**Lab 13 - Dictionaries and sets**

**Submission:**

This lab has to be submitted on Week 7, together with all other labs published before Week 7. You will have to explain your code, so make sure it is well documented and you understand it.

**Questions:**

**1.** Given D = {'a':3, 'x':7, 'r':5}:

(a) Write Python code that returns the value at key 'x'.  
(b) Write Python code that returns the key given value '7'.

**2.** If my dict = {'a':15 , 'c':35, 'b':20}, write Python code:

(a) to print all the keys.  
(b) to print all the values.  
(c) to print all the keys and values pairs.  
(d) to print all the keys and values pairs in order of key.  
(e) to print all the keys and values pairs in order of value.

**3.** Write a Python script to generate and print a dictionary that contains a number  
(between 1 and n) in the form (x, x\*x) .

**4.** Write a Python program to combine these two dictionaries adding values for  
common keys.

d1 = {'a': 100, 'b': 200, 'c':300}  
d2 = {'a': 300, 'b': 200, 'd':400}

Sample output: d3 = {'a': 400, 'b': 400, 'd': 400, 'c': 300}

**5.** Use a dictionary to create a program that prompts for an integer and prints out  
the integer using words. For example: 138 will print “one three eight”

**6.** Create a dictionary that maps countries to their capitals. You may start with an  
empty dictionary. Ask the user to input the name of the country and its capital and  
add them to the dictionary. E.g., capitals = {'Argentina':'Buenos Aires','France':'Paris', 'US': 'Washington D.C.'} Once the dictionary is created, print the capitals in alphabetical order.

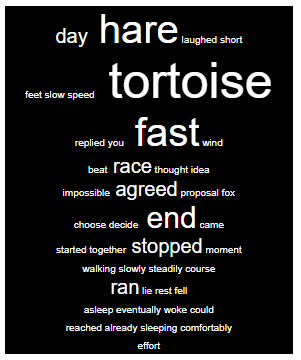
**7.** Given a string S of lower case characters check whether this string is Heterogram or not. A heterogram is a word, phrase, or sentence in which no letter of the alphabet occurs more than once. Hint: remember that sets by definition do not have repeated elements. So try to somehow use sets to solve the problem. Check whether the strings below are Heterograms or not:

string1 = 'the big dwarf only jumps'  
string2 = 'given string is Heterogram'

**8.** Given a string S of lower case characters check whether this string is a Pangram or not. Pangram is a sentence using every letter of a given alphabet at least once. Check whether the string below is a Pangram or not:

string = 'The quick brown fox jumps over the lazy dog'

**9.** Make a word cloud. A word cloud is a visual representation for text data typically used to depict keyword metadata on websites, or to visualize free form text. An example is given below:



The text of the HTML page is provided below:

<!DOCTYPE html>  
<html>  
<head lang="en">  
<meta charset="UTF-8">  
<title>Tag Cloud Generator</title>  
</head>  
<body>  
<div style="text-align: center; width: 15%; vertical-align: middle; font-family: arial; color: white; background-color:black; border:1px solid black">  
**\*\* Your SPAN elements should be inserted here \*\***  
</div>  
</body>  
</html>

The format of a span element is

<span style="font-size: XXpx"> WORD </span>

where XX is the size in pixels and WORD is the word being represented.  
So for example, <span style="font-size: 20px"> our </span>

1. The first part will be to read the speech from the hare\_and\_tortoise.txt file in brightspace and populate a dictionary using each word as a key, and value equals the frequency of the word. Your dictionary may look something like this:

{'tortoise': 5, 'hare': 4, 'stopped': 2, …}

1. For each word in the dictionary of frequencies you’ll need to write a SPAN tag to your HTML file. The font size will vary depending on the frequency of the word – for example if you use count\*10 to calculate the words’ size, then words that appear once will be size 10px, words that appear twice 20px, etc.
2. You should then create the completed HTML page and write it as a .html file. You can open this file to test it using a browser of your choice.
3. Modify your program to exclude common words from showing in the cloud. You can get a file of stopwords (like a, the, this, there, etc) in brightspace.