Assignment 4

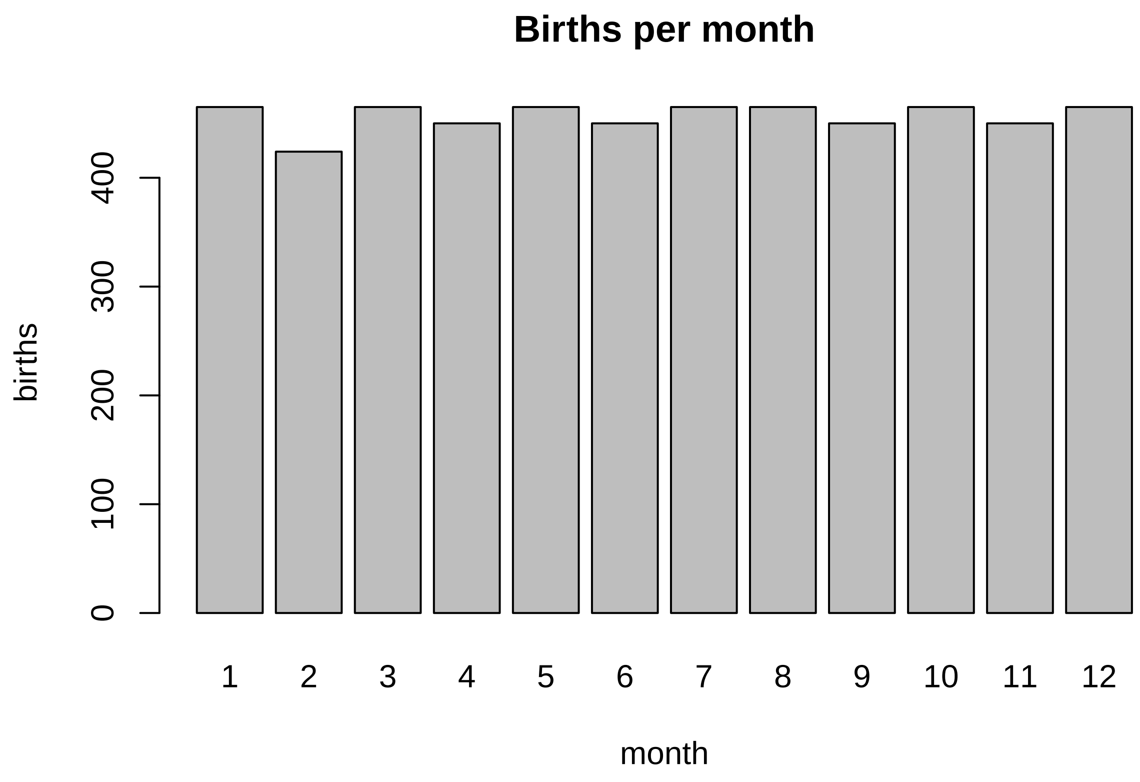
Pierre Visconti

Problem 1

```
USbirths = read.csv("US_births_2000-2014_SSA.csv")
unique(USbirths$month)
```

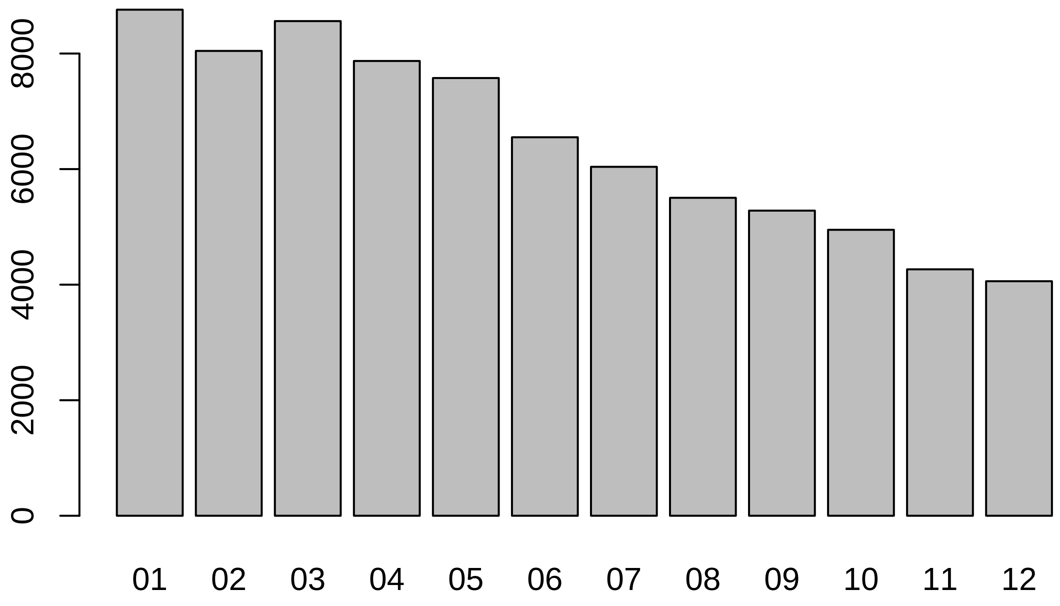
```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 NA
```

```
months = na.omit(USbirths$month)
barplot(table(months), xlab="month", ylab="births", main = "Births per month")
```



