**Python Assignment**

*(Solve using Python 2.7 or Python 3.5)*

1. Given a list, url = [[www.annauniv.edu](http://www.annauniv.edu), [www.google.com](http://www.google.com), [www.ndtv.com](http://www.ndtv.com), [www.website.org](http://www.website.org), [www.bis.org.in](http://www.bis.org.in), [www.rbi.org.in](http://www.rbi.org.in)]; Sort the list based on the top level domain (edu, com, org, in) using custom sorting

Program :

topDomainList=["com","org","net","in","edu","gov"]

urlList = ["www.annauniv.edu", "www.google.com", "www.ndtv.com", "www.website.org", "www.bis.org.in", "www.rbi.org.in"]

def sortDomainNames(domainList,urlList):

tempDomainList = domainList

tempUrlList = urlList

sortedUrlList = []

for v in tempDomainList:

for v1 in tempUrlList:

if v1.endswith(v):

sortedUrlList.append(v1)

print(sortedUrlList)

sortDomainNames(topDomainList,urlList)

1. Given a list of strings, return the count of the number of strings where the string length is 2 or more and the first and last chars of the string are the same.
2. ['axa', 'xyz', 'gg', 'x', 'yyy']
3. ['x', 'cd', 'cnc', 'kk']
4. ['bab', 'ce', 'cba', 'syanora']

Program :

l=['axa', 'xyz', 'gg', 'x', 'yyy']

for s in l:

if len(s)>=2:

length=len(s)

if s[0]==s[length-1]:

print(s)

1. Given a list of strings, return a list with the strings in sorted order, except group all the strings that begin with 'x' first. e.g. ['mix', 'xyz', 'apple', 'xanadu', 'aardvark'] yields

['xanadu', 'xyz', 'aardvark', 'apple', 'mix'].

Hint: this can be done by making 2 lists and sorting each of them before combining them.

1. ['bbb', 'ccc', 'axx', 'xzz', 'xaa']
2. ['mix', 'xyz', 'apple', 'xanadu', 'aardvark']

Program :

l=['mix', 'xyz', 'apple', 'xanadu', 'aardvark']

l1=[]

l2=[]

for s in l:

if s[0]=='x':

l1.append(s)

else :

l2.append(s)

l1.sort()

l2.sort()

l1.extend(l2)

print(l1)

1. Given a list of non-empty tuples, return a list sorted in increasing order by the last element in each tuple.

e.g. [(1, 7), (1, 3), (3, 4, 5), (2, 2)] yields [(2, 2), (1, 3), (3, 4, 5), (1, 7)]

Hint: use a custom key= function to extract the last element form each tuple.

1. [(1, 3), (3, 2), (2, 1)]
2. [(1, 7), (1, 3), (3, 4, 5), (2, 2)]

Program :

def last(a):

return a[-1]

l1=[(1, 3), (3, 2), (2, 1)]

l2=[(1, 7), (1, 3), (3, 4, 5), (2, 2)]

l1.sort(key=last)

l2.sort(key=last)

print(l1)

print(l2)

1. Given a list of numbers, return a list where all adjacent == elements have been reduced to a single element, so [1, 2, 2, 3] returns [1, 2, 3]. You may create a new list or modify the passed in list.
2. [1, 2, 2, 3], [2, 2, 3, 3, 3]

Program :

l1=[1, 2, 2, 3]

l2=[2, 2, 3, 3, 3]

l1=list(set(l1))

l2=list(set[l2])

print(l1)

print(l2)

1. Write a function to print the information in the dictionary(bookstore) in the given format

bookstore={"New Arrivals":{"COOKING":["Everyday Italian","Giada De Laurentiis","2005","30.00"],"CHILDREN":["Harry Potter”, J K. Rowling","2005","29.99"],"WEB":["Learning XML","Erik T. Ray","2003","39.95"]}}

BOOKSTORE

'Learning XML', 'Erik T. Ray', '2003', '39.95'

'Everyday Italian', 'Giada De Laurent is', '2005', '30.00']

'Harry Potter', 'J K. Rowling', '2005', '29.99']

