Assignment 1 IRS

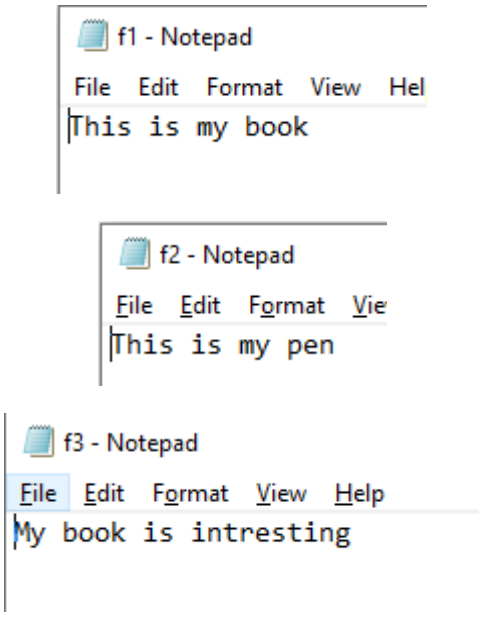
Information retrieval is the process of obtaining information system resources that are relevant to an information need from a collection of those resources. The core purpose of this assignment is to give you the flavor of IRS. You need to follow some steps listed below and in the end, you'll be able to build your own small IRS. So, let's start.

In [ ]:
```
# required imports
import numpy as np
import fnmatch
import os

```

Suppose we have 3 files containing data :

File Contents



Step 1 Create Files with Dummy data

You have to create few files with dummy data of your own choice as shown above.

Step 2 Traverse Directories

Now, You have to traverse the directories and store all the files into a dict type variable(files_dict).

In [ ]:
```
# Here we have intialized some variables, you can add more if required.

file_count = 0              # file_count to count number of files
files_dict = {}             # files_dic to store count of every file
unique_word_set = set()     # unique_word_set to store all the unique words i

```

In [ ]:
```
#Your code starts here

#Your code ends here
```

Displaying the count of files.

In [ ]:
```
print("\nTotal Number  of files\n", file_count)
```

Displaying Dictionary containing all files.

In [ ]:
```
print("\nDictionary containing  files\n", files_dict)
```

Step 3 Extract Unique Vocabulary

In [ ]:
```
# write code to print all the unique words in every file and store them in a
```

In [ ]:
```
#Your code starts here

#Your code ends here
```

Expected Output

```
unique words in files
 {'book', 'pen', 'my', 'this', 'intresting', 'is'}

count of files 3
```

Step 4 Create Term Document Matrix

Create Term-Doc-matrix using Bag of word approach.and display its contents initially and finally.

In [ ]:
```
# Create Term doc matrix such that colmns will be unique words and all the f
# Write code to count all the unique words appearances in all the files and
```

In [ ]:
```
#Your code starts here
#Your code ends here
```

Expected Output

```
TERM DOC MATRIX intially
 [[0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]]

dictionary of unique words
  {'book': 0, 'pen': 1, 'my': 2, 'this': 3, 'intresting': 4, 'is': 5}

dictionary of files
  {'f1.txt': 0, 'f2.txt': 1, 'f3.txt': 2}
```

Step 5 Fill Term Document Matrix

In [ ]:
```
# Fill the term doc matrix by checking if the unique word exists in a file o
# If it exists then substitute a 1 in term_doc_matrix (eg : TERM_DOC_MATRIX[
# Do the same for all the files present in the directory
```

In [ ]:
```
#Your code starts here

#Your code ends here
```

Expected Output

```
TERM DOC MATRIX after filling
 [[1. 0. 1. 1. 0. 1.]
 [0. 1. 1. 1. 0. 1.]
 [1. 0. 1. 0. 1. 1.]]
col vector(initially)
```

Step 6 Ask for a user Query

In [ ]:
```
# For user query make a column vector of length of all the unique words pres
```

In [ ]:
```
#Your code starts here

#Your code ends here
```

Expected Output

```
col vector(initially)

[[0.]
 [0.]
 [0.]
 [0.]
 [0.]
 [0.]]
```

In [ ]:
```
query = input("\nWrite something for searching  ")
# Check every word of query if it exists in the set of unique words or not
# If exixts then increment the count of that word in word dictionary

```

In [ ]:
```
#Your code starts here

#Your code ends here
```

Expected Output

```
Write something for searching  this book is intresting
```

Step 7 Display Resultant Vector

Display

1. Resultant vector.
2. Max value in resultant vector.
3. Index of max value in resultant vector.

In [ ]:
```
#Your code starts here

#Your code ends here
```

Expected Output

```
[[3.]
 [2.]
 [3.]]

Maximum in resultant is  3.0

 Index of maximum in resultant is  0
```

Step 8 Display the contents of file

In [ ]:
```
#Write the code to identify the file_name having maximum value in the result
```

In [ ]:
```
#Your code starts here

#Your code ends here
```

Congratulations Now you are able to build your own small IRS.