Judging by the bar chart it appears that our assumption about birthdays being randomly distributed is mostly correct. There is roughly the same amount of births every month which is what you would expect if it was truly random and there is no correlation between the number of births and certain months.

```
hockey = read.csv("player_dim.csv")
dob = hockey$DATE_OF_BIRTH
# removes NA entries if any exist
if (length(dob[is.na(dob)]) > 0) {
  dob = dob[!is.na(dob)]
}
dob = as.Date(dob, format="%Y-%m-%d")
dob.m = format(dob, "%m")
barplot(table(dob.m))
```



For this dataset, the number of births per month is not the same across each month. I would not call this dataset randomly distributed, there definitely appears to be a correlation between months early on in the year and number of births. I would expect for there to be a higher probability of at least a match compared to the previous example where we assumed that number of births across each month are the same. My reasoning for this is thinking about the extreme case where maybe 80% of the population was born in the same month. For that situation there would be a larger probability of at least a match, so since this dataset falls under a similar situation I would assume that the probability is higher than when evenly distributed.

```
set.seed(215)
dob.md = format(dob, "%m-%d")
count = 0
for (i in 1:10000) {
  indivs = sample(dob.md, 34, replace=T)
  if (length(unique(indivs)) < 34) {
    count = count + 1
  }
}
cat("Hockey players: ", count/ 10000, "\n")
```

```
## Hockey players:  0.8142
```

```
cnt <- 0
for (i in 1:10000){
class <- sample(1:366,34,replace=TRUE)
if(length(unique(class))<34){
cnt <- cnt+1
}
}
cat("Class: ", cnt/10000)
```

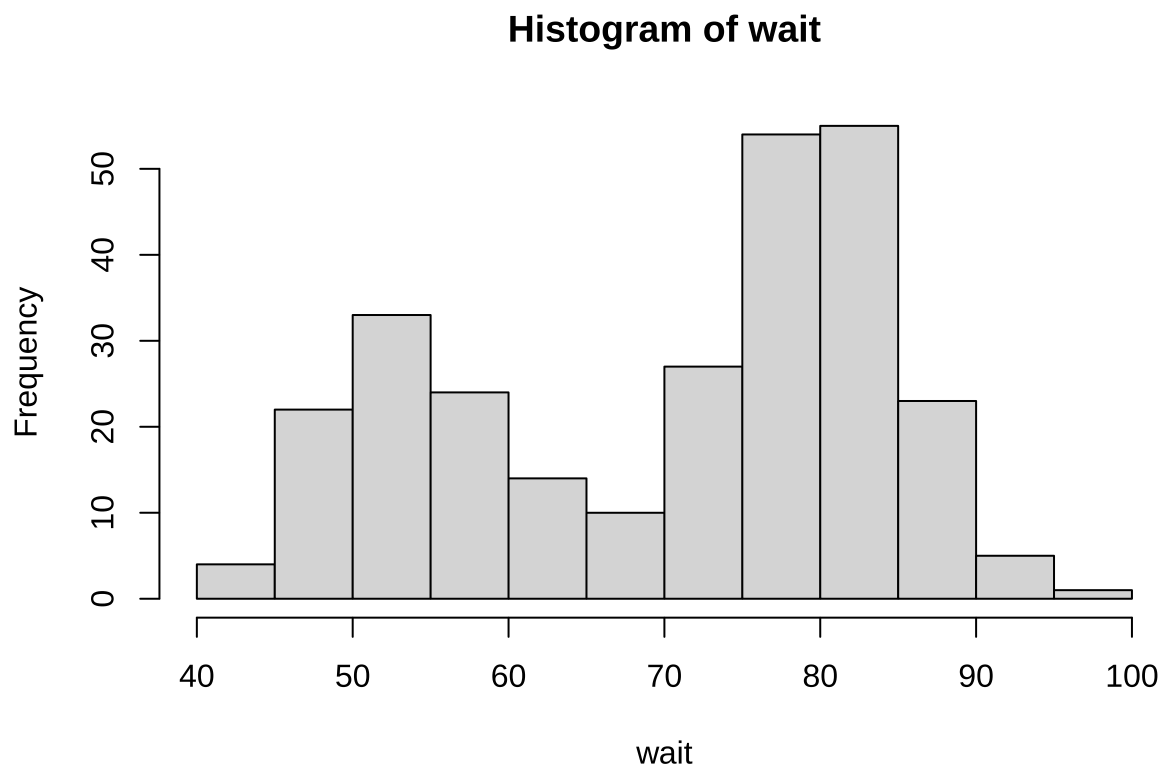
```
## Class:  0.7965
```

My hypothesis that the probability would be higher, appears to be correct.

Maybe people born in the winter like winter sports more. But then you would expect December to also represent a larger number of players yet it has the lowest. It seems very strange that January has the highest and then it just goes down linearly for each proceeding month (most of the time), and then when the year changes it instantly goes back up.

