Lab 7 – PART 1

Topics Covered:

- Files, testing input and output

Pre-Lab Questions

Question 1: Which of the following will correctly open the file text.txt for reading?

a) text = open("text.txt", "r")
b) text = open("text.txt")
c) text = open("text.txt", "read")
d) Both a) and b)

Question 2: Which of the following will correctly open the file mytext.txt for writing?

a) text = write("mytext.txt", "w")
b) text = write("mytext.txt")
c) text = open("mytext.txt", "w")
d) text = open("mytext.txt")

Question 3: Take a look at the following code segment and text file. What will be the output of the following code segment?

Code Segment	text.txt

`text = open("text.txt", "r")`
`for line in text:`
`  print(line)`
`text.close()`
	`cat`
	`dog`
	`monkey`

a) catdogmonkey

b) cat
dog
monkey

c) cat

dog

monkey

Question 4: Take a look at the following code segment and text file. What will be the output of the following code segment?

Code Segment	text.txt

`text = open("text.txt", "r")`
`for line in text:`
`  print(line, end="")`
`text.close()`
	`cat`
	`dog`
	`monkey`

a) catdogmonkey

b) cat
dog
monkey

c) cat

dog

monkey

Question 5: Take a look at the following code segment and text file. What will be the output of the following code segment?

Code Segment	text.txt

`text = open("text.txt", "r")`
`for line in text:`
`  print(line.rstrip())`
`text.close()`
	`cat`
	`dog`
	`monkey`

a) catdogmonkey

b) cat
dog
monkey

c) cat

dog

monkey

Question 6: Suppose a text file contains the lines:

Train

Car

Using the readline() function would return what string for the first call:

a) Train
b) Train\n
c) TrainCar
d) Train Car

Question 7: Suppose the text file data.txt contains the lines:

3,1,4

1,3,7

8,5,2

Determine the output if the following code is executed to work with the above data.txt file.

```
file = open("data.txt", "r")
for line in file:
    v = 0
    line = line.strip()
    nums = line.split(",")
    for n in nums:
        v += int(n)
    print(v, end=' ')
file.close()
```

a) 12 9 13
b) 8 11 15
c) 8, 5, 2
d) 3 1 4 1 3 7 8 5 2
e) It would produce an error

Lab 7 – PART 2

1. **Follow the instructions below to construct a complete program.**

   We are going to write a program that reads a textfile with keywords and their sentiment value, and calculates the sentiment of a tweet. The text file has some commonly used words and their sentiment values. If we group these keywords into categories, we can check any tweet by comparing the words in it against the different sentiment categories and their values and coming up with a sentiment value for the tweet. We will group the keywords from our text file that have a 20 sentiment value into a “positive” group, the ones with a value of 0 into a “neutral” group and the ones with a -10 sentiment value into a “negative group”. Note that if you copy and paste code from this document, the minus signs will likely cause issues with the code. It is better to type the minus signs rather than copy and paste them in to avoid these issues.

Sentiment Value	Sentiment Group
20	Positive
0	Neutral
-10	Negative

   These are the contents of our text file:
   love,20
   like,0
   hate,-10
   happy,20
   greatest,20
   hurt,-10
   alone,-10
   tired,0
   sad,-10
   regrets,-10

   We want to take the contents of the text file and place all the keywords with a 20 value into a positive list, the 0 into a neutral list, and the -10 into a negative list. Let's start by opening up the text file and creating our 3 lists:

```
f = open("text.txt","r")

positive = []
neutral = []
negative = []
```

Now, remember that every line of our text file is formatted like this:

love,20

Since we want to analyze the value and place the word into a list based on that value, it makes sense to split the line at the comma and place the 2 values into a list so we can analyze them individually.

```
for line in f:
    entries = line.split(",")
```

This will split the line where the comma is and place the 2 values into a list called entries. It will put the word at entries[0] and the sentiment value at entries[1].

