

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Programme | : | **B.Tech** | Semester | : | **Win Sem 21-22** |
| Course | : | **Web Mining** | Code | : | **CSE3024** |
| Faculty | : | **Dr.Bhuvaneswari A** | Slot | : | **L7+L8** |
| Date | : | **25-01-2022** | Marks | : | **10 Points** |

**Exercise 3: Inverted Index Creation and Searching**

**COLAB LINK:**

[**https://colab.research.google.com/drive/1ume3NFBAEcWMeL8bRwcrOMlD7lkea1NQ?usp=sharing**](https://colab.research.google.com/drive/1ume3NFBAEcWMeL8bRwcrOMlD7lkea1NQ?usp=sharing)

1. Build the inverted index for the following documents:

ID1 : Selenium is a portable framework for testing web applications;

ID2 : Beautiful Soup is useful for web scraping with selenium;

ID3: It is a python-package for parsing the pages using selenium

ID4: Java programming can be used for web applications

ID5: scraping web and crawling web is useful with selenium framework

Sample Output:

Selenium 4 D1,1,0 D2,1,5

Portable 1 D1,1,1

SORTING VOCAB- Alphabetical order(ASC)

**Procedure:**

1. We create file.txt files according to the question.
2. We will then pre-process the documents, and then split the documents into tokens.
3. The pre-processing includes conversion to lower case, removal of numbers and other special characters, and stop word removal.
4. Store the pre-processed data in a variable data.
5. After this we need to calculate the number of occurences of all the words.
6. Also, the position of each word is shown in (x,y) format x being the document number and y being the offset position in that particular document.
7. This inverted index map created for documents is stored in inverted.txt.

**CODE:**

import re

documents =['file1','file2','file3','file4','file5']

index={}

for id,doc in enumerate(documents):

   filename = doc+".txt"

   with open(filename,'r') as fp:

    data = "".join(fp.readlines())

    data = data.lower()

    ext\_words = re.findall(r"([a-z0-9-]+)",data)

    for pos,word in enumerate(ext\_words):

      if word[-1]=='s':

        if word[:-1] in index:

          word = word[:-1]

        elif word[:-2] in index:

          word = word[:-2]

      if word not in index:

        index[word]={ "frequency":1, "listing": [(id+1,pos)] }

      else:

        index[word]['frequency']+=1

        index[word]['listing'].append((id+1,pos))

from collections import OrderedDict

index = OrderedDict(sorted(index.items()))

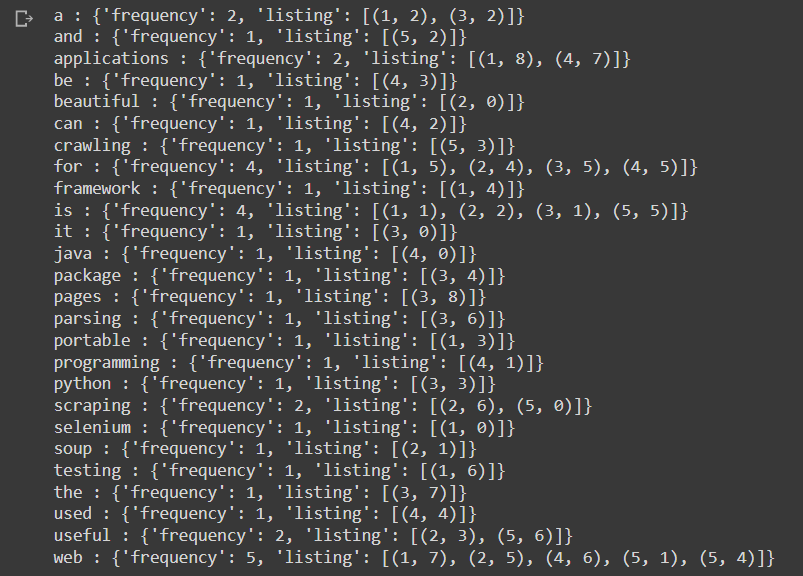
with open("inverted.txt",'w') as fp:

  for key in index:

    print(f"{key} : {index[key]}")

    fp.write(f"{key} : {index[key]}\n")

**OUTPUT:**

****

1. Search following words using the inverted index The output should be the document ids containing the search keywords
   1. Selenium AND web
   2. Selenium
   3. Python OR java
   4. Web AND crawl

**PROCEDURE:**

1. Create .txt files according to the question.
2. We will then pre-process the documents, and then split the documents into tokens.
3. The pre-processing includes conversion to lower case, removal of numbers and other special characters, and stop word removal.
4. Store the pre-processed data in a variable data.
5. After this we need to calculate the number of occurences of the specific word given in the question.
6. The word is searched through regular expression and if the word/s are found then their frequency and position is displayed.
7. The position of the words is shown in (x,y) format x being the document number and y being the offset position in that particular document.

