HOMEWORK 5 ANSWERS

```
3.5   a)  1.0-pbinom(7,20,.2) =  0.03214266
```

Note that in classical statistics, if a Null Hypothesis is true, then given a random sampling process, the p-value is the probability that the observed data would have occurred by chance. In this instance the Null Hypothesis is:

$$H_0: p = .2$$

And we have 20 Bernoulli trials. By chance the Probability that $X\geq8$ is our p-value. Because the Binomial is the sum of n Bernoulli trials, the $P(X < 7)$ is cumulative probability of X=0, X=1, etc. One minus this is $P(X\geq8)$ which is the probability that we would 8 or more "successes" in 20 trials with a success probability of 0.2.

```
b) pbetat(.2,.5,c(1,4),c(8,12))$post = 0.3410395

c)  partcprob1 <-pbetat(.2,.5,c(.5,2),c(8,12))$post = 0.3900752
    partcprob2 <-pbetat(.2,.5,c(2,8),c(8,12))$post =  0.328591
    partcprob3 <-pbetat(.2,.5,c(8,32),c(8,12))$post = 0.3855337
```

d) I would say "Yes". Guessing correctly 8 out of 20 times when the probability of a guess being correct is .2 is pretty impressive.

4.1 a) (SEE R CODE BELOW)

```
b)  > muquantile
      5%      95%
7.204701 8.461385
   sigmaquantile
      5%      95%
1.367723 2.317931

c)  meanprob75 = 8.988683
    sdprob75 =  0.4371955
```

USING THE NORMPOSTSIM(..) FUNCTION I GET:

```
> muquantilea
      5%      95%
7.216576 8.547621
> sigmaquantilea
      5%      95%
1.364303 2.320563

 meanprob75a = 8.988683
 sdprob75a =  0.4371955
```

R CODE

```
#
# Chapter 4 -- Bayesian Computation With R
#             ESP Problem
#
#  Remove all objects just to be safe
```

```
#
rm(list=ls(all=TRUE))
#
library(LearnBayes)
#
data <- c(9.0, 8.5, 7.0, 8.5, 6.0, 12.5, 6.0, 9.0, 8.5, 7.5, 8.0, 6.0, 9.0, 8.0, 6.0,
7.0, 10.0, 9.0, 7.5, 5.0, 6.5)
#
#  S is the sum of squares
#
S <- sum((data - mean(data))^2)
n <- length(data)
#
#  sigma2 is a 1000 random draw with entries equal to S divided by draws from the
#    chi-square distribution with n-1 degrees of freedom
#
sigma2 <- S/rchisq(1000, n - 1)
#
#  mu is a 1000 random draw from a N(mean, sd) distribution
#
mu <- rnorm(1000, mean = mean(data), sd = sqrt(sigma2)/sqrt(n))
muquantile <- quantile(mu, c(0.05, 0.95))
sigmaquantile <- quantile(sqrt(sigma2), c(0.05, 0.95))
#
prob75 <- mu + 0.647*sqrt(sigma2)
meanprob75 <- mean(prob75)
sdprob75 <- sd(prob75)
#
#  Alternatively, use the normpostsim function in LearnBayes package
#
result <- normpostsim(data, m=1000)
muquantilea <- quantile(result$mu, c(0.05, 0.95))
sigmaquantilea <- quantile(sqrt(result$sigma2), c(0.05, 0.95))
#
prob75a <- mu + 0.647*sqrt(result$sigma2)
meanprob75a <- mean(prob75)
sdprob75a <- sd(prob75)
```