Sagarika Muniraj
A20295475

ITMD – 513 Open Source Programming

Summer 2013 – MP1

This mini-project is to understand the basics of Python and get aware of its strengths.

The following concepts of Python are better understood through this project:

- Variable Types & Basic Operations
- Decision Making
- Loops
- Number, String functions
- Tuples
- Dictionary
- Lists

The project is basically to inculcate a strong foundation in reading input from URLs, usage of Data types and data structures in Python.

System Requirement

- OS – Windows 7
- IDE – Eclipse
- Python 2.7

Insights

MP1a.py

This script is developed to read the data from a URL and built a histogram with the different words in it along with their frequency of occurrences.

Also, the script counts the number of words that start with each English alphabet and calculates its percentage.

Few assumptions made and the insights:

- Words are case insensitive, ie – 'The' is considered to be the same as 'the' etc
- Punctuation marks at the end of the word are ignored (Special characters like '**?**' ,'/', '**!**', '**_**', '**-**', '**.**', '**,**')
- Spaces at the end of the words are ignored
- A dictionary with the words and the times of their occurrences is printed as output

- 26 lists are created, each list corresponding to an English alphabet. Each list contains
    - Letter alphabet
    - Number of words starting with the letter
    - Percentage of occurrence against the total number of words

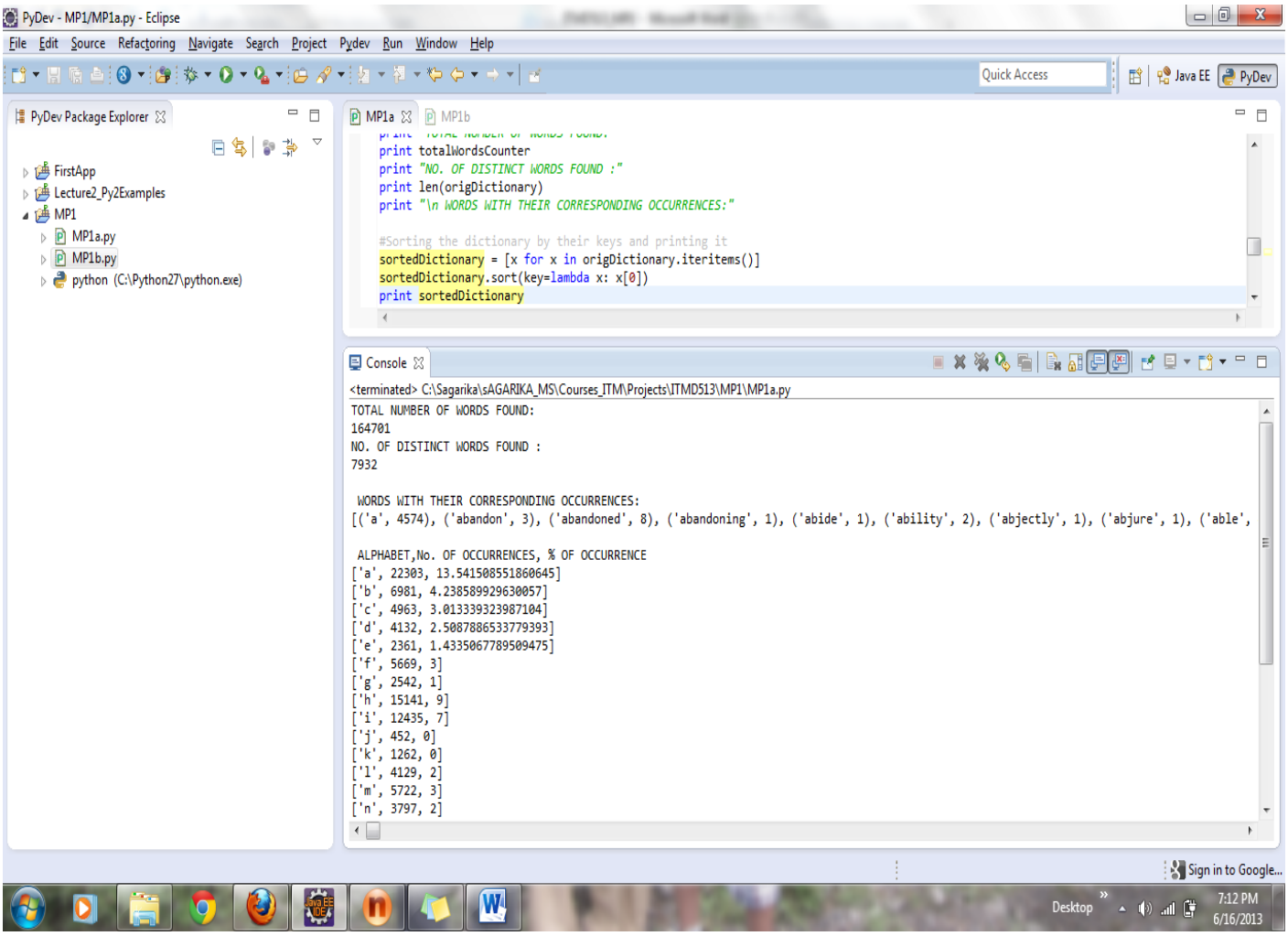
- The above lists are printed as output

Expected Result

- Total number of words
- Total number of distinct words
- Dictionary with key = word & value = number of occurrences
- 26 Lists, each consisting of alphabet, number of words starting with the alphabet, percentage of the number of words against the total number of words

TestCases

Output depicting the whole run



MP1b.py

This script generated 100 3-tuple random values, each depicting a 3-dimensional co-ordinate.

These tuples are sorted by x,y,z axes . The xSortedSpace holds the sorted list along x-axes, ySortedSpace along y and zSortedSpace along z axes.

Distance to each of these points from origin is calculated. minCoord and maxCoord denote the nearest and the farthest points from origin

Insights

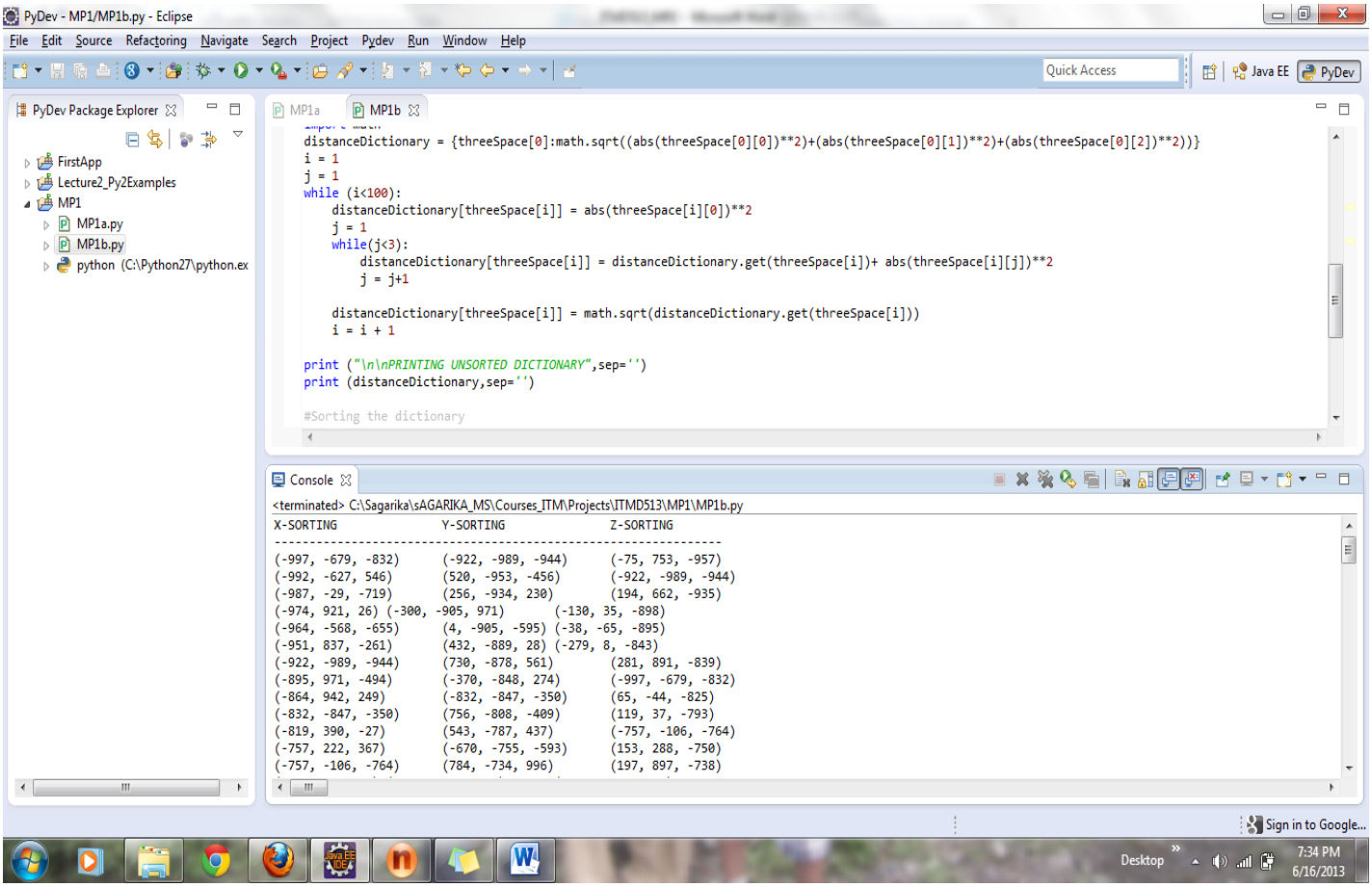
- threeSpace is created with 100 random 3-tuple co-ordinates
- These coordinates are sorted along x,y and z axes and stored in xSortedSpace, ySortedSpace and zSortedSpace respectively
- Distance to each of the points from origin is calculated and stored in distanceDictionary
- Nearest and farthest points are found out and printed as minCoord and maxCoord