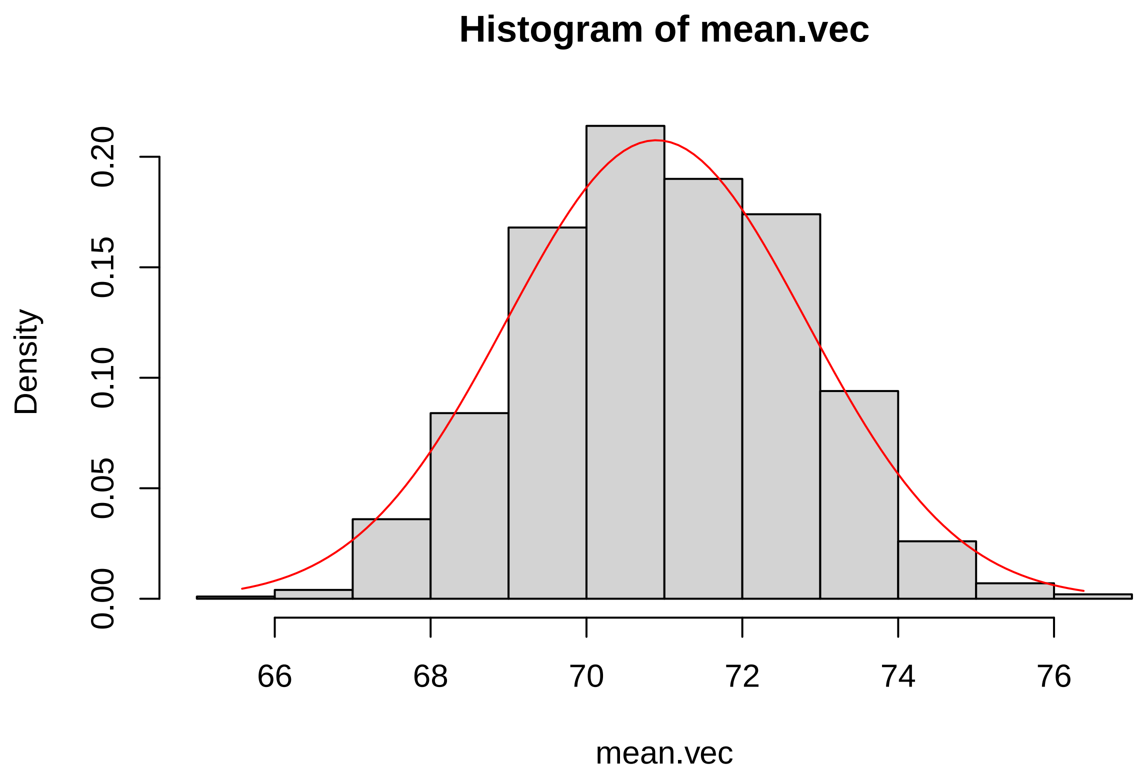
Problem 2

```
data(faithful)
wait = faithful$waiting
hist(wait)
```



The distribution does not look like a normal distribution, it appears to be a bimodial distribution. What we can conclude is that the dataset appears to have two groups, yet with the central limit theorem, as long as the sample size is large enough, it will not matter that this is a bimodial distribution.

```r
# adds sample means to mean.vec
mean.vec = c()
for (i in 1:1000) {
  sample.mean = mean(sample(wait,50))
  mean.vec = append(mean.vec, sample.mean)
}
# finds population mean and std dev
pop.mean = mean(wait)
pop.sd = sd(wait)
# create histogram
x = seq(min(mean.vec), max(mean.vec), by=0.1)
y = dnorm(x,pop.mean,pop.sd/sqrt(50))
hist(mean.vec, freq=F)
lines(x,y, col="red")
```



Yes the histogram looks like the normal distribution.

Problem 3

```
# using conv_unit from measurements package to convert units
#installed.packages("measurements")
library(measurements)
height = hockey$HEIGHT_CM
height = conv_unit(height,"cm","inch")
height = height[!is.na(height)]
```

```
pop.m = mean(height)
samp = sample(height, 250)
samp.m = mean(samp)
samp.sd = 1.96*sd(samp)
conf.int = c(samp.m-samp.sd/sqrt(250), samp.m+samp.sd/sqrt(250))
cat("Confidence interval: ", conf.int, "\n")
```

```
## Confidence interval:  71.43001 72.01881
```

```
cat("True mean: ", pop.m)
```

```
## True mean:  71.64411
```

The true mean does lie within the confidence interval

```r
pop.m = mean(height)
count = 0
for (i in 1:100) {
  samp = sample(height, 250)
  samp.m = mean(samp)
  samp.sd = 1.96*sd(samp)
  conf.int = (c(samp.m-samp.sd/sqrt(250), samp.m+samp.sd/sqrt(250)))
  if (pop.m>=conf.int[1] & pop.m<=conf.int[2]) {
    #print(conf.int)
    count = count + 1
  }
}
count/100
```

```
## [1] 0.96
```

The value changes with each run yet it does appear to be close to the theoretical 0.95 each time. I'm seeing anything from about 0.92 to 0.97.