boolean query processing

```
doc1="HIIi!! hello how are you@."
doc2="hello ! how are you feeling dear RAMANA AKKIREDDY."
doc3="heyy buddy VIJAYA$!"
l=[doc1.lower(),doc2.lower(),doc3.lower()]
print(l)
#removing the different symbols and white spaces:
punc = '''!()-[]{};:'"\, <>./?@#$%^&*_~'''
for ele in doc1:
    if ele in punc:
        doc1 = doc1.replace(ele, " ")
print(doc1)
HIIi   hello how are you
for i in range(0,len(l)):
  for ele in l[i]:
    if ele in punc:
      l[i]=l[i].replace(ele," ")
print(l)
['hiii   hello how are you  ', 'hello   how are you feeling dear ramana akkireddy ', 'heyy buddy
vijaya  ']
r=[]
# Remove multiple empty spaces from string List
# Using loop + strip()
for i in range(0,len(l)):
  z=l[i].split(" ")
  res = []
  for ele in z:
     if ele.strip():
        res.append(ele)
  r.extend(res)
print(r)
f_stop=r
#removing the stop words:
for i in f_stop:
    if i in stopwords:
      f_stop.remove(i)
print(f_stop)
#remove duplicates and start indexing
final=set(f_stop)
d={}
for i in final:
  d.update({i:[]})
for i in d.keys():
  for j in range(0,len(l)):
      if i in l[j]:
        d[i].append(j)
print(d)
['hiii', 'hello', 'how', 'are', 'you', 'hello', 'how', 'are', 'you', 'feeling', 'dear', 'ramana', 'akkireddy',
'heyy', 'buddy', 'vijaya']
```

['hiii', 'hello', 'how', 'you', 'hello', 'how', 'you', 'feeling', 'dear', 'ramana', 'akkireddy', 'heyy', 'buddy', 'vijaya']
{'dear': [1], 'akkireddy': [1], 'hiii': [0], 'hello': [0, 1], 'you': [0, 1], 'how': [0, 1], 'heyy': [2], 'buddy': [2], 'ramana': [1], 'feeling': [1], 'vijaya': [2]}

```
def intersection_words(s1,s2):
            p1=s1
            p2=s2
            l=[]
            i=0
            j=0
            if(len(p1)==0 and len(p2)!=0):
                return p2
            elif(len(p2)==0 and len(p1)!=0):
                return p1
            while(i<len(p1) and j<len(p2)):
                    if(p1[i]==p2[j]):
                            l.append(p1[i])
                            i=i+1
                            j=j+1
                    elif(p1[i]<p2[j]):
                             i=i+1
                    elif(p1[i]>p2[j]):
                             j=j+1
            return l
def or_operation(s1,s2):
    if(len(s1)==0):
        return s2
    elif(len(s2)==0):
        return s1
    else:
        k=set(s1)
        l=set(s2)
        t=k.union(l)
        return list(t)
#conversion of infix to postfix expression of given queries
s1=input("enter the query")
s=s1.lower()
l=s.split(" ")
print(l)
post=""
p_fix=[]
res=[]
for i in range(0,len(l)):
   if(l[i]=="not"):
        p_fix.append(l[i])
   elif(l[i]=="and"):
        if len(p_fix)==0:
```

```
            p_fix.append(l[i])
        else:
            while(len(p_fix)!=0 and p_fix[len(p_fix)-1]!="or"):
                if p_fix[len(p_fix)-1]=='not' or p_fix[len(p_fix)-1]=='and' :
                    z=p_fix.pop()
                    res.append(z)
            p_fix.append(l[i])
    elif(l[i]=="or"):
        if len(p_fix)==0:
            p_fix.append(l[i])
        else:
            while(len(p_fix)!=0):
                if p_fix[len(p_fix)-1]=='or' or p_fix[len(p_fix)-1]=='and' or p_fix[len(p_fix)-1]=="not":
                    z=p_fix.pop()
                    res.append(z)
            p_fix.append(l[i])
    else:
     post=post+l[i]
     res.append(l[i])
    #print(res)
    #print(p_fix)
while(len(p_fix)!=0):
  k1=p_fix.pop()
  res.append(k1)
print(res)
#print(post)
enter the querynot vijaya and not buddy
['not', 'vijaya', 'and', 'not', 'buddy']
['vijaya', 'not', 'buddy', 'not', 'and']

#implementing not:
def not_operation(ls):
    k=[ i  for i in range(0,len(l))]
    d=set(k)-set(ls)
    return list(d)#list of all documents

#print(res)
z=[]
i=0
while(i<len(res)):
  if(res[i]=="not"):
     r1=z.pop()
     sp=not_operation(r1)
     z.append(sp)
  elif(res[i]=='and'):
        #print(res[i])
        r1=z.pop()
        #print(r1)
        r2=z.pop()
        #print(r2)
        k=intersection_words(r1,r2)
```

```
        #print(k)
        z.append(k)
    elif(res[i]=='or'):
        r1=z.pop()
        r2=z.pop()
        k=or_operation(r1,r2)
        z.append(k)
    else:
        print(res[i])
        if(res[i] in d.keys()):
            #print(res[i])
            z.append(d[res[i]])
            #print(z)
        else:
            z.append([])
    i=i+1

print(z)
[[0, 1, 3, 4]]
```