**a.**

**CODE:**

import re

documents =['file1','file2','file3','file4','file5']

index={}

for id,doc in enumerate(documents):

   filename = doc+".txt"

   with open(filename,'r') as fp:

    data = "".join(fp.readlines())

    data = data.lower()

    if re.findall(r"\bselenium\b", data) and re.findall(r"\bweb\b", data):

      print("Match found in", filename)

      ext\_words1 = re.findall(r"\bselenium\b", data)

      ext\_words2 = re.findall(r"\bweb\b", data)

      for pos,word in enumerate(ext\_words1):

        if word not in index:

          index[word]={ "frequency":1, "listing": [(id+1,pos)] }

        else:

          index[word]['frequency']+=1

          index[word]['listing'].append((id+1,pos))

      for pos,word in enumerate(ext\_words2):

        if word not in index:

          index[word]={ "frequency":1, "listing": [(id+1,pos)] }

        else:

          index[word]['frequency']+=1

          index[word]['listing'].append((id+1,pos))

    else:

     print("No match has been found in", filename)

print("\n")

from collections import OrderedDict

index = OrderedDict(sorted(index.items()))

with open("inverted.txt",'w') as fp:

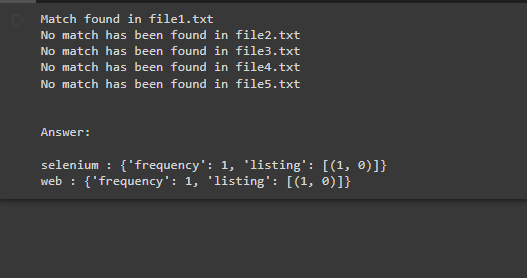
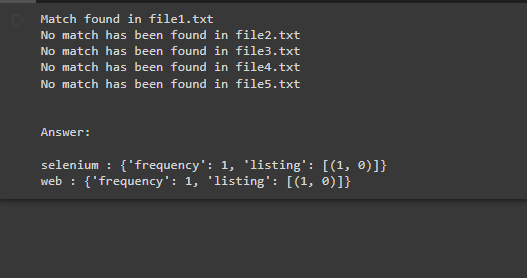
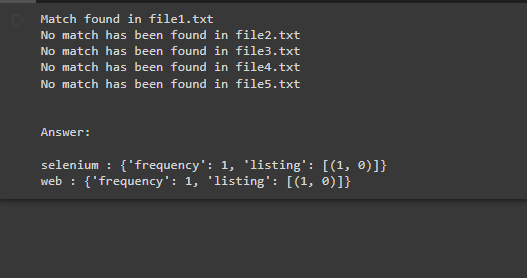
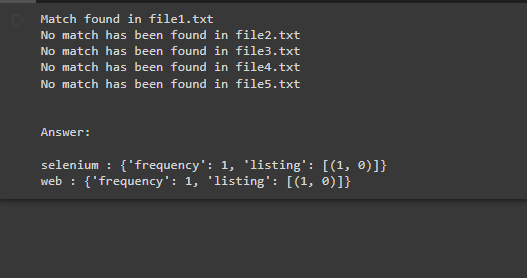
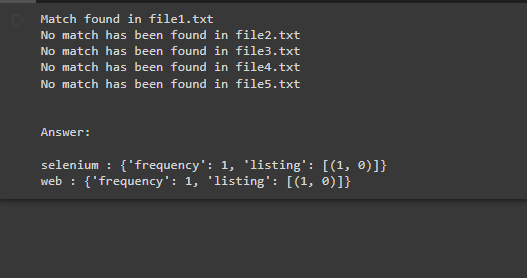
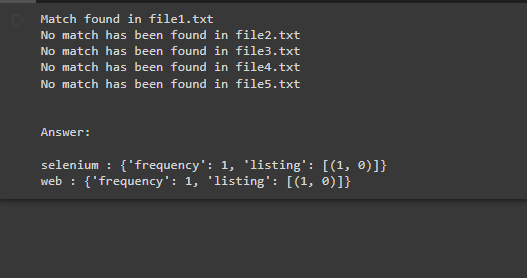
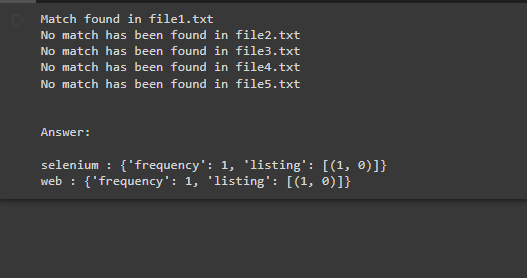
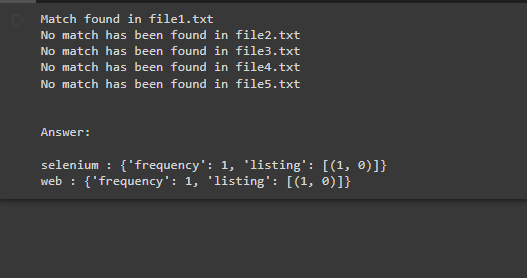
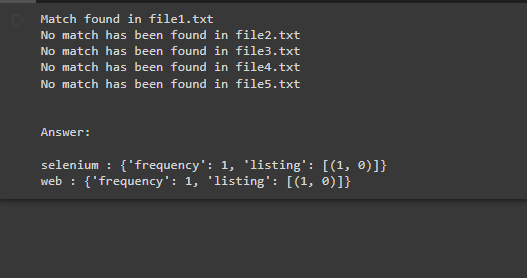
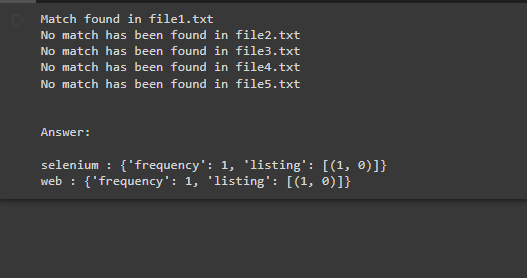
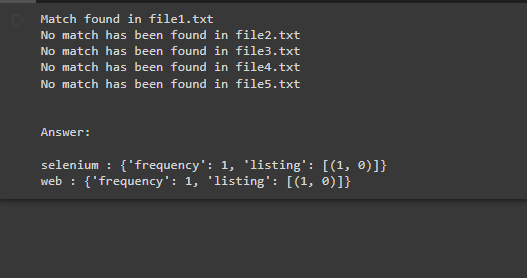
  print("Answer: \n")

