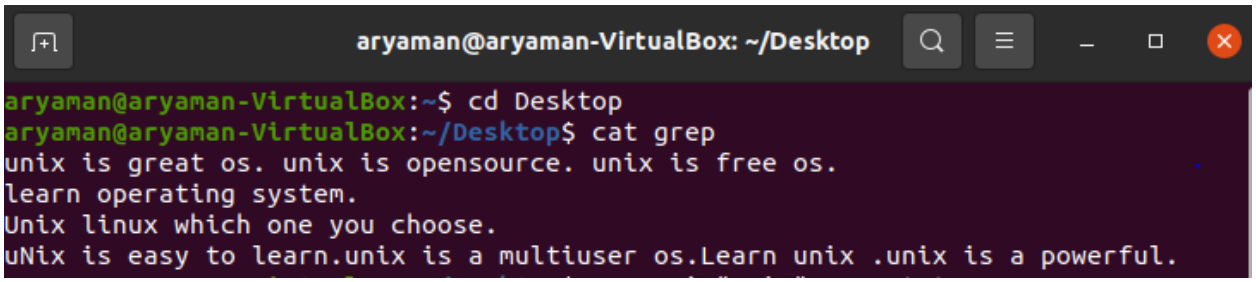
ARYAMAN MISHRA

19BCE1027 LAB 1

- Explore Unix command GREP



1. Case insensitive search : The -i option enables to search for a string case insensitively in the give file. It matches the words like “UNIX”, “Unix”, “unix”.

$grep -i "UNix" grep.txt unix is great os. unix is opensource. unix is free os.

Unix linux which one you choose.uNix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

2. Displaying the count of number of matches : We can find the number of lines that matches the given string/pattern

3. Display the file names that matches the pattern : We can just display the files that contains the given

PSEUDOCODE FOR NEXT QUESTION:

```
INTERSECT(p1, p2)
answer ← ⟨ ⟩
while p1 ≠ NIL and p2 ≠ NIL
do if docID(p1) = docID(p2)
      then ADD(answer, docID(p1))
           p1 ← next(p1)
           p2 ← next(p2)
      else if docID(p1) < docID(p2)
              then p1 ← next(p1)
              else p2 ← next(p2)
return answer
```

Write a program to create the inverted index and execute for the following document collections. (See Figure 1.3 for an example.)

a)

Doc 1 new home sales top forecasts
Doc 2 home sales rise in july
Doc 3 increase in home sales in july
Doc 4 july new home sales rise

b)

Doc 1 breakthrough drug for schizophrenia

Doc 2 new schizophrenia drug

Doc 3 new approach for treatment of schizophrenia

Doc 4 new hopes for schizophrenia patients

```
[57] doc_1 = "new home sales top forecasts "
     doc_2 = "home sales rise in july "
     doc_3 = "increase in home sales in july "
     doc_4 = "july new home sales rise"
```

```
docs = [doc_1, doc_2, doc_3,doc_4]
docs
```

```
['breakthrough drug for schizophrenia ',
 'new schizophrenia drug ',
 'new approach for treatment of schizophrenia ',
 'new hopes for schizophrenia patients']
```

```
[60] unique_terms = {term for doc in docs for term in doc.split()}
     unique_terms
```

```
{'forecasts', 'home', 'in', 'increase', 'july', 'new', 'rise', 'sales', 'top'}
```

-

```
[61] inverted_index = {}
     for i, doc in enumerate(docs):
         for term in doc.split():
             if term in inverted_index:
                 inverted_index[term].add(i)
             else: inverted_index[term] = {i}

     inverted_index
```

```
{'forecasts': {0},
 'home': {0, 1, 2, 3},
 'in': {1, 2},
 'increase': {2},
 'july': {1, 2, 3},
 'new': {0, 3},
 'rise': {1, 3},
 'sales': {0, 1, 2, 3},
 'top': {0}}
```

```
doc_1 = "breakthrough drug for schizophrenia "
doc_2 = "new schizophrenia drug "
doc_3 = "new approach for treatment of schizophrenia "
doc_4 = "new hopes for schizophrenia patients"
```

```
[63] docs = [doc_1, doc_2, doc_3,doc_4]
     docs
```

```
['breakthrough drug for schizophrenia ',
 'new schizophrenia drug ',
 'new approach for treatment of schizophrenia ',
 'new hopes for schizophrenia patients']
```

```
[65] unique_terms = {term for doc in docs for term in doc.split()}
     unique_terms
```

```
{'approach',
 'breakthrough',
 'drug',
 'for',
 'hopes',
 'new',
 'of',
 'patients',
 'schizophrenia',
 'treatment'}
```

```
[66] inverted_index2 = {}

     for i, doc in enumerate(docs):
         for term in doc.split():
             if term in inverted_index2:
                 inverted_index2[term].add(i)
             else: inverted_index2[term] = {i}

     inverted_index2
```

```
{'approach': {2},
 'breakthrough': {0},
 'drug': {0, 1},
 'for': {0, 2, 3},
 'hopes': {3},
 'new': {1, 2, 3},
 'of': {2},
 'patients': {3},
 'schizophrenia': {0, 1, 2, 3},
 'treatment': {2}}
```

