

## Mini Project 1

### **Application mp1a.py:**

Aim: To create a Python script that provides a word frequency histogram in the form of a dictionary with the word as key and the number of occurrences as the value and to provide filters to collect words based on first character into separate lists in order to find and display the ratio (as a percentage) of each starting character list size against the total number of words.

### Project Description:

- Read the contents of the file from the given URL into a list
- Omit the white spaces, special characters present in the text by using split and store it in list variable called 'lines'
- Convert all the read words into lower case and find the total number of occurrences of each word and save the count in the dictionary. Form of the dictionary could be word: number of occurrences
- Plot a histogram using the dictionary contents. The histogram is saved in a file called MP1\_Histogram.pdf in the main project root folder
- Compare the first characters of every key with its neighboring key and add the values if a match is found
- Calculate the total number of words in the read text and calculate the percentage ratio of every character's occurrences to the total number of words in the text. Display this for every character

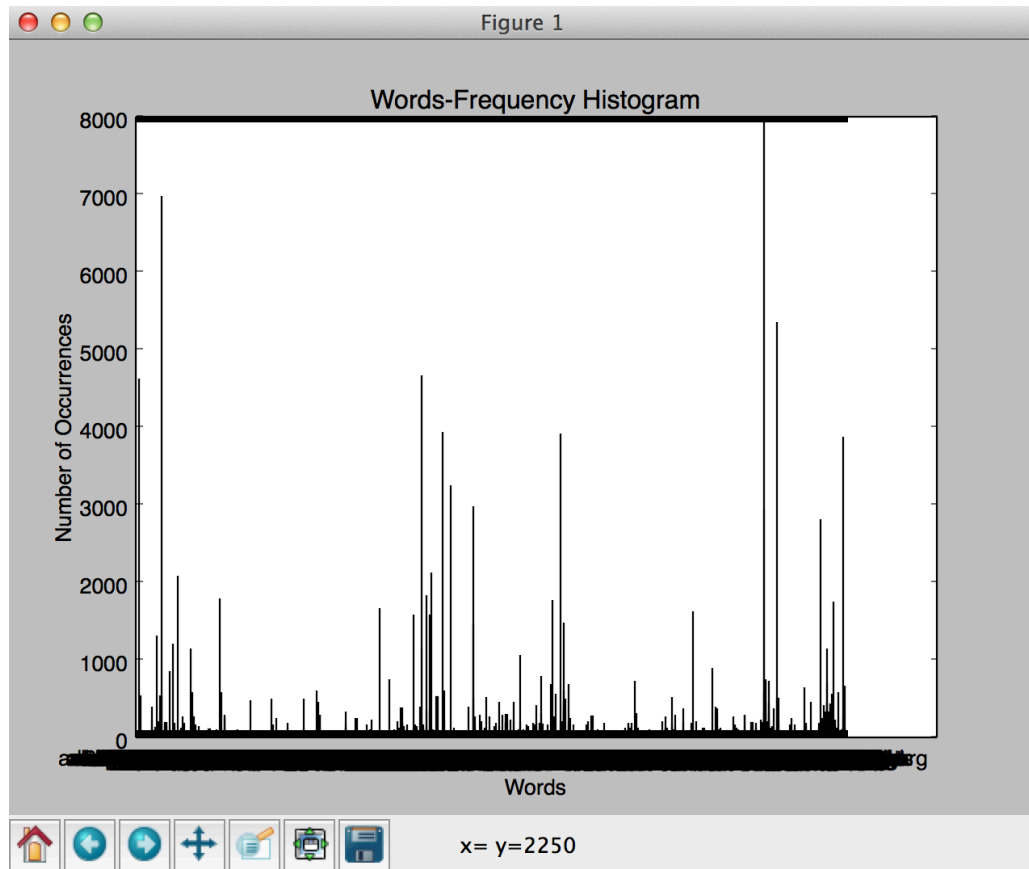
### Project Insight:

- Used Dictionary Elements for storing the Word as a key and the number of occurrences as value
- Used split() and isalnum() in order to get rid of the white spaces and the special characters
- Compare the first characters of every key with its neighboring key and add the values that is the number of occurrences if a match is found
- Calculated the Ratio by dividing the individual character's number of occurrences with the total number of words

