Sheet

```
d0 = "Welcom to our first lab session"
d1 = "In this lab, you will lean more about Term Document Incidence Matrix"
d2 = "You will also learn how to visualize it"
d3 = "Hope you enjoy learning it"
collection = {"doc0": d0,
"doc1": d1,
"doc2": d2,
"doc3": d3}
collection
```

```
{'doc0': 'Welcom to our first lab session',
 'doc1': 'In this lab, you will lean more about Term Document Incidence Matrix',
 'doc2': 'You will also learn how to visualize it',
 'doc3': 'Hope you enjoy learning it'}
```

```
boolean_operators = {'AND', 'OR', 'NOT'}
```

```
#list of terms
def get_terms (data):
    terms=[]
    for doc in data:
        for term in data[doc].split() :
            terms.append(term)
    return terms

#list of unique terms
def get_unique_terms(terms):
    unique_terms=[]
    for d in terms :
        if d not in unique_terms:
            unique_terms.append(d)
    return unique_terms

#document collection terms
def get_document_collection_terms(data):
    docs_colllection={}
    for doc in data:
        if doc not in boolean_operators :
            docs_colllection[doc]=get_unique_terms(data[doc].split())
    return docs_colllection

def display_dict(dic):
    print("\n")
    for i in dic:
        print (i , " : " ,dic[i])
    print("\n")
```

```
#print our collection
print("### documents content ###")
display_dict(collection)
```

```
### documents content ###


doc0  :  Welcom to our first lab session
doc1  :  In this lab, you will lean more about Term Document Incidence Matrix
doc2  :  You will also learn how to visualize it
doc3  :  Hope you enjoy learning it
```

```
#print the terms available in the collection
print ("\n### Terms in all docs ###\n" , *get_terms(collection) ,sep= " \n ")
```

```
### Terms in all docs ###

 Welcom
 to
 our
 first
 lab
 session
 In
 this
 lab,
 you
 will
 lean
 more
 about
 Term
 Document
 Incidence
 Matrix
 You
 will
 also
 learn
 how
 to
 visualize
 it
 Hope
```

```
you
enjoy
learning
it
```

```
#print the unique_terms available in the collection
terms=get_terms(collection)
print ("\n### Unique Terms in 5 DOCs###\n", *get_unique_terms(terms) ,sep=" | ")
```

```
### Unique Terms in 5 DOCs###
 | Welcom | to | our | first | lab | session | In | this | lab, | you | will | lear
```

```
#print the unique terms in each collection in a dictionary format
print ("\n###Document terms Collection ###" )
display_dict(get_document_collection_terms(collection))
```

```
###Document terms Collection ###


doc0  :  ['Welcom', 'to', 'our', 'first', 'lab', 'session']
doc1  :  ['In', 'this', 'lab,', 'you', 'will', 'lean', 'more', 'about', 'Term', 'Do
doc2  :  ['You', 'will', 'also', 'learn', 'how', 'to', 'visualize', 'it']
doc3  :  ['Hope', 'you', 'enjoy', 'learning', 'it']
```

```
#this function takes the collection of documents in a form of dictionary as an input
def term_document_incidence_matrix(collection):
    ## list of terms from the data file collection
    terms = get_terms(collection)
    #list of unique terms
    uique_terms = get_unique_terms(terms)
    #Document collection terms
    docs_terms=get_document_collection_terms(collection)
    #TermDocumentIncidenceMatrix
    term_docs_matrix= { }
    for term in uique_terms :
        vector=[]
        for c in docs_terms:
            if term in docs_terms[c]:
                vector.append(1)
            else :
```

```
                vector.append(0)

        term_docs_matrix[term]=vector
    return term_docs_matrix

#this fucntion takes a term and a terms-document incidence matrix and returns the inc
#this function just for explanation and display purposes
def term_incidence_vector(term,term_docs_incid_matrix):
    try:
        return term_docs_incid_matrix[term]
    except:
        return "term not found"
#build the term-document incidence matrix
term_docs_incid_matrix=term_document_incidence_matrix(collection)
print("Term-Document incidence Matrix\n")
#formatted Display
display_dict(term_docs_incid_matrix)
```

```
Term-Document incidence Matrix



Welcom  :  [1, 0, 0, 0]
to  :  [1, 0, 1, 0]
our  :  [1, 0, 0, 0]
first  :  [1, 0, 0, 0]
lab  :  [1, 0, 0, 0]
session  :  [1, 0, 0, 0]
In  :  [0, 1, 0, 0]
this  :  [0, 1, 0, 0]
lab,  :  [0, 1, 0, 0]
you  :  [0, 1, 0, 1]
will  :  [0, 1, 1, 0]
lean  :  [0, 1, 0, 0]
more  :  [0, 1, 0, 0]
about  :  [0, 1, 0, 0]
Term  :  [0, 1, 0, 0]
Document  :  [0, 1, 0, 0]
Incidence  :  [0, 1, 0, 0]
Matrix  :  [0, 1, 0, 0]
You  :  [0, 0, 1, 0]
also  :  [0, 0, 1, 0]
learn  :  [0, 0, 1, 0]
how  :  [0, 0, 1, 0]
visualize  :  [0, 0, 1, 0]
it  :  [0, 0, 1, 1]
Hope  :  [0, 0, 0, 1]
enjoy  :  [0, 0, 0, 1]
learning  :  [0, 0, 0, 1]