Generate term-document incidence matrix for a) and b).

```
[67] doc_1 = "new home sales top forecasts "
     doc_2 = "home sales rise in july "
     doc_3 = "increase in home sales in july "
     doc_4 = "july new home sales rise"
```

```
[68] docs = [doc_1, doc_2, doc_3,doc_4]
     docs
```

```
['new home sales top forecasts ',
 'home sales rise in july ',
 'increase in home sales in july ',
 'july new home sales rise']
```

```
[69] unique_terms = {term for doc in docs for term in doc.split()}
     unique_terms
```

```
{'forecasts', 'home', 'in', 'increase', 'july', 'new', 'rise', 'sales', 'top'}
```

```python
doc_term_matrix = {}

for term in unique_terms:
    doc_term_matrix[term] = []

    for doc in docs:
        if term in doc:
            doc_term_matrix[term].append(1)
        else: doc_term_matrix[term].append(0)

doc_term_matrix
```

```
{'forecasts': [1, 0, 0, 0],
 'home': [1, 1, 1, 1],
 'in': [0, 1, 1, 0],
 'increase': [0, 0, 1, 0],
 'july': [0, 1, 1, 1],
 'new': [1, 0, 0, 1],
 'rise': [0, 1, 0, 1],
 'sales': [1, 1, 1, 1],
 'top': [1, 0, 0, 0]}
```

```python
[62] doc_1 = "breakthrough drug for schizophrenia "
     doc_2 = "new schizophrenia drug "
     doc_3 = "new approach for treatment of schizophrenia "
     doc_4 = "new hopes for schizophrenia patients"
```

```python
[63] docs = [doc_1, doc_2, doc_3,doc_4]
     docs
```

```
['breakthrough drug for schizophrenia ',
 'new schizophrenia drug ',
 'new approach for treatment of schizophrenia ',
 'new hopes for schizophrenia patients']
```

```python
[65] unique_terms = {term for doc in docs for term in doc.split()}
     unique_terms
```

```
{'approach',
 'breakthrough',
 'drug',
 'for',
 'hopes',
 'new',
 'of',
 'patients',
 'schizophrenia',
 'treatment'}
```

```python
for i, doc in enumerate(docs):
    for term in doc.split():
        if term in inverted_index2:
            inverted_index2[term].add(i)
        else: inverted_index2[term] = {i}

inverted_index2
```

```
{'approach': {2},
 'breakthrough': {0},
 'drug': {0, 1},
 'for': {0, 2, 3},
 'hopes': {3},
 'new': {1, 2, 3},
 'of': {2},
 'patients': {3},
 'schizophrenia': {0, 1, 2, 3},
 'treatment': {2}}
```

[67]
```python
doc_1 = "new home sales top forecasts "
doc_2 = "home sales rise in july "
doc_3 = "increase in home sales in july "
doc_4 = "july new home sales rise"
```

