

```

import numpy as np
import pandas as pd

def check_parenthesis(s):
    l=[]
    for i in s:
        if i=='(':
            l.append(i)
        elif i==')' and len(l)>0:
            l.pop()
        else:
            continue
    if len(l)==0:
        return True
    else:
        return False

def Precedence(a):
    if a.lower()=='and': return 1
    elif a.lower()=='or': return 2
    elif a.lower()=='not': return 3
    else: return 0

def Operator(a):
    if a.lower() in ['(', ')', 'and', 'or', 'not']: return True
    else: return False

def infix_to_postfix(s):
    operators=[]
    operands=[]
    for i in s:
        if i=='(': operators.append(i)
        elif i==')':
            while(len(operators)!=0 and operators[-1]!='('):
                operands.append(operators.pop())
            operators.pop()
        elif not Operator(i):
            operands.append(i)
        else:
            while(len(operators)!=0 and
Precedence(i)<=Precedence(operators[-1])):
                operands.append(operators.pop())
            operators.append(i)
    while(len(operators)!=0):
        operands.append(operators.pop())
    return operands

def queryPrint(r):
    if len(r)!=0:
        print('The Documents related to the given query are:')

```

```

        for i in r:
            print(i)
    else:
        print('No relevant documents found')

def AND_OF(a,b,d,docTerms):
    if type(a) is str:
        if a in docTerms:
            l1=d[a]
        else:
            return []
    else:
        l1=a
    if type(b) is str:
        if b in docTerms:
            l2=d[b]
        else:
            return []
    else:
        l2=b
    r=[]
    for i in l1:
        if i in l2:
            r.append(i)
    return r

def OR_OF(a,b,d,docTerms):
    if type(a) is str:
        if a in docTerms:
            l1=d[a]
        else:
            return []
    else:
        l1=a
    if type(b) is str:
        if b in docTerms:
            l2=d[b]
        else:
            l2=[]
    else:
        l2=b
    for i in l1:
        if i not in l2:
            l2.append(i)
    return l2

def NOT_OF(a,d,dn,docTerms):
    if type(a) is str:
        if a in docTerms:
            l=d[a]

```

```

        else:
            return []
    else:
        l=a
        r=[]
        for i in dn:
            if i not in l:
                r.append(i)
        return r

def queryEval(s,d,dn,dt):
    if(check_parenthesis(s)):
        s=infix_to_postfix(s)
        operands=[]
        for i in s:
            if not Operator(i):
                operands.append(i)
            elif i.lower()=='and':
                b=operands.pop()
                res=AND_OF(operands.pop(),b,d,dt)
                operands.append(res)
            elif i.lower()=='or':
                b=operands.pop()
                res=OR_OF(operands.pop(),b,d,dt)
                operands.append(res)
            else:
                res=NOT_OF(operands.pop(),d,dn,dt)
                operands.append(res)
        res=operands.pop()
        queryPrint(res)
    else:
        print('Invalid no of Parenthesis in the given query.')
        return

docNames=['DOC1','DOC2','DOC3','DOC4']
docs=[]
for i in docNames:
    try:
        with open(f'{i}.txt') as file:
            data=file.readlines()
            docs.extend(data)
    except:
        print(f'There is a Problem opening the file {i}.txt')

for i in range(len(docNames)):
    print(f'{docNames[i]}:\n{docs[i]}')

docTerms=[]
for i in docs:
    for j in i.lower().strip().split():

```

```

        if j not in docTerms:
            docTerms.append(j)
docTerms.sort()

dic={}
for i in range(len(docTerms)):
    l=[]
    for j in range(len(docs)):
        if docTerms[i].lower() in docs[j].lower():
            l.append(docNames[j])
    dic[docTerms[i]]=l

print('\nTerms\t Docs')
print('-----')
for i in dic:
    print(i, '-----', dic[i])

```

```

s=input('\nEnter your query:').split()
c=0
for i in s:
    c=c+1
if c==0:
    print('Empty query is given')
else:
    queryEval(s,dic,docNames,docTerms)
#look and even and slightly
 #( established and many ) or not lorem
 #( long and established ) or shreyas
#systems and powered and ai

```

DOC1:

It is a long established fact that a reader will be distracted by the readable content of a page have suffered when looking at its layout

DOC2:

Many desktop publishing packages and web page editors have suffered alteration now use Lorem Ipsum as their default model text, and a search for lorem ipsum will uncover many web sites still in their infancy

