# Lab2

Christopher Munoz

- **Problem 1a**

  For this problem we are tasked with modelling the probability distribution of the sums of 4 consecutive dice rolls and we will be using the sample size of 100 with the assumption that will provide us with increased precision.
  The problem also states that we earn 1/4th of the sum so that if roll 1,1,1,1, then we will receive a 1 dollar
  **Theory:** I think that the majority of the distribution will land close to or on the median of values or the given average of the sums between 4 and 24(since distribution is symmetric median and average should be the same) divided by 4. The minimum value we can roll is 4 and the maximum is 24, the median being 14, when we divide that by 4 I think we should earn close to
  **Conclusion:** ***3.50$ average per game.***

- **Problem 1b**

  The minimum value of four dice rolls is a 4 and the maximum is a 24 so we use the following R code provided and modify it and we take a sample size of 100 for the most precise measurement(at least that is what my intuition tells me), below is the dice roll code.

```r
sampleSize<-100
numberDice=4

sumOfDice=rep(0,sampleSize)

str=sprintf("Generating %d samples of the experiment   `Find the sum of %d dice':  ",
            sampleSize,numberDice)
cat(str)
```

```
Generating 100 samples of the experiment   `Find the sum of 4 dice':
```

```r
cat("\n")
```

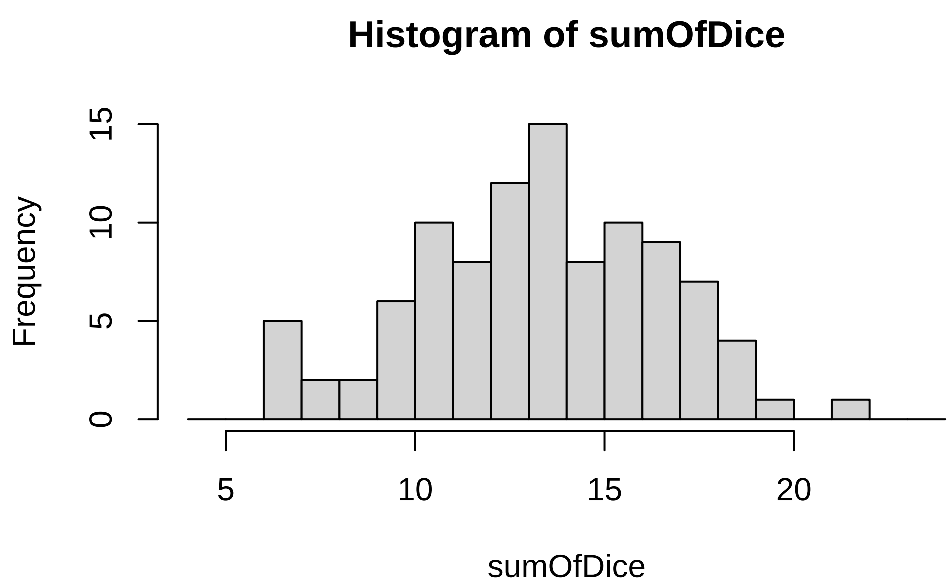
```r
for (ii in 1:sampleSize){
  sumOfDice[ii]=sum(floor(runif(numberDice,1,7)))
  #print(sumOfDice[ii])
}
#This is where we divide our list of values by 4
money <- sumOfDice / 4

if (sampleSize<=100){
  print(sumOfDice)
}
```

```
 [1] 15 14 14 10 13  7 14 12 15 18 18 15 11 17  7 19 16 16 14 11 12 17 18 13 14
[26] 18 12 13  8 16 10 12 17 13 16 11 16 13  7  7  9 17 17 13 13 11 14 10  9 14
[51] 13 13 14 11 11 14 15 19 19 15 17 16 18 13 11 11 13 16 11 16 16 16 14 14 14
[76] 11 15 13 20  7 12 17 15 12 19 10 10 18 22 10 12 14 12 15 14 18 17 14 17  8
```