[68]
```python
docs = [doc_1, doc_2, doc_3,doc_4]
docs
```

```
['new home sales top forecasts ',
 'home sales rise in july ',
 'increase in home sales in july ',
 'july new home sales rise']
```

[69]
```python
unique_terms = {term for doc in docs for term in doc.split()}
unique_terms
```

```
{'forecasts', 'home', 'in', 'increase', 'july', 'new', 'rise', 'sales', 'top'}
```

```
[70] doc_term_matrix = {}

     for term in unique_terms:
         doc_term_matrix[term] = []

         for doc in docs:
             if term in doc:
                 doc_term_matrix[term].append(1)
             else: doc_term_matrix[term].append(0)

     doc_term_matrix
```

```
{'forecasts': [1, 0, 0, 0],
 'home': [1, 1, 1, 1],
 'in': [0, 1, 1, 0],
 'increase': [0, 0, 1, 0],
 'july': [0, 1, 1, 1],
 'new': [1, 0, 0, 1],
 'rise': [0, 1, 0, 1],
 'sales': [1, 1, 1, 1],
 'top': [1, 0, 0, 0]}
```

```
[71] doc_1 = "breakthrough drug for schizophrenia "
     doc_2 = "new schizophrenia drug "
     doc_3 = "new approach for treatment of schizophrenia "
     doc_4 = "new hopes for schizophrenia patients"
```

```python
docs = [doc_1, doc_2, doc_3,doc_4]
docs
```

```
['breakthrough drug for schizophrenia ',
 'new schizophrenia drug ',
 'new approach for treatment of schizophrenia ',
 'new hopes for schizophrenia patients']
```

```python
[73] unique_terms = {term for doc in docs for term in doc.split()}
     unique_terms
```

```
{'approach',
 'breakthrough',
 'drug',
 'for',
 'hopes',
 'new',
 'of',
 'patients',
 'schizophrenia',
 'treatment'}
```

```python
[74] doc_term_matrix = {}

     for term in unique_terms:
         doc_term_matrix[term] = []

         for doc in docs:
             if term in doc:
                 doc_term_matrix[term].append(1)
             else: doc_term_matrix[term].append(0)

     doc_term_matrix
```

```
{'approach': [0, 0, 1, 0],
 'breakthrough': [1, 0, 0, 0],
 'drug': [1, 1, 0, 0],
 'for': [1, 0, 1, 1],
 'hopes': [0, 0, 0, 1],
 'new': [0, 1, 1, 1],
 'of': [0, 0, 1, 0],
 'patients': [0, 0, 0, 1],
 'schizophrenia': [1, 1, 1, 1],
 'treatment': [0, 0, 1, 0]}
```

- For the document collections shown in a) and b), compute the results for these queries using above matrix as well as inverted index created above :

  a. schizophrenia AND drug

  b. for AND NOT(drug OR approach)

```
[75] import numpy as np

     docs_array = np.array(docs, dtype='object')

     v1 = np.array(doc_term_matrix['schizophrenia'])
     v2 = np.array(doc_term_matrix['drug'])
     print(v1)
     print(v2)
     v3=v1 & v2
     print('------------')
     print(v3)
```

```
[1 1 1 1]
[1 1 0 0]
------------
[1 1 0 0]
```

```
[76] [doc for doc in v3 * docs_array if doc]
```

```
['breakthrough drug for schizophrenia ', 'new schizophrenia drug ']
```

```
[80] v1 = np.array(doc_term_matrix['for'])
     v2 = np.array(doc_term_matrix['drug'])
     v3 = np.array(doc_term_matrix['approach'])
     print(v1)
     print(v2)
     print(v3)
     print('------------')
     v4 =v1 & ~(v2 | v3)
     print(v4)
```

```
[1 0 1 1]
[1 1 0 0]
[0 0 1 0]
------------
[0 0 0 1]
```

```
[doc for doc in v4 * docs_array if doc]
```

```
['new hopes for schizophrenia patients']
```