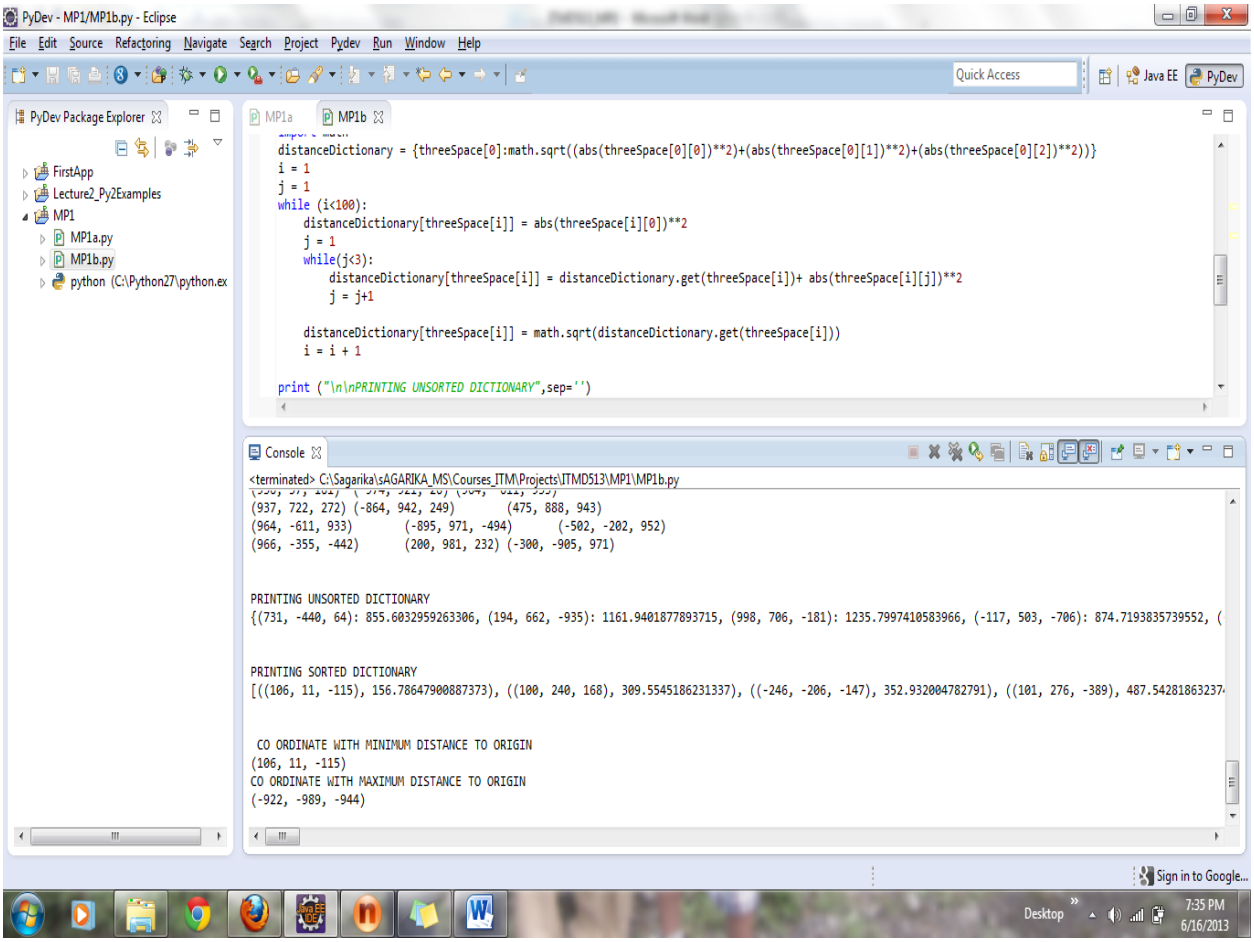
Expected Results

- Lists with sorting along x, y and z axes
  xSortedSpace <tab> ySortedSpace <tab> zSortedSpace
- Print the unsorted dictionary – points with its from origin
- Print the sorted dictionary – sorted order of points with their distances from the origin
- Co-ordinates with the minimum and maximum distance

Testcases

Screenshot depicting the entire output


PyDev - MP1/MP1b.py - Eclipse
File Edit Source Refactoring Navigate Search Project Pydev Run Window Help
PyDev Package Explorer
FirstApp
Lecture2_Py2Examples
MP1
MP1a.py
MP1b.py
python (C:\Python27\python.ex
MP1a
MP1b
distanceDictionary = {threeSpace[0]:math.sqrt((abs(threeSpace[0][0])**2)+(abs(threeSpace[0][1])**2)+(abs(threeSpace[0][2])**2))}
i = 1
j = 1
while (i<100):
distanceDictionary[threeSpace[i]] = abs(threeSpace[i][0])**2
j = 1
while(j<3):
distanceDictionary[threeSpace[i]] = distanceDictionary.get(threeSpace[i])+ abs(threeSpace[i][j])**2
j = j+1
distanceDictionary[threeSpace[i]] = math.sqrt(distanceDictionary.get(threeSpace[i]))