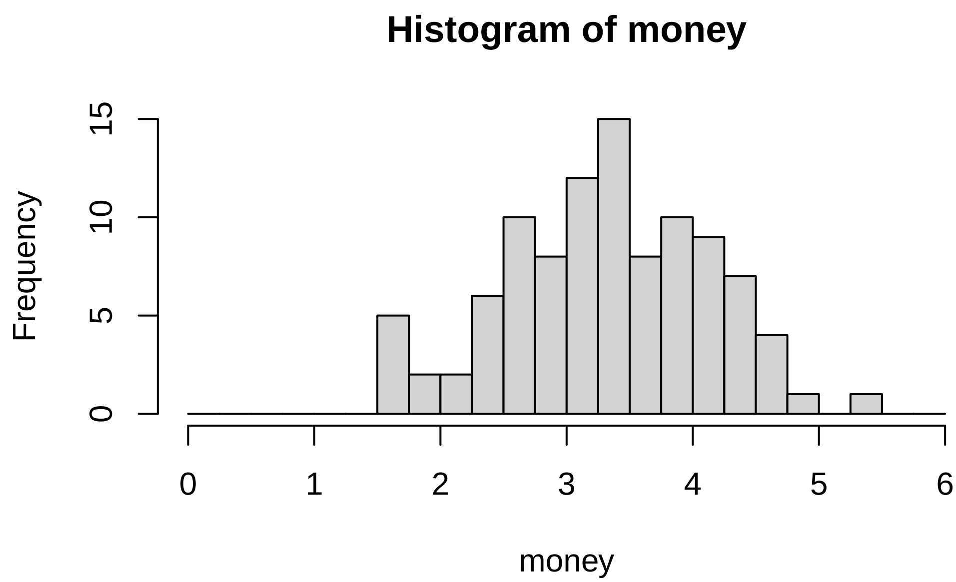
**Below is the histogram of the sums of 4 dice rolls and their distribution.**

```r
hist(sumOfDice,seq(4,24,1),xlim = c(4, 24),freq=T)
```



Dividing our x axis by 4 gives us the distribution of earned money from the dice rolls.

```
hist(money,seq(0,6,0.25), freq=T)
```




```
mean(money)
```

```
[1] 3.445
```

```
sd(money)
```

```
[1] 0.8076184
```

**conclusion:**

***We get a mean of 3.49$ earned with a standard deviation of 0.95(or 1 cent)***

- **problem 1c**

  In this game we are given three white dice and one blue dice, in this case we earn 0.25$ times the sum of the three white dice and we lose 0.50$ times what we roll for the blue

dice. Given 3 dice rolls for white my minimum sum is 3 and our maximum is 18. Our minimum blue roll is 1 and maximum is 6.
**Theory:** A rough model I came up with is *Earned = (SumofWhite).25 - (Blue)*.50*, using my last theory I expect the majority distribution of the sums of whites will land around the median/average between the values of 3 and 18. In this case we find that the value is 10.5. For the blue dice roll there is a 1/6th probability of landing on any side. New formula is Earned = 10.5 * .25 - (Blue) *.50 or Earned = 2.5625 - (Blue) *.50. If enter the values 1 and 6 into our formula for blue then we can get a range for what our expected earnings would be. Our values should fall between (-0.4375, 2.0625), taking the average using the extremes of this range I would assume we would make about 0.8125 or …

**Conclusion:**
***`0.80 cents on average`***

- **Problem 2**
  For this problem we are given 3 binary datasets that we load now

```
load(file="earnings1000.dat")
load(file="earnings100.dat")
load(file="earnings10.dat")
```

We are tasked with finding the probability of obtaining heads given the dataset containing all the earnings. The problem states that the coin is unfair and that we earn 4$ for each Head and lose 3$ for each tail. We will start with using the expected earnings formula and try to logic/algebra our way to a conclusion.

**Theory:** Expected Earnings given the all the information is Earnings = p * 4 + (1-p) * (-3), doing some algebra hackery we get the formula p = (E/N+3)/(4+3). E in this case will be the total earnings and N will be the amount of flips, for example for e1000 there will be 3000 total flips over the 1000 earnings in the dataset.

**Conclusion:**
Here we convert our formula to an R function and extract the mean from the datasets:

```
#Here we find our averages of the sum
avg10 = sum(earnings10)/10
avg100 = sum(earnings100)/300
avg1000 = sum(earnings1000)/3000

#Here is our function

probability <- function(E)
```

```
{
  p <- (E+3) / 7
  return(p)
}
```

And finally our calculations for each probability:

```
prob10 = (probability(avg10))
prob100 = (probability(avg100))
prob1000 = (probability(avg1000))
print(prob10)
```

```
[1] 0.7428571
```

```
print(prob100)
```

```
[1] 0.62
```

```
print(prob1000)
```

```
[1] 0.626
```

Averaging over the 3 values of our 3 probabilities we get

```
totavg = (prob10 + prob100 + prob100)/3
print(totavg)
```

```
[1] 0.6609524
```

***I estimate that p = 0.6609524 given the datasets because our earnings given the coinflips, naively I first assumed we would divide by by just N until I remembered that we perform 3 coinflips, I got values of over 1 initially which made no sense until I remembered we are actually performing 30 coinflips for 10 earnings, 300 for 100 and 3000 for the 1000 sets.***