```

```
print("Incidence Vector of 'Matrix' ",term_incidence_vector('Matrix',term_docs_incid
```

```
Incidence Vector of 'Matrix'  [0, 1, 0, 0]
```

```
#Query Filteration
#input : Query
#output : List of terms of a given query which match with the terms in our collectior
def query_filteration(query,collection):
    terms= get_terms(collection)
    unique_terms=get_unique_terms(terms)
    qterms=[]
    splitted_query=query.split()
    for qterm in splitted_query:
        if qterm in unique_terms or qterm in boolean_operators:
            qterms.append(qterm)
    return qterms
```

```
query="learn AND Matrix AND you"
qterms=query_filteration(query,collection)
print(qterms)
```

```
['learn', 'AND', 'Matrix', 'AND', 'you']
```

```
#and should be capitalized and اليمن is not in our collection
query="learn and Matrix AND Taibah"
qterms=query_filteration(query,collection)
print(qterms)
```

```
['learn', 'Matrix', 'AND']
```

```
#Boolean Operator Processing
# input : Boolean Operator ,Next term Incedence Vector ,Previous term Incedence Vecto
def boolean_operator_processing(bool_operator,prevV,nextV):
    if bool_operator == "AND":
       return [a & b for a, b in zip(prevV, nextV)]
    elif bool_operator=="OR" :
        return [a | b for a, b in zip(prevV, nextV)]
    elif bool_operator == "NOT":
        return [1-a for a in prevV]
```

```
v1=term_incidence_vector('learn',term_docs_incid_matrix)
v2=term_incidence_vector('Matrix',term_docs_incid_matrix)
v3=[]
print('v1',v1)
print('v2',v2)
```

```
v1 [0, 0, 1, 0]
v2 [0, 1, 0, 0]
```

```
print(boolean_operator_processing("AND",v1,v2))
```

```
[0, 0, 0, 0]
```

```
print(boolean_operator_processing("OR",v1,v2))
```

```
[0, 1, 1, 0]
```

```
print(boolean_operator_processing("NOT",v1,v3))
```

```
[1, 1, 0, 1]
```

```
# Boolean retrieval function
# input : Query
def boolean_retrieval(query,collection):
    #build a terms_documents incidence matrix
    term_docs_incid_matrix=term_document_incidence_matrix(collection)
    bitwiseop=""
    #get the query terms
    qterms=query_filteration(query,collection)
    result=[]
    result_set={}
    has_previous_term=False
    has_not_operation=False
    inc_vec_prev=[]
    inc_vec_next=[]
    for term in qterms :
        if term not in boolean_operators:
```

```
            if has_not_operation:
                if has_previous_term:
                    inc_vec_next=boolean_operator_processing("NOT",term_docs_incid_ma
                else :
                    inc_vec_prev=boolean_operator_processing("NOT",term_docs_incid_ma
                    result=inc_vec_prev
                has_not_operation=False
            elif has_previous_term:
                inc_vec_next=term_docs_incid_matrix[term]
            else :
                inc_vec_prev=term_docs_incid_matrix[term]
                result= inc_vec_prev
                has_previous_term=True

        elif term=="NOT":
          has_not_operation=True
        else :
          bitwiseop=term
        if len(inc_vec_next)!= 0 :
            result = boolean_operator_processing(bitwiseop,inc_vec_prev,inc_vec_next)
            inc_vec_prev=result
            has_previous_term=True
            inc_vec_next= []
    for i,doc in zip(result,collection):
      result_set[doc]=i
    return result_set
```

```
collection
```

```
{'doc0': 'Welcom to our first lab session',
 'doc1': 'In this lab, you will lean more about Term Document Incidence Matrix',
 'doc2': 'You will also learn how to visualize it',
 'doc3': 'Hope you enjoy learning it'}
```

```
query1 = " Document OR lab OR learn"
print("query1 boolean retrieval ",boolean_retrieval(query1,collection))
```

```
query1 boolean retrieval  {'doc0': 1, 'doc1': 1, 'doc2': 1, 'doc3': 0}
```

```
v1=term_incidence_vector("Document",term_docs_incid_matrix)
v1
```

```
[0, 1, 0, 0]
```

```
v2=term_incidence_vector("lab",term_docs_incid_matrix)
v2
```

```
[1, 0, 0, 0]
```

```
Qpart1=boolean_operator_processing("OR",v1,v2)
Qpart1
```

```
[1, 1, 0, 0]
```

```
v3=term_incidence_vector("learn",term_docs_incid_matrix)
v3
```

```
[0, 0, 1, 0]
```

```
boolean_operator_processing("OR",Qpart1,v3)
```

```
[1, 1, 1, 0]
```

```
collection
```

```
{'doc0': 'Welcom to our first lab session',
 'doc1': 'In this lab, you will lean more about Term Document Incidence Matrix',
 'doc2': 'You will also learn how to visualize it',
 'doc3': 'Hope you enjoy learning it'}
```

```
query2 = "Lean AND Matrix AND NOT lab"
print("query2 boolean retrieval ",boolean_retrieval(query2,collection))
```

```
query2 boolean retrieval  {'doc0': 0, 'doc1': 1, 'doc2': 0, 'doc3': 0}
```