```
##############################################################################
## Lepage Test                                                              ##
##                                                                          ##
## Author:  Andreas Schulz                                                  ##
## slightly modified by Markus Neuhäuser (neuhaeuser@rheinahrcampus.de)     ##
## Last modified:   14/04/2009                                              ##
##############################################################################



##############################################################################
##############################################################################
LTperm <- function(X1, X2, S=10000 ,exact=TRUE){
##############################################################################
##############################################################################



##############################################################################
## Calculation of the test statistic
Pvall <- function(X1, X2, N1, N2, N, ST=0, EAB , VAB){


Z=c(X1,X2);
RZ=rank(Z, na.last = TRUE, ties.method ="average");
RZS=sort(RZ[1:N1]);



#Ansari-Bradley (AB) calculation
SV=sum(abs(RZS - 0.5*(N+1)));
AB=0.5*N1*(N+1) - SV;



#Wilcoxon (W) Calculation
Q=(N1*N2*(N+1)-( ST/(N*(N-1)) )  )/12;
if (Q==0){
W=0;
} else
{
Wr=N1*(N+1)/2;
W=(sum(RZS) - Wr)/sqrt(Q);
}

L=W^2   +   ((AB-EAB)/sqrt(VAB))^2;            # test statistic


Out=L;                                         # output
}
##############################################################################


##############################################################################
##  permutation test
twert=0;
Z=c(X1,X2);                                        # combined samples
N1=length(X1);                                     # sample sizes
N2=length(X2);
N=N1+N2;
mod=(N-floor(N/2)*2);
```

```
if (mod==0){
Scores=c(1:(N/2),(N/2):1);
} else
{
Scores=c(1:((N-1)/2), (N+1)/2 ,((N-1)/2):1);
}



# Calculation of the variance in the presence of ties
Z=sort(Z);

gABk=0;
k=1;
ties=0;
tk=1;
for (i in 1:(N-1)) {
if (Z[i]==Z[i+1]) {
k=k+1;
gABk[tk]=gABk[tk]+Scores[i];
if (i==(N-1)){
ties[tk]=k;
gABk[tk]=gABk[tk]+Scores[i+1];

}
}
if (Z[i]!=Z[i+1]) {
ties[tk]=k;
k=1;
gABk[tk]=gABk[tk]+Scores[i];
tk=tk+1;
gABk[tk]=0;
if (i==(N-1)) {
gABk[tk]=gABk[tk]+Scores[i+1];
ties[tk]=1;
}
}
}


Rj=gABk/ties;
ST=(ties-1)*ties*(ties+1);
ST=sum(ST);
EAB=(0.25*N1*((N+1)^2)/N)*mod + (0.25*N1*(N+2))*abs(mod-1);
VAB=( N1*N2*(16*sum(ties*(Rj)^2)-N*(N+2)^2)/(16*N*(N-1)) )*abs(mod-1) +
    ( N1*N2*(16*N*sum(ties*(Rj)^2)-(N+1)^4)/(16*N^2*(N-1)) )*mod;




## if total sample size > 20 perform an approximate permutation test
if(exact==TRUE & N>20){
gg=c("too many permutations for an exact test: N=", choose(N,N1));
print(gg,quote = FALSE);
print("Approximate permutation test will be performed based on randomly selected
permutations." ,quote = FALSE);                                                  ##
exact=FALSE;
}


## all permutations
if(exact==TRUE){
k=N1;
```

```
P=array(0, c(N, 1));

for(i in 1:N)
{
P1=P;
P2=P;
P1[N-i+1,1:dim(P)[2]]=1;
P2[N-i+1,1:dim(P)[2]]=0;
P=array(c(P1, P2), c(N, (2*dim(P)[2])));
}

Sums=apply(P, 2, sum);

kpers=1:length(Sums);
kpers=kpers[Sums==k]
P=P[1:N,  kpers];


Z1=array(Z, c(N, dim(P)[2]));
Z1=Z1[P==1];
perm1=array(Z1, c(k, dim(P)[2]));

Z2=array(Z, c(N, dim(P)[2]));
Z2=Z2[P==0];
perm2=array(Z2, c(N-k, dim(P)[2]));


for(i in 1:dim(perm1)[2])
{
out=Pvall(perm1[  , i], perm2[  , i], N1, N2, N, ST, EAB , VAB);
twert[i]=out;
}
}


## randomly selected permutations
if(exact==FALSE){
for(i in 1:S){
Pe=rep(0, N);
while(sum(Pe)<N1){
r=floor(runif(1, min=1, max=(N+1)));
Pe[r]=1;
}
perm1=Z[Pe==1];
perm2=Z[Pe==0];

out=Pvall(perm1, perm2, N1, N2, N,  ST, EAB , VAB);
twert[i]=out;
}
}

twert=sort(twert);
testimate=Pvall(X1, X2, N1, N2, N,  ST, EAB , VAB);   # Original value of test
statistic


# search for values larger or equal to original value of test statistic
pval=0;
for(i in 1:length(twert))
{
     if(testimate<=twert[i])
     {
        pval=pval+1;
```

```
     }
}

pval=pval/length(twert);                              # p-value


appro=1-pchisq(testimate, 2, ncp=0, lower.tail = TRUE, log.p = FALSE);
##############################################################################


##############################################################################
## definition of output

print("Lepage test",quote = FALSE);
testimate=round(testimate, digits = 6);
gt=c("test statistic L =", testimate);
print(gt,quote = FALSE);

if(exact==FALSE){
gg=c("test based on S randomly selected permutations: S=",S);
print(gg,quote = FALSE);
}
if(exact==TRUE){
gg=c("Exact permutation tests, number of permutations: ",choose(N,N1));
print(gg,quote = FALSE);
}

pval=round(pval, digits = 6)
gg=c("p-value =", pval);
print(gg,quote = FALSE);
print("",quote = FALSE);

appro=round(appro, digits = 6)
gg=c("Asymptotic p-value (based on Chi-square distribution with df=2) =", appro);
print(gg,quote = FALSE);
print("",quote = FALSE);


##############################################################################


# complete output
Allout=pval;
}
##############################################################################
##############################################################################


# data entry
X1=c(14,24,26,98,12,105,85);
X2=c(40,14,18,28,11,39,17,37,52,30,65,35);


# submit test procedure (print “exact=TRUE” for exact permutation test)
LTperm(X1, X2, S=10000, exact=FALSE);
```