CSE 140 Section 02 Problems

For this section, you will be writing logic that can analyze a set of information about people's names, ages, and weights. You are not expected to get through all of these problems, but the problems that you do not complete during section make good practice problems outside of section.

1. We can start with a file, data.txt, that contains the following:

```
20
17
21
```

   Write a small program to read in this file and compute the average of these values. Use a for loop rather than the readline() function, in order to prepare your solution to be robust enough to handle larger files. Make sure to properly convert your data into int or float where necessary.

2. Consider a slightly different file, data.txt, that contains data for various people's names, ages, and weights. Here is one example if what might be contained in the data.txt.

```
Alice
20
120.0
Bob
17
130.5
Freddie
21
190.6
```

   Write a small program to read in this file and compute the average age. You will use a similar strategy as you did for the previous problem, but you will need logic to only selectively target the age data within the file.

3. Modify your second program, so that it prints whether a given age is younger, older, or equal to the average age. Your program will have a structure like this:

```
age = 42 # This could be any integer >= 0

# Your code from Problem 2

# Your code for Problem 3
```

4. Write a program that finds and prints the maximum weight within the data set. You can assume that all weights are positive.

5. A more common and useful format is csv -- comma separate values. This is how our data set would read if it were formatted as csv:

```
Alice,20,120.0
Bob,17,130.5
Freddie,21,190.6
```

Write a program that reads in data.txt and outputs the csv formatted version to a file called data.csv. You do not need to convert the data into ints or floats, but you will need to remove extra newlines from each line. You can use rstrip to do this. For example, if you have the following string:

```
example_string = "A string with a newline at the end\n"
```

Then you can remove it

```
stripped_string = example.rstrip()
```

And the contents of stripped_example would be:

```
"A string with a newline at the end"
```

6. Write a program that reports whether a particular person is younger, older, or equal to the average age. You may assume that the given name does exist inside of the data set. Your program will have a structure like this:

```
name = "Alice" # This could be any string

# Your code for Problem 5
```

CSE 140 Section 02 Code Examples

1. An example of opening a file and printing all of its contents:

```
input = open("input.txt")
for line in input:
    print line,

input.close()
```

The comma at the end of the print statement stops python from printing a redundant \n in addition to the one that is already in the line variable.

2. An example of opening a file and counting the amount of lines it contains:

```
input = open("input.txt")
line_number = 0
for line in input:
    line_number = line_number + 1

input.close()

print line_number
```

3. An example of opening a file and only printing the odd numbered lines (where the first line is 1):

```
input = open("input.txt")
line_number = 0
for line in input:
    if line_number % 2 == 0:
        print line,
    line_number = line_number + 1

input.close()
```

4. An example of opening a file and writing to it:

```
output_file = open("output.txt", "w")

output_file.write("apple\n")
output_file.write("banana\n")
output_file.write("nyan\n")

output_file.close()
```

5. An example of combining data from separate lines:

```
input = open("input.txt")
line_number = 0
for line in input:
  if line_number % 3 == 0:
      name = line.rstrip() # rstrip removes the extra \n
```

```
            elif line_number % 3 == 1:
                age = int(line)
                print "%s: %d" % (name, age)
            line_number = line_number + 1

        input.close()
```

For the file, input.txt

```
Alice
20
120.0
Bob
17
130.5
Freddie
21
190.6
```

This program prints the output

```
Alice: 20
Bob: 17
Freddie: 21
```

6. An example of looking for data matching a specific condition:

```
name = "Alice"
person_found = False

input = open("input.txt")
line_number = 0
for line in input:
    if line_number % 3 == 0:
        if line.rstrip() == name:
            person_found = True
    elif line_number % 3 == 1:
        if person_found:
            age = int(line)
            person_found = False
    line_number = line_number + 1

input.close()

print "%s: %d" % (name, age)
```

7. An example of String formatting:

```
    print "%s %d %f" % ("Example String", 42, 9001.0)
```

CSE 140 Section 02 Solutions

1. One possible solution appears below:

```
age_sum = 0

input = open("data.txt")
line_number = 0
for line in input:
    age_sum = age_sum + int(line)
    line_number = line_number + 1

input.close()
print float(age_sum) / line_number
```

2. One possible solution appears below:

```
age_sum = 0

input = open("data.txt")
line_number = 0
for line in input:
    if line_number % 3 == 1:
        age_sum = age_sum + int(line)
    line_number = line_number + 1

input.close()
print float(age_sum) / (line_number / 3)
```

3. One possible solution appears below:

```
age = 42
age_sum = 0

input = open("data.txt")
line_number = 0
for line in input:
    if line_number % 3 == 1:
        age_sum = age_sum + int(line)
    line_number = line_number + 1

input.close()
average_age = float(age_sum) / (line_number / 3)
print average_age

if age == average_age:
    print "age equals the average age"
elif age > average_age:
    print "age is older than the average age"
else: # age < average age:
    print "age is younger than the average age"
```

4. One possible solution appears below:

```
max_weight = 0
```

```
input = open("data.txt")
line_number = 0
for line in input:
    if line_number % 3 == 2:
        if float(line) > max_weight:
            max_weight = float(line)
    line_number = line_number + 1

input.close()
print max_weight
```

5. One possible solution appears below:

```
input = open("data.txt")
line_number = 0
for line in input:
    if line_number % 3 == 0:
        name = line.rstrip()
    if line_number % 3 == 1:
        age = line.rstrip()
    if line_number % 3 == 2:
        weight = line.rstrip()
        print "%s,%s,%s" % (name, age, weight)
    line_number = line_number + 1

input.close()
```

6. One possible solution appears below:

```
name = "Alice"

name_found = False
age_sum = 0

input = open("data.txt")
line_number = 0
for line in input:
    if line_number % 3 == 0:
        if name == line.rstrip():
            name_found = True
    if line_number % 3 == 1:
        age_sum = age_sum + int(line)
        if name_found:
            age = line.rstrip()
    if line_number % 3 == 2:
        if name_found:
            weight = line.rstrip()
    line_number = line_number + 1

input.close()
average_age = float(age_sum) / (line_number / 3)

if age == average_age:
    print "%s's age equals the average age" % name
```

```
elif age > average_age:
    print "%s is older than the average age" % name
else: # age < average age:
    print "%s is younger than the average age" % name
```