3.1. Sample R code for generating a 2-parameter (shifted) exponential variate:

```r
#note: if X ~ Exp.2p(lambda,eta), Y = X-eta ~ Exp(lambda)  
#so generate Y and deliver X = Y+eta

set.seed(301)
rexp.2parm = function(n, lambda=1, eta=1) {
  #generate 2-param Exponential
  y = rexp(n, lambda)
  x = y + eta
  return(x)
}
#end function

#generate sample
n = 1000
X = rexp.2parm(n, 1, 2)
#compare quantiles to theory
Q = qexp( ppoints(n), 1 ) + 2
qqplot( Q, X ); abline(0, 1)
```

This produces the following plot:
### 3.4. Sample R code for generating Rayleigh variate:

```r
#note: if V ~ Exp(lambda)
#      then X = sqrt(2*lambda * sigma^2 * V) ~ Rayleigh(sigma)
#so generate V and deliver X = sigma*sqrt(2*lambda*V)

cat("set.seed(304)
rRayleigh = function(n, sigma=1) {
    #generate Rayleigh
    v = rexp(n, 1)
    x = sigma*sqrt(2*v)
    return(x)
} #end function

cat("#generate sample"
    n = 1000
    X = rRayleigh(n, 5) #sigma=5

    #compare mode via histogram
    hist(X, main='')
```

This produces the following plot; notice that the sample mode is at the expected mode of \( \sigma = 5 \):
3.8. Sample R code for generating lognormal variate:

```r
#note: if Y ~ N(mu,sigma^2)
#      then X = exp(Y) ~ LogN(mu,sigma^2) (via p.32)
#so generate Y and deliver X = exp(Y)

set.seed(308)
rlognorm = function(n, mu=0, sigma=1) {
  #generate Lognormal
  y = rnorm(n, mean=mu, sd=sigma)
  x = exp(y)
  return(x)
}

#generate sample
n = 1000
X = rlognorm(n,0,1)

#compare histogram and density
hist( X, prob=T, xlim=c(0,30), ylim=c(0,.75), breaks='Scott',
     ylab='', main='', xlab='' )
par(new=T)
curve( dlnorm(x), xlim=c(0,30), ylim=c(0,.75), ylab='' )

This produces the following plot:
```
```
Sample R code for generating Epanechnikov variate:

```r
rEpanch = function(n) {
    #generate Epanechnikov via trivariate uniforms
    u = matrix(runif(n*3,-1,1),ncol=3)
    x = numeric(n)
    for (i in 1:n) {
        x[i] = ifelse(abs(u[i,3]) >= abs(u[i,2]) && abs(u[i,3]) >= abs(u[i,1]),
                        u[i,2], u[i,3] )
    } #end for loop
    return(x)
} #end function

#generate sample
set.seed(309)
n = 1000
X = rEpanch(n)
#compare histogram
hist( X, prob=T, ylab='', main='', xlab='', breaks='Scott',
     xlim=c(-1,1), ylim=c(0,1) )
par(new=T)
curve(.75*(1-x^2), xlim=c(-1,1), ylim=c(0,1), ylab='')
```

This produces the following plot:
3.11. Sample R code for generating discrete Normal mixture:

```r
set.seed(311)
n = 1000
   p1 = .75
x1 = rnorm(n,0,1)
x2 = rnorm(n,3,1)
u = runif(n)
k = as.integer( u < p1 )
x = k*x1 + (1-k)*x2

#histogram
hist( x, prob=T, ylab='', main='', xlab='', ylim=c(0,.3) )
lines( density(x) )  #kernel density estimator (see Ch. 10)
```

This produces the following plot; note the indication of slight bimodality:

Now vary p1:

```r
par( mfrow=c(3,3) )  #creates 3x3 'matrix' of plots

for (p1 in seq(.1,.9,.1) ){
x1 = rnorm(n,0,1)
x2 = rnorm(n,3,1)
u = runif(n)
k = as.integer( u < p1 )
x = k*x1 + (1-k)*x2
hist( x, prob=T, ylab='', main='', xlab='', xlim=c(-6,6), ylim=c(0,.4),
     main=paste( 'p =',p1 )
lines( density(x) )
    }
#end for loop
```
3.11. (cont’d):

This produces the following multiple plot; bimodality appears strongest in range \(.4 < p < .6:\)

![Multiple histograms showing bimodality strength for different values of p.]

3.12. Sample R code for generating exponential-gamma mixture:

```r
n = 1000
r = 4
beta = 2

set.seed(312)
Lambda = rgamma( n, r, beta )
y = rexp( n, Lambda )

#histogram
hist( y, prob=T, ylab='', main='', breaks = "Scott")
```

This produces the following plot:
3.14. Sample R code for generating multivariate Normal via Choleski factorization:

```r
n = 200
rmvn.Choleski <- function(n, mu, Sigma) {
    # textbook Choleski function (p. 74)
    d <- length(mu)
    Q <- chol(Sigma)
    Z <- matrix(rnorm(n * d), nrow = n, ncol = d)
    X <- Z %*% Q + matrix(mu, n, d, byrow = TRUE)
    X
} # end function

set.seed(314)
mu = 0:2
Sigma = matrix( c(1,-.5,.5,-.5,1,-.5,.5,-.5,1), byrow=T, nrow=3)
X = rmvn.Choleski( n, mu, Sigma )

# scatterplot matrix
pairs( X, pch=19 )
```

This produces the following scatterplot matrix; notice that the sample patterns are consistent with the modeled correlations:
3.14. (cont’d):

For the correlation matrix:

```r
# compare sample correlation matrix.
cor(X)
```

This `cor(X)` command produces the following output; notice that the sample correlation matrix is consistent with the modeled correlations:

```
[,1] [,2] [,3]
[1,] 1.0000000 -0.4412048  0.3941173
[2,] -0.4412048  1.0000000 -0.4642167
[3,]  0.3941173 -0.4642167  1.0000000
```

3.18. Sample R function for generating multivariate Wishart matrix via Bartlett’s (1933) decomposition (one can do this many different ways):

```r
Wishart = function(Sigma, n){
  d = dim(Sigma)[1]
  # lower-tri T matrix
  T = diag( sqrt(rchisq( d, df=seq(n,n+1-d,-1) )) )
  T[lower.tri(T)] = rnorm( n=d*(d-1)/2, 0, 1 )
  # Choleski
  L = t(chol(Sigma))  # chol() gives upper tri.
  M = L %*% T
  return( M %*% t(M) )
}  # end function
```
3.19. Sample R code for generating gambler's Random Walk and producing a plot of the walk (one can do this many different ways):

```r
set.seed(319)
n = 1000
Sn = numeric(n)
i = 1
Sn[i] = 10  # initial stake
S = Sn[1]
while (S>0 & S<20) { i = i+1
  S = S + sample( c(-1,1), size=1 )  # coin flip: p=.5
  Sn[i] = S
}  # end while loop

plot(Sn[1:i], type='o', pch=19, ylim=c(0,20), ylab='', xlab='time' )
abline( h=0, lty=2); abline( h=20, lty=2)  # absorbing barriers
```

This produces the following plot (looks like s/he won this time:)

![Plot of gambler's Random Walk](image)

**time**