#HW02 Solution

##1.

#(a)

v1=sample(c(1,2,3,4),1000,replace=T,prob=c(0.1,0.2,0.4,0.3))

v1

#(b) use random uniform generator, generate a random vector follows a multinomial distribution with probability (0.1, 0.2, 0.4, 0.3)

my\_sample<-function(x,n){

y<-runif(n)

y[y<=0.1]=x[1]

y[y>0.1 & y<=0.3]=x[2]

y[y>0.3 & y<=0.7]=x[3]

y[y>0.7 & y<=1]=x[4]

return(y)

}

v2<-my\_sample(c("A","B","C","D"),1000)

##2. Generate 100 exponentially distributed random variables with rate2, and plot their empirical distribution function.

x<-rexp(100,2) # Plot the empirical distribution function

ordered.x<-sort(x)

plot(ordered.x,(1:100)/100,type='s',ylim=c(0,1),xlab="Sample Quantiles of X",ylab="",main=

"empirical distribution function of Expo(2)")

##3.

logret <- read.table("http://www.ams.sunysb.edu/~pfkuan/Teaching/AMS597/Data/d\_logret\_6stocks.txt", header = T)

#(a) perform a t-test for American Express with the null hypothesis that the mean of its log return is zero.

t.test(logret$AmerExp)

#p-value is 0.8516, which means we fail to reject the null hypothesis that true mean is equal to 0

#(b)

wilcox.test(logret$AmerExp)

#p-value is 0.3225, which means we fail to reject the null hypothesis that true mean is equal to 0

#(c)

var.test(logret$Pfizer,logret$AmerExp)

# p-value=0.03896, the variance is not equal

t.test(logret$Pfizer,logret$AmerExp)

# p-value=0.318, true difference in means is equal to 0

#(d)

var.test(logret$Pfizer,logret$AmerExp)

# p-value=0.03896, the variance is not equal

#(e)

wilcox.test(logret$Pfizer,logret$AmerExp)

#p-value is 0.1662, which means we fail to reject the null hypothesis that true location shift is equal to 0

##4.

my.t.test=function(x,y="default",alternative="two.sided",mu=0){

if(!is.character(y)&&length(y)!=1){ # two sample case

n1=length(x)

n2=length(y)

v1=sd(x)^2

v2=sd(y)^2

if(var.test(x,y)$p.value<=0.05){ # unequal variance

stat=(mean(x)-mean(y))/sqrt(v1/n1+v2/n2)

df=(v1/n1+v2/n2)^2 / ((v1/n1)^2/(n1-1)+(v2/n2)^2/(n2-1))

}

else{ #equal variance

df=n1+n2-2

sp\_square =((n1-1)\*v1+(n2-1)\*v2)/df

stat=(mean(x)-mean(y))/sqrt(sp\_square\*(1/n1+1/n2))

}

}

else{ # one sample

if(y!="default"){

if(length(y)==1&&is.numeric(y)){

mu=y

}

else{

if(is.numeric(alternative)){

mu=alternative

}

alternative=y

}

}

stat=(mean(x)-mu)/(sqrt(sd(x)^2/length(x)))

df=length(x)-1

}

if(alternative=="two.sided"){ # double side

p.value=2\*pt(abs(stat),df,lower.tail=F)

}

else if(alternative=="greater"){ #one side

p.value=pt(stat,df,lower.tail=F)

}

else{

p.value=pt(stat,df,lower.tail=T)

}

return(c(stat=stat,df=df,p.value=p.value))

}

## check the code##

set.seed(123)

x=rnorm(50)

y=rexp(100,2)

my.t.test(x,y,"two.sided",0) #p=0.002948964

set.seed(123)

x=rnorm(50)

y=rexp(100)

my.t.test(x,y,"two.sided",0) #p=1.548638e-08

my.t.test(x,y,"less",0) #p=7.743188e-09

my.t.test(x,y,"greater",0) #p=1.000000

set.seed(123)

x=rnorm(50)

my.t.test(x,"two.sided",0.5) #p=0.0008465263

my.t.test(x,"greater",0.5) #p=0.9995767

my.t.test(x,"less",0.5) #p=0.0004232632

##5.

#(a) derive the formula

# RES = sum((y - beta \* x)^2)

# we want minimum

# d(RES) / d(beta)=0

# we get sum(x \* (y - beta \* x))=0

# beta=sum(x\*y)/sum(x^2)

#(b)

set.seed(123)

x<-rnorm(50)

y<-2\*x+rnorm(50)

# compute beta using formula in (a)

beta=sum(x \* y) / sum(x^2) # beta.hat is 1.970958

plot(x,y,main = "scatterplot with fitted line")

abline(a = 0,b = beta)

#(c)

fit <- lm(y~x-1)

summary(fit)