Program :

bookstore={"New Arrivals":{"COOKING":["Everyday Italian","Giada De Laurentiis","2005","30.00"],"CHILDREN":["Harry Potter”, J K. Rowling","2005","29.99"],"WEB":["Learning XML","Erik T. Ray","2003","39.95"]}}

l=[]

l=list(bookstore['New Arrivals'].values())

print("BOOKSTORE \n")

for i in range(0,len(l)):

for j in range(0,len(l[i])):

print(l[i][j],end=",")

print("\n")

1. Given a string, str1=""”Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly braces or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale .Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation."""
   1. Build a dictionary, with "words as key" --> Frequency of occurrence as value

E.g. Python 🡪7, is🡪3

* 1. Print the top 5 words with their frequency of occurrence

Program :

def keyfunction(k):

return d[k]

str1="""Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly braces or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale . Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython , the reference implementation of Python , is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.""

l=str1.split(" ")

count=0

d={}

d1={}

l1=[]

for i in range(0,len(l)):

l1.append(d.keys())

if l[i] not in l1:

for j in range(0,len(l)):

if l[i]==l[j]:

count+=1

d[l[i]]=count

count=0

for key in sorted(d, key=keyfunction, reverse=True)[:5]:

print ("%s\t -> %i" % (key, d[key]))

OUTPUT:

a -> 8

and -> 8

Python -> 7

code -> 4

to -> 4

1. Given a string, str1=""”Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly braces or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale .Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation."""

**Hint:** Assume that the first word is preceded by " "

* 1. Build a dictionary where the key is a word and the value is the list of words that are likely to follow.
     1. E.g. Python 🡪 [is, has, features, interpreters, code, Software]. In this example the words listed are likely to follow “Python”
  2. Given a word predict the word that’s likely to follow.

Program :

str1=""" Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly braces or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale . Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library. Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython, the reference implementation of Python , is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation."""

l=str1.split(" ")

d={}

l1=[]

word="Python"

for i in range(0,len(l)):

if word==l[i]:

l1.append(l[i+1])

d[word]=l1

print(d)

OUTPUT:

(base) C:\Users\xwankan\Desktop\python>python untitled6.py

{'Python': ['is', 'has', 'features', 'interpreters', 'code', ',', 'Software']}

1. Below is the output of # show ip interface brief command on a router

**Interface IP-Address OK?** Method Status Protocol

FastEthernet0/0 192.168.1.242 YES manual up up

FastEthernet1/0 unassigned YES unset down

Serial2/0 192.168.1.250 YES manual up up

Serial3/0 192.168.1.233 YES manual up up

FastEthernet4/0 unassigned YES unset down

FastEthernet5/0 unassigned YES unset down

* 1. Use regular expressions to extract and display Interface and method status for all the interfaces.
     1. E.g. FastEthernet0/0, manual up

Program :

import re

str1="""Interface IP-Address OK? Method Status Protocol

FastEthernet0/0 192.168.1.242 YES manual up up

FastEthernet1/0 unassigned YES unset down

Serial2/0 192.168.1.250 YES manual up up

Serial3/0 192.168.1.233 YES manual up up

FastEthernet4/0 unassigned YES unset down

FastEthernet5/0 unassigned YES unset down"""

print("\n\n\n")

for line in str1.splitlines():

matchObj = re.match( r'(\w+\d\/\d)\s+[.0-9a-z]+\s+\w+\s+(\w+\s?\w+?)\s+\w+', line, re.M|re.I)

if matchObj:

print(matchObj.group(1),",",matchObj.group(2),"\n")

print("\n\n\n")

OUTPUT:

FastEthernet0/0 , manual up

FastEthernet1/0 , unset

Serial2/0 , manual up

Serial3/0 , manual up

FastEthernet4/0 , unset

FastEthernet5/0 , unset

1. Given a number line from -infinity to +infinity. You start at 0 and can go either to the left or to the right. The condition is that in i’th move, you take i steps. In the first move take 1 step, second move 2 steps and so on.

**Hint:** 3 can be reached in 2 steps (0, 1) (1, 3). 2 can be reached in 3 steps (0, 1) (1,-1) (-1, 2)

a) Find the optimal number of steps to reach position 1000000000 and -1000000000.