The exercise is executed in Python 3.7.3

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

```
url=["www.annauniv.edu","www.google.com","www.ndtv.com","www.website.org","www.bis.org.in","www.r
bi.org.in"]
DomainList = ["com","edu","in","org"]

def cusSortUrl(domainList,url):
  tempDomainList = domainList
  tempUrlList = url
  sortedUrlList = []

  for dindex,dvalue in enumerate(tempDomainList):
    for uind,uval in enumerate(tempUrlList):
      if uval.endswith(dvalue):
        sortedUrlList.append(uval)

  print (sortedUrlList)

cusSortUrl(DomainList,url)
```

```
['www.google.com', 'www.ndtv.com', 'www.annauniv.edu', 'www.bis.org.in', 'www.rbi.org.in',
'www.website.org']
```

---

2. 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.

   i. ['axa', 'xyz', 'gg', 'x', 'yyy']
   ii. ['x', 'cd', 'cnc', 'kk']
   iii. ['bab', 'ce', 'cba', 'syanora']

```
lista = ['axa','xyz','gg','x','yyy']
listb = ['x','cd','cnc','kk']
listc = ['bab','ce','cba','syanora']

def strcount(listt):
  cnt = 0
  for strg in listt:
    if (len(strg)>=2) and (strg[0] == strg[-1]):
      cnt+=1
  return cnt

print (strcount(lista))
print (strcount(listb))
print (strcount(listc))
```

```
3
2
1
```

---

3. 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'].

```
lista = ['bbb', 'ccc', 'axx', 'xzz', 'xaa']
listb = ['mix', 'xyz', 'apple', 'xanadu', 'aardvark']

xlist = []
def sortlist1(listt):
   for stng in listt[:]:
      if stng[0].startswith('x'):
         xlist.append(stng)
         listt.remove(stng)
   print(sorted(xlist)+sorted(listt))
   del xlist[:]

 print(sortlist1(lista))
print(sortlist1(listb))
```

```
['xaa', 'xzz', 'axx', 'bbb', 'ccc']
['xanadu', 'xyz', 'aardvark', 'apple', 'mix']
```

---

4. 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)]

```
tup1= [(1, 3), (3, 2), (2, 1)]
tup2 = [(1, 7), (1, 3), (3, 4, 5), (2, 2)]

def lastElement(tup):
   return tup[-1]

def srtTup(listt):
   tmpvalue = listt
   tmpvalue.sort(key = lastElement)
   return tmpvalue
```

```
print(srtTup(tup1))
print(srtTup(tup2))
```

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

5. 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.

   [1, 2, 2, 3], [2, 2, 3, 3, 3]

```
li_a=[1, 2, 2, 3]
li_b=[2, 2, 3, 3, 3]
def listUni(listt):
   i = 1
   for elt in listt[ : ]:
       while i < len(listt):
            if listt[i] == listt[i-1]:
               listt.pop(i)
               i -= 1
            i += 1
   return listt

print(listUni(li_a))
print(listUni(li_b))
```

```
[1, 2, 3]
[2, 3]
```

6. 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']
```

```
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"]}}
print (bookstore)

for dicn in bookstore.values():
    for lis in dicn.values():
        strn=str(lis)
        strn=strn[1:len(strn)-1]
        print (strn)
{'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']}}
```

```
'Everyday Italian', 'Giada De Laurentiis', '2005', '30.00'
'Harry Potter', 'J K.Rowling', '2005', '29.99'
'Learning XML', 'Erik T. Ray', '2003', '39.95'
```

---

7. Given a string

   Build a dictionary, with "words as key" --> Frequency of occurrence as value

   E.g. Python →7, is→3

   Print the top 5 words with their frequency of occurrence

```
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."

wordCountV={}
wordList=str1.split()
for word in wordList:
   wordCountV[word]=str1.count(word)



values=wordCountV.values()
values=sorted(values)
values.reverse()
```

```
topFive=values[0:5]
printCount=0

for v in topFive:
   for word in  wordCountV:
      if wordCountV[word] == v:
         print (word,wordCountV[word])
         printCount+=1
         if printCount==5:
            break
   if printCount==5:
      break
```

```
a 93
on 19
in 18
Python 9
and 9
```

---

8. Given a string

   Build a dictionary where the key is a word and the value is the list of words that are likely to follow.

   i. E.g. Python → [is, has, features, interpreters, code, Software]. In this example the words listed are likely to follow "Python"

   Given a word predict the word that's likely to follow.

```
 str1="Python is a widely used high-level programming langauage for general-purpose prograaming, 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 concpets in fewer lines of code than
possible languages such as C++ or Java. The language provides constructs inteneded to enable writing clear
programs on both a small scale and a large scale. Python featues a dynamic type system and sutomatic
memory management and supports multiple programming paradgms,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 opne source software and has a community-
based development model, as do nearly all of its variant implementations. CPython os managed by the non-
profit Python Software Foundation."

def getDicn(strn):
   dicn={}
   wordList=strn.split()
   length=len(wordList)
   for index,word in enumerate(wordList):
      if word in dicn:
         continue
```

```
        tmpList=[]
        for i in range(index,length):
          if index==length-1:
            break
          if word==wordList[i]:
            nextWord=wordList[i+1]
            tmpList.append(nextWord)

        dicn[word]=tmpList
    return dicn

def project(strn,word):
    wordDictionary=getDicn(strn)
    print ( "In the given string, the word ' ",word,"' is followed by the list of words",wordDictionary[word])

#calling the function
project(str1,"programming")
```

```
In the given string, the word '  programming ' is followed by the list of words ['langauage',
'paradgms,including']
```

---

9. 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
```

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

```
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    upup
Serial3/0          192.168.1.233  YES     manual    upup
FastEthernet4/0       unassigned     YES     unset     down
```

```
FastEthernet5/0       unassigned     YES     unset     down"""
print ("     ")
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))
```

```
FastEthernet0/0 , manual
FastEthernet1/0 , unset
Serial2/0 , manual
Serial3/0 , manual
FastEthernet4/0 , unset
FastEthernet5/0 , unset
```

---

10. 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.

```
def reachNumber(target):

   #Taking the abs to accept the positive and negative number
   target = abs(target)
   # Proceeding while sum is  smaller or difference is odd
   aggregate = 0
   step = 0
   while (aggregate < target or (aggregate - target) %
                          2 != 0) :
      step = step + 1
      aggregate = aggregate + step

   return step


target = 1000000000
print(reachNumber(target))
```

```
44723
```