Now that our word and sentiment value are separated we can use the sentiment value to determine which group the word belongs to and append it to that list.

```
if entries[1] == 20:
    positive.append(entries[0])
elif entries[1] == 0:
    neutral.append(entries[0])
else:
    negative.append(entries[0])
```

Let's print our lists and see if our code works:

```
print("The positive keywords are {}".format(positive))
print("The neutral keywords are {}".format(neutral))
print("The negative keywords are {}".format(negative))
```

Output:
The positive keywords are []
The neutral keywords are []
The negative keywords are ['love', 'like', 'hate', 'happy', 'greatest', 'hurt', 'alone', 'tired', 'sad', 'regrets']

Something is wrong, and none of our words are being categorized properly. Let's try to print the contents of the list entries to try to troubleshoot. We will add these print statements to the for loop:

```
print(entries[0])
print(entries[1])
```

The output looks like this:
love

20

like
0

hate
-10

…

Where we have a big space between the entries of every line. This means that there is a newline character attached to every number, and it is not being read as an integer. We must strip the newline character from the entries[1] before we try to sort the words to get the proper number. We also must convert entries[1] to an integer value, because reading it from the textfile will be a string by default. We will add this line after we split the line into a list:

```
entries = line.split(",")
entries[1] = int(entries[1].rstrip("\n"))
```

This line converts entries[1] into an integer, as well as does a right strip of the “\n” character. Let’s run our code again:

```
f = open("text.txt","r")
positive = []
neutral = []
negative = []
for line in f:
    entries = line.split(",")
    entries[1] = int(entries[1].rstrip("\n"))
    if entries[1] == 20:
      positive.append(entries[0])
    elif entries[1] == 0:
      neutral.append(entries[0])
    else:
      negative.append(entries[0])

print("The positive keywords are {}".format(positive))
print("The neutral keywords are {}".format(neutral))
print("The negative keywords are {}".format(negative))

f.close()
```

Output:
The positive keywords are ['love', 'happy', 'greatest']

The neutral keywords are ['like', 'tired']
The negative keywords are ['hate', 'hurt', 'alone', 'sad', 'regrets']

We see now that our tweets are being categorized properly and we have 3 lists with positive, neutral and negative keywords. Now we can start analyzing tweets. We will split up the tweet into individual words in a list and check every word to see if it is one of the positive or negative keywords. We can ignore the neutral keywords because they are valued at 0 points which won't change our total. If a word matches a word in our positive list, we will add 20 to a total sentiment value. If our word matches a word in our negative list, we will add -10 to the total sentiment value.

```
tweet = "I really am very happy for you I love the weather I am also sad and have some reg
rets about being so tired"
tweet_words = tweet.split()
sentiment = 0
for word in tweetWords:
  if word in positive:
    sentiment+=20
  elif word in negative:
    sentiment-=10
```

Finally, we can print our sentiment value, and have this as the final code:

```
f = open("text.txt","r")
positive = []
neutral = []
negative = []
for line in f:
   entries = line.split(",")
   entries[1] = int(entries[1].rstrip("\n"))
   if entries[1] == 20:
     positive.append(entries[0])
   elif entries[1] == 0:
     neutral.append(entries[0])
   else:
     negative.append(entries[0])

tweet = "I really am very happy for you I love the weather I am also sad and have some reg
rets about being so tired"
tweetWords = tweet.split()
sentiment = 0
for word in tweetWords:
  if word in positive:
    sentiment+=20
  elif word in negative:
    sentiment-=10
print("The positive keywords are {}".format(positive))
print("The neutral keywords are {}".format(neutral))
print("The negative keywords are {}".format(negative))
```

```
print("The sentiment of the tweet is {}".format(sentiment))

f.close()
```

Output:
The positive keywords are ['love', 'happy', 'greatest']
The negative keywords are ['hate', 'hurt', 'alone', 'sad', 'regrets']
The neutral keywords are ['like', 'tired']
The sentiment of the tweet is 20

We can also test out different tweets or change our program to make the user enter a string for the tweet, but for now we will leave it hard-coded.

2. Identify errors and fix the code.

Look at the following code segment. This reads from a file called text.txt which has only one line. The program should read this line and split it into individual words; these words should be written to the screen and to the file myfile.txt with each word on its own line. There are 4 errors in this code; find them and correct the code.

```
text = open("text.txt","r")
myfile = open("myfile.txt","r")
line = text.read
words = line.split()
for word in words:
    print(word + "\n")
    myfile.write(word)

# Close the files
text.close()
myfile.close()
```