Question 1:

(1) Query plan:

→ ***Round 1:***
Key : paperid
Map : emit
Reduce : (1) GroupBy,

→ ***Round 2:***
Key : authid
Map : emit (pair of auth id)
Reduce : (1) GroupBy, (2) count

→ ***Round 2:***
Key : authid
Map : (1) local sort, (2) top k troncate
Reduce : (1) Sort all in a single reducer, (2) top k troncate

(2) MapReduce description:

→ ***Job 1:***
Goal : Return all the authors for each papers
Map : emit tuple <paperid, authid>
Reduce : For each paperid store authors in a string “authID1, authID2, authID3, …” (tried in an arrayWrittable with no succes)
Write the result in a temporary file.

→ ***Job 2:***
Goal : For each authors count his numbers of coauthors
Map : emit all pairs <authid, authid>
Reduce : For each author, store his unique coauthors in a Hashset and count his length n.
Write the tuples <authid, n> in a temporary file.

→ ***Job 3:***
Goal : Find the K authors
Map : Perform local sorts by storing the tuples <authid, n> in a treemap in a decreasing order (the key being n).
Troncate top K.
Reduce : Do a global sort in a single reducer
Write the tuples <authid, n> in a final file .

(3) Instructions :

→ ***Get Data file:***
Copy the file paperauths.tsv in hdfs

→ ***Compile and JAR file:***
Compile from source Q1.java

→ ***Run application:***
hadoop jar Q1.jar Q1 /inputPath/ /outputPath/ /top_k (top k is an integer)

(4) Output file (k = 5):
fzapfack/A5/Q1

Question 2:

(1) Query plan:

→ ***Round 1:***
Key : paperid
Map : emit
Reduce : (1) GroupBy,

→ ***Round 2:***
Key : authid-authid
Map : emit (pair of authid, 1)
Reduce : (3) count

→ ***Round 3:***
Key : authid
Map : emit authid , (coauthid,n)
Reduce : (1) Sort (2) top k troncate

(2) MapReduce description:

→ ***Job 1:***
Goal : Return all the authors for each papers
Map : emit tuple <paperid, authid>
Reduce : For each paperid store authors in a string “authID1, authID2, authID3, …” (tried in an arrayWrittable with no succes)
Write the result in a temporary file.

→ ***Job 2:***
Goal : Count the number of time each tuple appear together
Map : emit all pairs each pairs of coauthors (authid-coauthid,1)
Reduce : Count the number of time n each pairs (authid-coauthid) appears
Write the tuples <authid-coauthid,n> in a temporary file.

→ ***Job 3:***

Goal : Find the K coauthors for each author

Map : take a line <authid-coauthid,n> and emit <authid, coauthid-n> and <coauthid, authid-n>

Reduce : For each author, store his tuple <coauthid,n> in a treemap by n
Troncate the top k
Write the tresult in a final file .

(3) Instructions :

→ ***Get Data file:***

Copy the file paperauths.tsv in hdfs

→ ***Compile and JAR file:***

Compile from source Q2.java

→ ***Run application:***

hadoop jar Q2.jar Q2 /inputPath/ /outputPath/ /top_k (top k is an integer)

(4) Output file (k = 3):

fzapfack/A5/Q2