DOC3:

There are many variations of passages of Lorem Ipsum available but the majority have suffered alteration in some form by injected humour or randomised words which donot look even slightly believable

DOC4:

But I must explain to you how all this mistaken idea of denouncing pleasure and praising pain was born and I will give you a complete account of the system and expound the actual teachings of the great explorer of the truth the master builder of human happiness

Terms Docs

a ----- ['DOC1', 'DOC2', 'DOC3', 'DOC4']
account ----- ['DOC4']
actual ----- ['DOC4']
all ----- ['DOC4']
alteration ----- ['DOC2', 'DOC3']
and ----- ['DOC2', 'DOC3', 'DOC4']
are ----- ['DOC3']
as ----- ['DOC2', 'DOC3', 'DOC4']
at ----- ['DOC1', 'DOC2', 'DOC3', 'DOC4']
available ----- ['DOC3']
be ----- ['DOC1', 'DOC3']
believable ----- ['DOC3']
born ----- ['DOC4']
builder ----- ['DOC4']
but ----- ['DOC3', 'DOC4']
by ----- ['DOC1', 'DOC3']
complete ----- ['DOC4']
content ----- ['DOC1']
default ----- ['DOC2']
denouncing ----- ['DOC4']
desktop ----- ['DOC2']
distracted ----- ['DOC1']
donot ----- ['DOC3']
editors ----- ['DOC2']
established ----- ['DOC1']
even ----- ['DOC3']
explain ----- ['DOC4']
explorer ----- ['DOC4']
expound ----- ['DOC4']
fact ----- ['DOC1']
for ----- ['DOC2', 'DOC3']
form ----- ['DOC3']
give ----- ['DOC4']
great ----- ['DOC4']
happiness ----- ['DOC4']
have ----- ['DOC1', 'DOC2', 'DOC3']
how ----- ['DOC4']
human ----- ['DOC4']
humour ----- ['DOC3']
i ----- ['DOC1', 'DOC2', 'DOC3', 'DOC4']
idea ----- ['DOC4']
in ----- ['DOC1', 'DOC2', 'DOC3', 'DOC4']
infancy ----- ['DOC2']
injected ----- ['DOC3']
ipsum ----- ['DOC2', 'DOC3']
is ----- ['DOC1', 'DOC2', 'DOC3', 'DOC4']
it ----- ['DOC1', 'DOC2', 'DOC3']
its ----- ['DOC1']
layout ----- ['DOC1']

long ----- ['DOC1']
look ----- ['DOC1', 'DOC3']
looking ----- ['DOC1']
lorem ----- ['DOC2', 'DOC3']
majority ----- ['DOC3']
many ----- ['DOC2', 'DOC3']
master ----- ['DOC4']
mistaken ----- ['DOC4']
model ----- ['DOC2']
must ----- ['DOC4']
now ----- ['DOC2']
of ----- ['DOC1', 'DOC3', 'DOC4']
or ----- ['DOC2', 'DOC3', 'DOC4']
packages ----- ['DOC2']
page ----- ['DOC1', 'DOC2']
pain ----- ['DOC4']
passages ----- ['DOC3']
pleasure ----- ['DOC4']
praising ----- ['DOC4']
publishing ----- ['DOC2']
randomised ----- ['DOC3']
readable ----- ['DOC1']
reader ----- ['DOC1']
search ----- ['DOC2']
sites ----- ['DOC2']
slightly ----- ['DOC3']
some ----- ['DOC3']
still ----- ['DOC2']
suffered ----- ['DOC1', 'DOC2', 'DOC3']
system ----- ['DOC4']
teachings ----- ['DOC4']
text, ----- ['DOC2']
that ----- ['DOC1']
the ----- ['DOC1', 'DOC2', 'DOC3', 'DOC4']
their ----- ['DOC2']
there ----- ['DOC3']
this ----- ['DOC4']
to ----- ['DOC2', 'DOC4']
truth ----- ['DOC4']
uncover ----- ['DOC2']
use ----- ['DOC2']
variations ----- ['DOC3']
was ----- ['DOC4']
web ----- ['DOC2']
when ----- ['DOC1']
which ----- ['DOC3']
will ----- ['DOC1', 'DOC2', 'DOC4']
words ----- ['DOC3']
you ----- ['DOC1', 'DOC4']

Enter your query:(established and many) or not lorem

The Documents related to the given query are:

DOC1

DOC4