i = i + 1
print ("\n\nPRINTING UNSORTED DICTIONARY",sep='')
print (distanceDictionary,sep='')
#Sorting the dictionary
Console
<terminated> C:\Sagarika\sAGARIKA_MS\Courses_ITM\Projects\ITMD513\MP1\MP1b.py
X-SORTING Y-SORTING Z-SORTING
(-997, -679, -832) (-922, -989, -944) (-75, 753, -957)
(-992, -627, 546) (520, -953, -456) (-922, -989, -944)
(-987, -29, -719) (256, -934, 230) (194, 662, -935)
(-974, 921, 26) (-300, -905, 971) (-130, 35, -898)
(-964, -568, -655) (4, -905, -595) (-38, -65, -895)
(-951, 837, -261) (432, -889, 28) (-279, 8, -843)
(-922, -989, -944) (730, -878, 561) (281, 891, -839)
(-895, 971, -494) (-370, -848, 274) (-997, -679, -832)
(-864, 942, 249) (-832, -847, -350) (65, -44, -825)
(-832, -847, -350) (756, -808, -409) (119, 37, -793)
(-819, 390, -27) (543, -787, 437) (-757, -106, -764)
(-757, 222, 367) (-670, -755, -593) (153, 288, -750)
(-757, -106, -764) (784, -734, 996) (197, 897, -738)
Sign in to Google...
Desktop
7:34 PM
6/16/2013



Conclusion

Thus the implementation of the project successfully meets the requirements.