  for key in index:

    print(f"{key} : {index[key]}")

    fp.write(f"{key} : {index[key]}\n")

**OUTPUT:**

****

**b.**

**CODE:**

import re

documents =['file1','file2','file3','file4','file5']

index={}

for id,doc in enumerate(documents):

   filename = doc+".txt"

   with open(filename,'r') as fp:

    data = "".join(fp.readlines())

    data = data.lower()

    ext\_words = re.findall(r"\bsoup\b", data)

    for pos,word in enumerate(ext\_words):

      if word not in index:

        index[word]={ "frequency":1, "listing": [(id+1,pos)] }

      else:

        index[word]['frequency']+=1

        index[word]['listing'].append((id+1,pos))

from collections import OrderedDict

index = OrderedDict(sorted(index.items()))

with open("inverted.txt",'w') as fp:

  for key in index:

    print(f"{key} : {index[key]}")

    fp.write(f"{key} : {index[key]}\n")

**OUTPUT:**

****

**c.**

**CODE:**

import re

documents =['file1','file2','file3','file4','file5']

index={}

for id,doc in enumerate(documents):

   filename = doc+".txt"

   with open(filename,'r') as fp:

    data = "".join(fp.readlines())

    data = data.lower()

    #print(len(data))

    ext\_words1 = re.compile(r"\bpython\b | \bjava\b",flags=re.I | re.X)

    ext\_words2=ext\_words1.findall(data)

    for pos,word in enumerate(ext\_words2):

      if word not in index:

        index[word]={ "frequency":1, "listing": [(id+1,pos)] }

      else:

        index[word]['frequency']+=1

        index[word]['listing'].append((id+1,pos))

from collections import OrderedDict

index = OrderedDict(sorted(index.items()))

