some simple automated tests:

```
$ 2041 autotest python_print
```

When you are finished working on this exercise, you must submit your work by running `give` :

```
$ give cs2041 lab07_python_print python_print.py
```

before **Monday 17 July 12:00 (midday)** (2023-07-17 12:00:00) to obtain the marks for this lab exercise.

SOLUTION:

Sample solution for `python_print.py`

```python
#!/usr/bin/env python3

# Output a Python program which when run will print the
# string supplied as a commandline argument
# written by d.brotherston@unsw.edu.au for COMP2041/9044

import sys

print(
    f"""\
#!/usr/bin/env python3

print({repr(sys.argv[1])})
"""
)
```

CHALLENGE EXERCISE:

A Python Program that Prints Python that Prints Python that ...

Write a Python program `python_print_n.py` which is given a two arguments, an integer **n** and a string.

If **n** is 1 it should output a Python program which prints the string.

If **n** is 2 it should output a Python program which prints a Python program which prints the string.

If **n** is 3 it should output a Python program which prints a Python program which prints a Python program which prints the string.

And so on for any value of **n**.

```
$ ./python_print_n.py 1 "hello world, how are you?" | python3
hello world, how are you?
$ ./python_print_n.py 2 "hello world, how are you?" | python3 | python3
hello world, how are you?
$ ./python_print_n.py 3 "hello world, how are you?" | python3 | python3 | python3
hello world, how are you?
$ ./python_print_n.py 10 "Now that's a lot of python" | python3 | python3 | python3 | python3 | python3 |
python3 | python3 | python3 | python3 | python3
Now that's a lot of python
```

NOTE:

You can assume n is a positive integer.

You can assume the string contains only ASCII characters.

You can not make other assumptions about the characters in the string.

Your answer must be Python only. You can not use other languages such as Shell, Perl or C.

Make the first line of your script `#!/usr/bin/env python3`

You may not run external programs.

When you think your program is working, you can use `autotest` to run some simple automated tests:

```
$ 2041 autotest python_print_n
```

When you are finished working on this exercise, you must submit your work by running `give` :

```
$ give cs2041 lab07_python_print_n python_print_n.py
```

before **Monday 17 July 12:00 (midday)** (2023-07-17 12:00:00) to obtain the marks for this lab exercise.

SOLUTION:

Sample solution for `python_print_n.py`

```python
#!/usr/bin/env python3

# Output a Python program which when run will print the
# string supplied as a commandline argument
# written by d.brotherston@unsw.edu.au for COMP2041/9044

import sys
import textwrap

program = sys.argv[2]

for _ in range(int(sys.argv[1])):
    program = textwrap.dedent(
        f"""
        #! /usr/bin/env python3
        print({repr(program)})
    """
    ).strip()

print(program)
```

CHALLENGE EXERCISE:

A Python Program that Prints itself forever

Write a Python program `python_print_inf.py` which prints it's own soruce code.

The program **can not** use the `open` function.
The program **can not** use the `inspect` module.

```
$ ./python_print_inf.py > new_program.py
$ diff -s python_print_inf.py new_program.py
Files python_print_inf.py and new_program.py are identical
$ chmod +x new_program.py
$ ./new_program.py > second_order_program.py
$ diff -s python_print_inf.py second_order_program.py
Files python_print_inf.py and second_order_program.py are identical
$ ./python_print_inf.py | python3 | python3 | python3 | python3 | python3 > nth_program.py
$ diff -s python_print_inf.py nth_program.py
Files python_print_inf.py and nth_program.py are identical
```

NOTE:

Your answer must be Python only. You can not use other languages such as Shell, Perl or C.

Make the first line of your script `#!/usr/bin/env python3`

You may not run external programs.

When you think your program is working, you can use `autotest` to run some simple automated tests:

```
$ 2041 autotest python_print_inf
```

When you are finished working on this exercise, you must submit your work by running `give` :

```
$ give cs2041 lab07_python_print_inf python_print_inf.py
```

before **Monday 17 July 12:00 (midday)** (2023-07-17 12:00:00) to obtain the marks for this lab exercise.

SOLUTION:

Sample solution for `python_print_inf.py`

```python
#!/usr/bin/env python3
var = 'print("#!/usr/bin/env python3\\nvar = " + repr(var) + "\\neval(var)")'
eval(var)
```

SOLUTION:

Alternative solution for `python_print_inf.py`

```
s='s=%r;print(s%%s)';print(s%s)
```

Submission

When you are finished each exercises make sure you submit your work by running `give .`

You can run `give` multiple times. Only your last submission will be marked.

Don't submit any exercises you haven't attempted.

If you are working at home, you may find it more convenient to upload your work via give's web interface.

Remember you have until **Week 8 Monday 12:00:00 (midday)** to submit your work.

You cannot obtain marks by e-mailing your code to tutors or lecturers.

You check the files you have submitted here.

Automarking will be run by the lecturer several days after the submission deadline, using test cases different to those `autotest` runs for you. (Hint: do your own testing as well as running `autotest .`)

After automarking is run by the lecturer you can view your results here. The resulting mark will also be available via give's web interface.

Lab Marks

When all components of a lab are automarked you should be able to view the the marks via give's web interface or by running this command on a CSE machine:

```
$ 2041 classrun -sturec
```
