```
## this code reads k datasets that we are using to compare their medians
## calculates the test statistic T for KW One way ANOVA test
## does NOT compute p value

## this code assumes there are no ties
## if this assumption is not true, this code will produce incorrect value of
 test statistic


## keep reading datasets and add to a data list
data_list = list()

data_list[[1]] = as.matrix(read.table("folderpath/filename01.extension",
 header = T or F))

data_list[[2]] = as.matrix(read.table("folderpath/filename02.extension",
 header = T or F))

data_list[[3]] = as.matrix(read.table("folderpath/filename03.extension",
 header = T or F))

## and so on...

k = length(data_list)

print(paste("Number of datasets = k = ",k,sep=""))

## Sample size determination for each dataset

sample_size_vector = array(0,k)
for (i in 1:k)
{
sample_size_vector[i] = length(data_list[[i]])
print(paste("Sample size for dataset from Population ", i, " is = n", i," = ",
 sample_size_vector[i], sep=""))
}

## now construct the table to compute the test statistics

## now pool the data
data_pooled =  unlist(data_list)

## initiate construction of the indicator column (2nd column of the table)
## since we have not sorted the pooled data yet,
##first n1 positions correspond to dataset 01, allocated an antry = 1
## next n2 positions correspond to dataset 02, allocated an antry = 2
## next n3 positions correspond to dataset 03, allocated an antry = 3
## and so on...
unsorted_indicator = rep(1:k, sample_size_vector)

## now sort the pooled data from smallest to largest
```

```
data_pooled_sorted = sort(data_pooled)

##  rearrange the indicator column depending on the order of the pooled data
sorted_indicator = unsorted_indicator[order(data_pooled)]

## the rank column has all ranks from 1 to (n1+n2+...+nk)
all_ranks = 1:sum(sample_size_vector)

## combine the columns
output = data.frame( sorted_values = data_pooled_sorted, population_indicator
 = sorted_indicator, all_ranks = all_ranks )

## print it

print("-----------------------------------------------------")
print(output,row.names=F)
print("-----------------------------------------------------")

## Now fill in the fourth column
R_vector = array(0,k)

for (i in 1:k)
{
  R_vector[i] = sum(all_ranks[(sorted_indicator== i)])
  print(paste("Sum of ranks for dataset ",i," = R",i," = ",
   R_vector[i],sep=""))
}

## compute the test statistic

n = sum(sample_size_vector)

print(paste("Combined sample size = n = ", n,sep=""))

T = (12/(n*(n+1)))*(sum(R_vector^2/sample_size_vector)) - 3*(n+1)

print(paste("Test Statistic T = ", round(T,2), sep=""))

## Now use this value of T to find range for the p-value
## using chi-square table, row corresponds to df = (k -1)
```