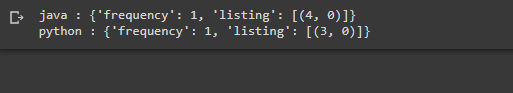
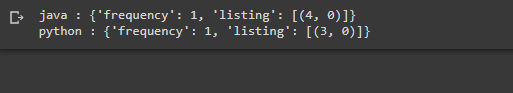
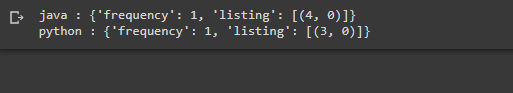
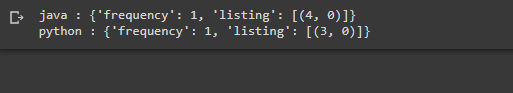
with open("inverted.txt",'w') as fp:

  for key in index:

    print(f"{key} : {index[key]}")

    fp.write(f"{key} : {index[key]}\n")

**OUTPUT:**

****

**d.**

**CODE:**

import re

documents =['file1','file2','file3','file4','file5']

index={}

for id,doc in enumerate(documents):

   filename = doc+".txt"

   with open(filename,'r') as fp:

    data = "".join(fp.readlines())

    data = data.lower()

    if re.findall(r"\bweb\b", data) and re.findall(r"\bcraw\b", data):

      print("Match found in", filename)

      ext\_words1 = re.findall(r"\bweb\b", data)

      ext\_words2 = re.findall(r"\bscraw\b", data)

      for pos,word in enumerate(ext\_words1):

        if word not in index:

          index[word]={ "frequency":1, "listing": [(id+1,pos)] }

        else:

          index[word]['frequency']+=1

          index[word]['listing'].append((id+1,pos))

      for pos,word in enumerate(ext\_words2):

        if word not in index:

          index[word]={ "frequency":1, "listing": [(id+1,pos)] }

        else:

          index[word]['frequency']+=1

          index[word]['listing'].append((id+1,pos))

    else:

     print("No match has been found in", filename)

print("\n")

from collections import OrderedDict

index = OrderedDict(sorted(index.items()))

with open("inverted.txt",'w') as fp:

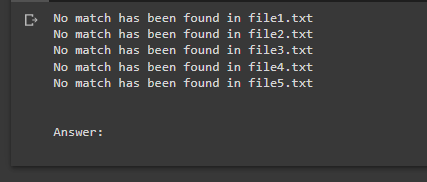
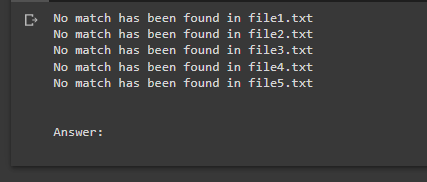
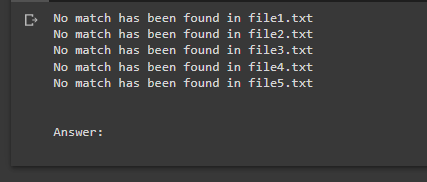
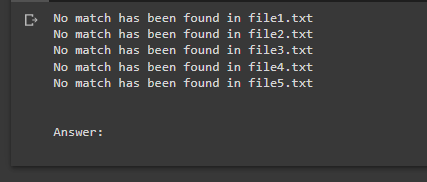
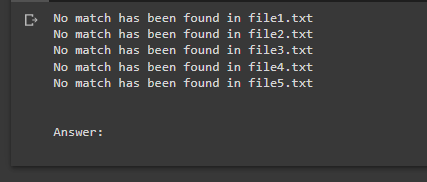
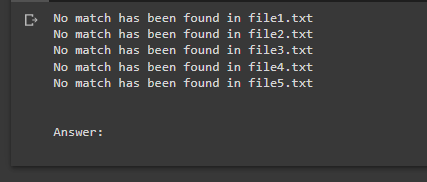
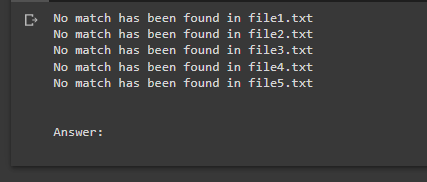
  print("Answer: \n")

  for key in index:

    print(f"{key} : {index[key]}")

    fp.write(f"{key} : {index[key]}\n")

**OUTPUT:**

****

**HINTS:**

We will first pre-process the documents, and then split the documents into tokens or words. The pre-processing includes conversion to lower case, removal of numbers and other special characters, and stop word removal. After this we tokenise this document using the **NLTK** library.

After this we then take all these words after tokenisation and form a inverted index, which will include the word, and a list of occurrences in all the documents, and in each entry of this array would include the document number, number of times this word has occurred in this document, and also a list of offset position where the word occurs in the document.This map developed in the above stated process is our **Inverted Index** map for the documents.