### Expected Results:

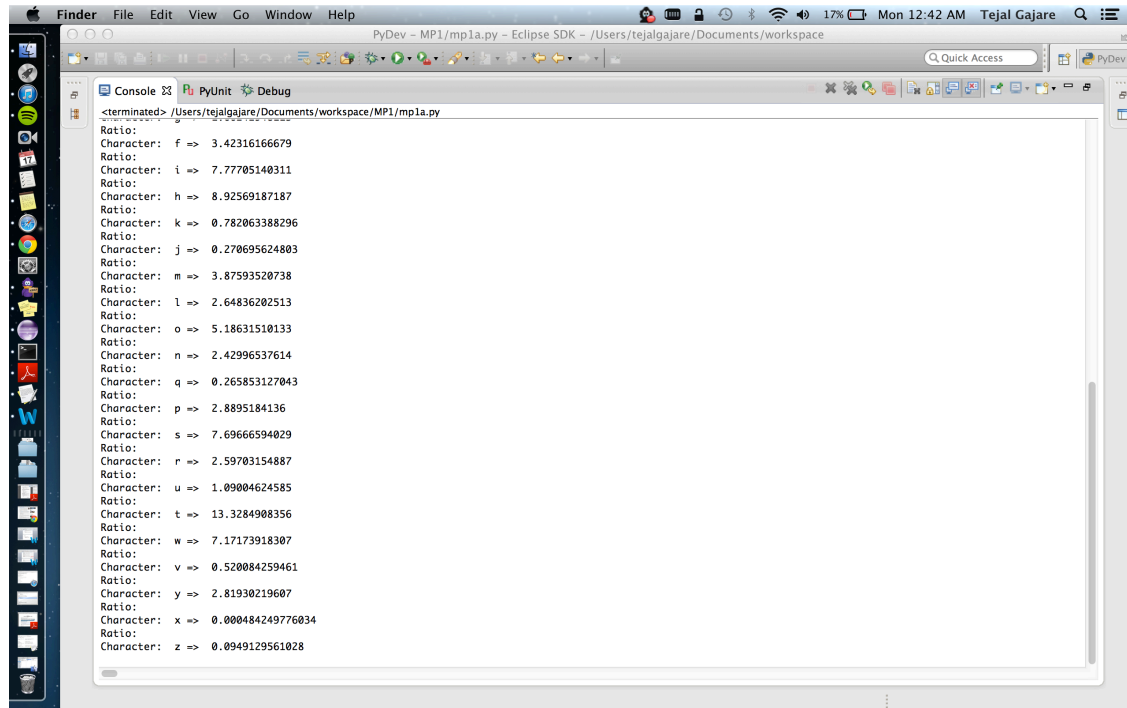
- The project will display the dictionary object with all words and their occurrences
- It will plot a histogram
- It will display all the ratios of the individual characters to the total number of words

Screen Captures:



Histogram

```
Finder File Edit View Go Window Help
PyDev - MPI/mp1a.py - Eclipse SDK - /Users/tejalgajare/Documents/workspace
Console PyUnit Debug
-terminated- /Users/tejalgajare/Documents/workspace/MPI/mp1a.py
Reading the contents from the given URL...
Omitting the white spaces...
Omitting the special Characters...
Sorted Dictionary in the form of => Words : Number of Occurrences:
OrderedDict([('1', 2), ('14', 1), ('15', 1), ('1500', 1), ('1849', 1), ('1859', 1), ('1861', 1), ('1864', 1), ('1880', 1), ('1a', 1), ('1b', 1), ('1c', 2), ('1c', 2), ('1d', 1), ('1e', 1), ('1f', 1), ('1g', 1), ('1h', 1), ('1i', 1), ('1j', 1), ('1k', 1), ('1l', 1), ('1m', 1), ('1n', 1), ('1o', 1), ('1p', 1), ('1q', 1), ('1r', 1), ('1s', 1), ('1t', 1), ('1u', 1), ('1v', 1), ('1w', 1), ('1x', 1), ('1y', 1), ('1z', 1), ('2', 1), ('3', 1), ('4', 1), ('5', 1), ('6', 1), ('7', 1), ('8', 1), ('9', 1), ('a', 25043), ('b', 8545), ('c', 7107), ('d', 6478), ('e', 3143), ('f', 7069), ('g', 3433), ('h', 18432), ('i', 18432), ('j', 18432), ('k', 18432), ('l', 18432), ('m', 18432), ('n', 18432), ('o', 18432), ('p', 18432), ('q', 18432), ('r', 18432), ('s', 18432), ('t', 18432), ('u', 18432), ('v', 18432), ('w', 18432), ('x', 18432), ('y', 18432), ('z', 18432), (' ', 206505)
List of individual characters along with their respective occurrences:
['1', 47, '2', 9, '3', 5, '4', 5, '5', 7, '6', 2, '8', 4, '9', 4, 'a', 25043, 'b', 8545, 'c', 7107, 'd', 6478, 'e', 3143, 'f', 7069, 'g', 3433, 'h', 18432, 'i', 18432, 'j', 18432, 'k', 18432, 'l', 18432, 'm', 18432, 'n', 18432, 'o', 18432, 'p', 18432, 'q', 18432, 'r', 18432, 's', 18432, 't', 18432, 'u', 18432, 'v', 18432, 'w', 18432, 'x', 18432, 'y', 18432, 'z', 18432, ' ', 206505]
Total number of words are: 206505
Ratio:
Character: 1 => 0.0227597394736
Ratio:
Character: 3 => 0.00242124888017
Ratio:
Character: 2 => 0.00435824798431
Ratio:
Character: 5 => 0.00338974843224
Ratio:
Character: 4 => 0.00242124888017
Ratio:
Character: 6 => 0.000968499552069
Ratio:
Character: 9 => 0.00193699910414
Ratio:
Character: 8 => 0.00193699910414
Ratio:
Character: a => 12.1270671412
Ratio:
Character: c => 3.44156315828
Ratio:
Character: b => 4.13791433621
Ratio:
Character: e => 1.52199704608
Ratio:
Character: d => 3.13697004915
Ratio:
Character: g => 1.66242948113
Ratio:
Character: f => 3.42316166679
Ratio:
Character: h => 18.432
Ratio:
Character: i => 18.432
Ratio:
Character: j => 18.432
Ratio:
Character: k => 18.432
Ratio:
Character: l => 18.432
Ratio:
Character: m => 18.432
Ratio:
Character: n => 18.432
Ratio:
Character: o => 18.432
Ratio:
Character: p => 18.432
Ratio:
Character: q => 18.432
Ratio:
Character: r => 18.432
Ratio:
Character: s => 18.432
Ratio:
Character: t => 18.432
Ratio:
Character: u => 18.432
Ratio:
Character: v => 18.432
Ratio:
Character: w => 18.432
Ratio:
Character: x => 18.432
Ratio:
Character: y => 18.432
Ratio:
Character: z => 18.432
Ratio:
Character: space => 206.505
```



```
<terminated> /Users/tejalgajare/Documents/workspace/MP1/mp1a.py
Ratio:
Character: f => 3.42316166679
Ratio:
Character: i => 7.77705140311
Ratio:
Character: h => 8.92569187187
Ratio:
Character: k => 0.782063388296
Ratio:
Character: j => 0.270695624803
Ratio:
Character: m => 3.87593520738
Ratio:
Character: l => 2.64836202513
Ratio:
Character: o => 5.18631510133
Ratio:
Character: n => 2.42996537614
Ratio:
Character: q => 0.265853127043
Ratio:
Character: p => 2.8895184136
Ratio:
Character: s => 7.69666594029
Ratio:
Character: r => 2.59703154887
Ratio:
Character: u => 1.09004624585
Ratio:
Character: t => 13.3284908356
Ratio:
Character: w => 7.17173918307
Ratio:
Character: v => 0.520084259461
Ratio:
Character: y => 2.81930219607
Ratio:
Character: x => 0.000484249776034
Ratio:
Character: z => 0.0949129561028
```

## Output Screens

### Application mp1b.py:

Aim: To create a Python script that accepts 100 random three-dimensional coordinate (x,y,z) tuples into a list called 3Space and to calculate and display the two coordinates that are the nearest (named minCoord) and farthest (named maxCoord) from the origin.

### Project Description:

- Create 3 Lists of 100 Random Coordinates for x, y, z with range 250 to -250
- Sort the lists of tuples in ascending order
- Store the sorted list in a dictionary element
- Using Distance Formula calculate the nearest and the farthest distance of two coordinate sets from the origin
- Display all the results

