Lab 4 report

I took the following steps to calculate the offer weight for each term and to calculate the extended queries:

1. Take a query from the command line and get 10 documents resulting from the search.
2. For each document:
    a. For each term in a document:
        i. Calculate the number of documents this term occurs in
        ii. Calculate the offer weight of the term
        iii. Add the key/value pair of offer weight/term to a file
    b. Take the file of key/value pairs and sort it in descending order, removing duplicates. (duplicates are terms with identical offer weight ie terms which are the same after stemming)
3. Get the top 10 terms from the above sorted file of offer weights, create a new query.
4. Get the 10 document ids returned by searching with the new query.

#examples from my bash script
#calculating offer weight:
```
function getOfferWeight {
    RI=$(bc <<< "scale=2; $4")
    N=$(bc <<< "scale=2; $2")
    NI=$(bc <<< "scale=2; $1")
    R=$(bc <<< "scale=2; $3")
    rw=$(bc <<< "scale=2; (($RI + 0.5)*($N - $NI - $R + $RI +
0.5))/(($NI - $RI + 0.5)*($R - $RI + 0.5))")
    rwl=$(echo $rw | bc -l)
    ow=$(bc <<< "scale=2; $rwl * $RI")

    echo "$ow"
}
```

#finding number of documents a term occurs in
```
function getNumRelTermDocs {

    RI=$(bc <<< "scale=2; 0")
    while read filename
    do
        term=$(sed 's_-__g' <<< $2)
        if grep -Fxq "$term" $filename
        then
            # code if found
            RI=$(bc <<< "scale=2; $RI + 1")
        fi
    done < "$1"
    echo $RI
}
```

#sorting terms by offer weight and removing duplicates
```
sort -u -n -r "all-terms.txt" > "sorted.txt"
```

#preparing each term from a document
```
while read newurl
do
    ID=$(sed 's_.*=\([^$]*\)$_\1_' <<< $newurl)
    #echo "id: $ID"
    #echo "$URL/$newurl"
    wget --no-http-keep-alive -q -O- "$URL/$newurl" >
"$ID-output.txt"
    sed '/body>/d' "$ID-output.txt" > "$ID-results.txt"
    rm "$ID-output.txt"
    java WordCounter -1 "$ID-results.txt" > "$ID-terms.txt"
    echo "$ID-terms.txt" >> term-docs.txt

done < "output.txt"
```

Lab 05 report
The original queries I used for this section were:
Topic 301: International organized crime
Topic 302: Poliomyelitis and post-polio
Topic 303: Hubble telescope achievements

The expanded query terms I used for each topic were as follows:
Topic 301: vaculik, pm1506102194, wolny, principato, raisch, lesch, malbakhov, malbakhova, trajanov, omerta

Topic 302: polio, gohil, poliomyelitis, yamini, chhea, nathani, opv, paralytic, bhagat, phichit

Topic 303: hubble, telescope, waelkens, vlt, hofstadt, nasa, glaswerke, europan, silla, liftoff

The values for P5 and P10 respectively which I achieved for each topic were as follows:
Topic 301: 0.4000, 0.4000
Topic 302: 0.4000, 0.5000
Topic 303: 0.0000, 0.1000

Unfortunately, I could not find any variation in my results when I varied the number of documents considered relevant or the number of terms which I used for query expansion. I think this may have been due to how I generated my file to pass into trec_eval, but I am not sure. Below is the example output of one of my input files I used while trying various values:
//begin output

```
303 Q0  FT921-7107  1         13.860487      BM25.1.2.0.75
303 Q0  LA071090-0047   1   13.860487      BM25.1.2.0.75
303 Q0  LA050390-0109   1   13.860487      BM25.1.2.0.75
303 Q0  LA041990-0151   1   13.860487      BM25.1.2.0.75

//end output
```

I tried 3 values for documents to be relevant: 2, 4 and 6 documents (and of course 10 documents). For each of these I also tried testing query expansion with 5, 10, 15 and 20 terms. I saw no difference in my results, and due the the extremely time consuming process of testing each combination I chose to try this with Topic 303 only, and with no more combinations than mentioned above.