Problem 1 (a)

Definitions

Prior to beginning our work, we load the requiste packages:

In [1]:
```
import math
```

We also load all 26 letters (*in lower case*) into an array for later use. This is done by importing a text file containing these letters

In [2]:
```
with open('../LowerCaseAlphabet.txt', 'r') as myFile:
    lowerAlpha = myFile.readlines()
```

and then stripping the return characters (\n) from each line

In [3]:
```
for i in range(len(lowerAlpha)):
    lowerAlpha[i] = lowerAlpha[i].replace('\n','')

lowerAlphaB = ''.join(str(x) for x in lowerAlpha)
```

Finally, we check to see if the alphabet imported properly,

In [4]:
```
print(lowerAlphaB)
```

```
abcdefghijklmnopqrstuvwxyz
```

as well as create an upper case version

In [5]:
```
upperAlpha = []
for x in lowerAlpha:
    upperAlpha.append(x.upper())

upperAlphaB = ''.join(str(x) for x in upperAlpha)
```

and check it

In [6]:
```
print(upperAlphaB)
```

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Load File

We begin our work by loading the text from the source file:

In [7]:
```
with open('../Text-Files/sawyer-ascii.txt', 'r') as myFile:
        tempData = myFile.readlines()
```

Then, we get some basic information about the data imported from the file

In [8]:
```
print(len(tempData))
```

```
8807
```

Now, convert the array of strings to a single string.

In [9]:
```
data = ''.join(str(x) for x in tempData)
```

Then find and store the length of the resulting string:

In [10]:
```
charCNT = len(data)
print(charCNT)
```

```
402665
```

Next, the length compare it to the combined length of all the strings in the initial array we got from importing the text file.

In [11]:
```
cnt = 0
for x in tempData:
    cnt = cnt + len(x)

print(cnt)
```

```
402665
```

Since the character counts are accurate, we can proceed.

Get Character List

First, we will obtain a list of all characters occuring in the text

In [12]:
```
myChars = list(set(data))

myChars2 = ''.join(str(x) for x in myChars)
print(myChars)
print(myChars2)
```

```
['?', 'P', 'O', '+', '#', 'y', 'p', 'h', ';', 'w', '2', "'", 'c', 'R', 'T', ')',
'.', '_', '(', 'o', 's', '/', '4', '>', '\n', ']', '-', 'E', 'L', 't', ':', 'B',
'0', ',', 'd', 'I', 'x', 'l', 'k', '6', '9', '5', '$', '1', 'i', 'F', 'V', '!',
'K', 'a', 'v', 'H', 'W', '&', 'N', '<', '"', 'Y', '8', '[', 'A', 'm', 'C', 'D',
'n', 'j', 'g', '~', 'U', 'u', '*', 'q', 'b', 'f', 'r', 'S', 'z', 'M', 'G', ' ',
'%', '3', '7', 'J', '@', 'e', 'Q', 'X']
?PO+#yph;w2'cRT)._(os/4>
]-ELt:B0,dIxlk695$1iFV!KavHW&N<"Y8[AmCDnjg~Uu*qbfrSzMG %37J@eQX
```

Next, we remove our non-alphabetic characters from the list of characters to eliminate

In [13]:
```
for x in lowerAlphaB:
    myChars2 = myChars2.replace(x,'')

for x in upperAlphaB:
    myChars2 = myChars2.replace(x,'')

print(len(myChars2))
print(myChars2)
```

```
37
?+#;2').(/4>
]-:0,695$1!&<"8[~* %37@
```

and check the results

In [14]:
```
print(list(set(myChars2)))
print(len(list(set(myChars2))))
```

```
['?', '+', '#', ':', '&', '0', ';', '2', "'", '<', ',', '"', '8', '[', ' ', '6',
')', '_', '.', '*', '(', '5', '%', '$', '3', '/', '7', '1', '@', '4', '>', '\n',
'!', '~', ']', '9', '-']
37
```

Clean the Text

Now, we can eliminate these characters from the text

In [15]:
```
#create copy of imported data
data2 = data

for x in myChars2:
    data2 = data2.replace(x,'')
```

and check to make sure the lengths have changed

In [16]:
```
print(len(data))
print(len(data2))
```

```
402665
307917
```

Last, we convert all letters in the text to lowercase

In [17]:
```
data2 = data2.lower()
```

Get Frequencies and Probabilities

Frequencies

To get the Frequencies of each letter, we first create an array to store them

In [18]:
```
counts = []
```

and then go though the alphabet counting

In [19]:
```
for x in lowerAlpha:
    counts.append(data2.count(x))

print(counts)
print(len(counts))
```

```
[24352, 5221, 6873, 15302, 37080, 6270, 6841, 19997, 19642, 692, 3138, 12565, 74
44, 20959, 24325, 4950, 194, 16092, 18376, 29970, 9340, 2474, 8244, 387, 7032, 1
57]
26
```

Now, we can compute the probabilities. First, we store the number of characters in the cleaned text

In [20]:
```
charTOT = len(data2)

print(charTOT)
```

```
307917
```

which we check against the frequencies we just calculated

In [21]:
```
print(sum(counts))
```

```
307917
```

Probabilities

Again, we first create and empty array to hold the probabilities

In [22]:
```
prbs = []
```

then we loop through the list of frequencies, using them to create each probability

In [23]:
```
for x in counts:
    prbs.append(x / charTOT)
```

this gives

In [24]:
```
print(prbs)
```

```
[0.07908624726793259, 0.016955867977409497, 0.022320950126170365, 0.049695210072
844304, 0.12042206178937831, 0.02036263018930426, 0.022217026016751268, 0.064942
8255016774, 0.0637899174128093, 0.002247358866187966, 0.01019105797991017, 0.040
80645108909219, 0.02417534595361737, 0.068067044041089, 0.07899856130061023, 0.0
1607576067576652, 0.0006300399133532738, 0.05226083652412825, 0.0596784198339162
8, 0.09733142372782276, 0.03033284943669885, 0.008034632709463915, 0.02677344868
909479, 0.0012568321982872007, 0.0228373230448465, 0.0005098776618374432]
```

Entropy Estimate

We can now estimate the entropy of the converted text (*all lower case, no special characters, spaces, tabs, or returns*). To do this, we first initialize a variable to hold our value for the entropy

In [25]:
```
entropTOT = 0
```

Then we loop through all the probabilities, computing the entropy for each and adding it to the total

In [26]:
```
for x in prbs:
    entropTOT = entropTOT - x * math.log2(x)
```

to get

In [27]:
```
print(entropTOT)
```

```
4.184820826080936
```