### Project Insight:

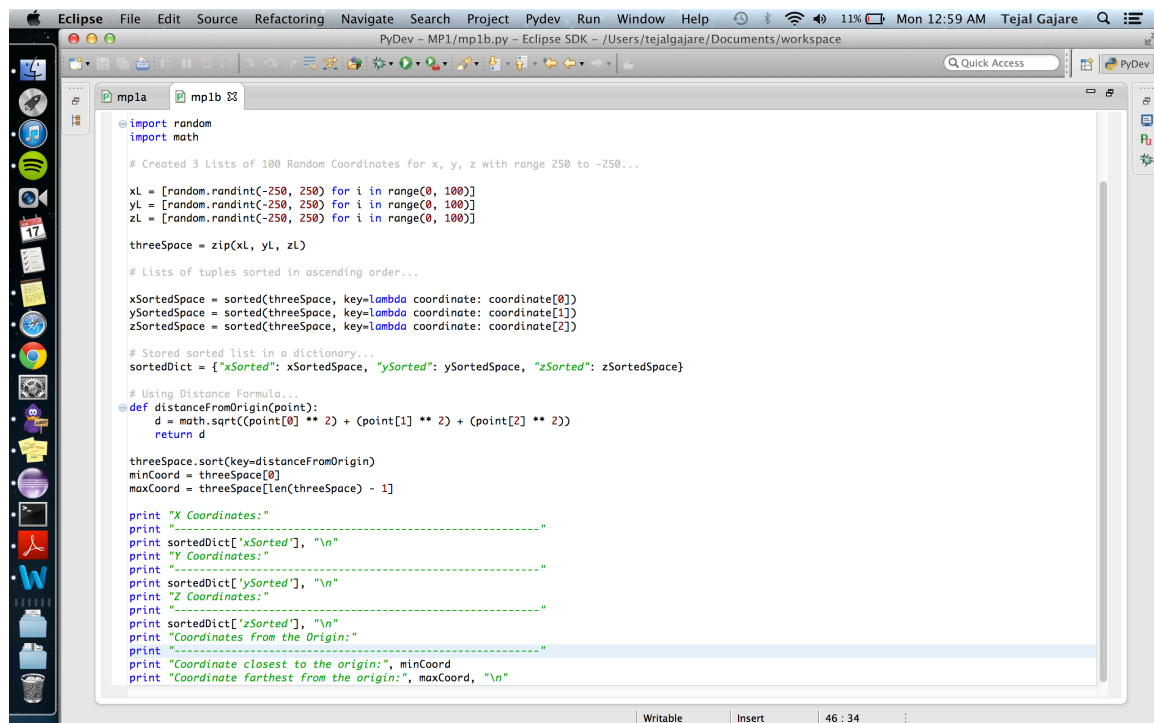
- Import random and math and then use random.randint in order to form the x, y and z co-ordinates lists
- Zip these lists into one threeSpace by using 'zip'
- Sort these lists in ascending order

- Call the distance formula function and sort on the returned value which is the key

### Expected Results:

- The program will display the sorted lists of x, y and z co-ordinates
- It will also display the 2 sets of threeSpace nearest and farthest from the origin

### Screen Captures:



```
import random
import math

# Created 3 Lists of 100 Random Coordinates for x, y, z with range 250 to -250...
xl = [random.randint(-250, 250) for i in range(0, 100)]
yl = [random.randint(-250, 250) for i in range(0, 100)]
zl = [random.randint(-250, 250) for i in range(0, 100)]

threeSpace = zip(xl, yl, zl)

# Lists of tuples sorted in ascending order...
xSortedSpace = sorted(threeSpace, key=lambda coordinate: coordinate[0])
ySortedSpace = sorted(threeSpace, key=lambda coordinate: coordinate[1])
zSortedSpace = sorted(threeSpace, key=lambda coordinate: coordinate[2])

# Stored sorted list in a dictionary...
sortedDict = {"xSorted": xSortedSpace, "ySorted": ySortedSpace, "zSorted": zSortedSpace}

# Using Distance Formula...
def distanceFromOrigin(point):
    d = math.sqrt((point[0] ** 2) + (point[1] ** 2) + (point[2] ** 2))
    return d

threeSpace.sort(key=distanceFromOrigin)
minCoord = threeSpace[0]
maxCoord = threeSpace[len(threeSpace) - 1]

print "X Coordinates:"
print "-----"
print sortedDict['xSorted'], "\n"
print "Y Coordinates:"
print "-----"
print sortedDict['ySorted'], "\n"
print "Z Coordinates:"
print "-----"
print sortedDict['zSorted'], "\n"
print "Coordinates from the Origin:"
print "-----"
print "Coordinate closest to the origin:", minCoord
print "Coordinate farthest from the origin:", maxCoord, "\n"
```

### Input Screen



- Python 2.7 / Python 3
- IDE: Eclipse Juno (Configure Eclipse to have PyDev)
- OS used: Mac OS 10.8.3
- Numpy, Matplotlib and FreeType for plotting histogram