Q7

(i) Doesn't exist
(ii) X1 → X2
(iii) X2 → X1
(iv) X1 → X2 → X3
(v) SES → WATER → CHOLERA
(vi) ELEVATION ← SES → WATER → CHOLERA

Q1

P(Neither E nor F) = 1-P(E or F) = 1-0.75 = 0.25

Q3

{HHH , HHT, HTH,HTT, THH,THT,TTH,TTT}

Q4

A. At least 2 heads
{HHT , HHH, HTH, THH}
⇨ 4/8 = ½ = 0.5
B. The first 2 tosses are head
{HHH, HHT}
⇨ 2/8 = ¼ = 0.25
C. The last tosses are tail
{HHT, HTT, THT, TTT}
⇨ 4/8 = 0.5

Q2.

$$p^2 + p = 1$$

(i)

$$p^2 + p - 1 = 0$$
$$p = \frac{-1-\sqrt{5}}{2}, \frac{-1+\sqrt{5}}{2}$$
$$\mathrm{p} \simeq 0.618\,,\ \mathrm{p} \simeq -1.618$$

(ii)

$$\phi = \frac{1}{p}$$

$$1 + \frac{1}{\phi} \simeq 1.618$$

And

$$1 + \frac{1}{\phi} \simeq 0.618$$

```
library(tidyverse)
```

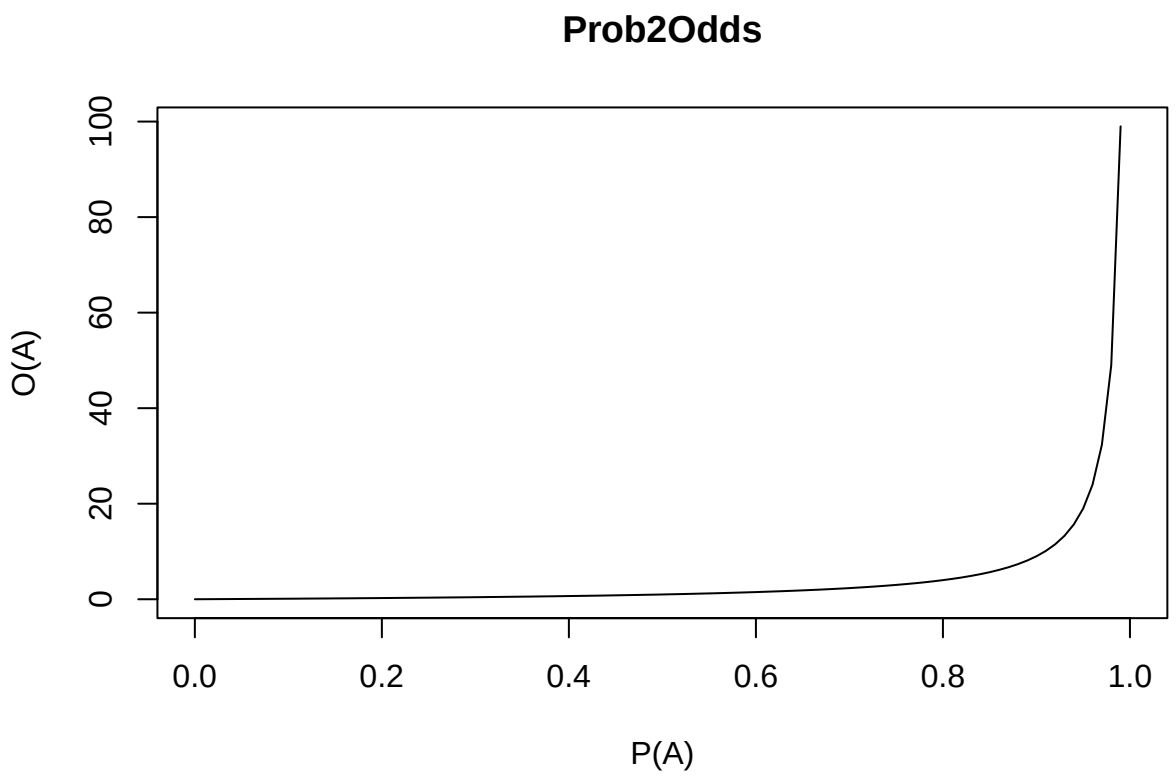
```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.1
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Q5

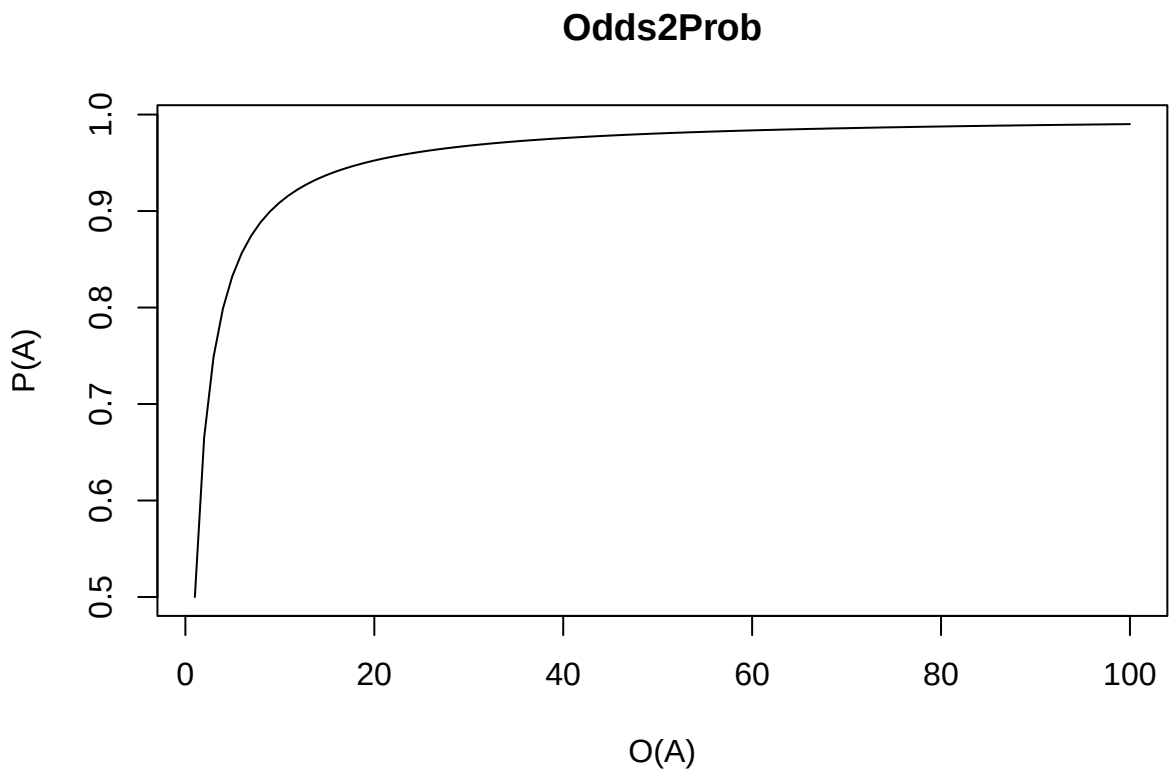
```
probtoodds <- function(p){
  return(p/(1-p))
}

oddstoprob <- function(o){
  return(o/(1+o))
}
```

Q6

```
curve(probtoodds, from=0, to=1,xlab = 'P(A)',ylab = 'O(A)',main="Prob2Odds")
```

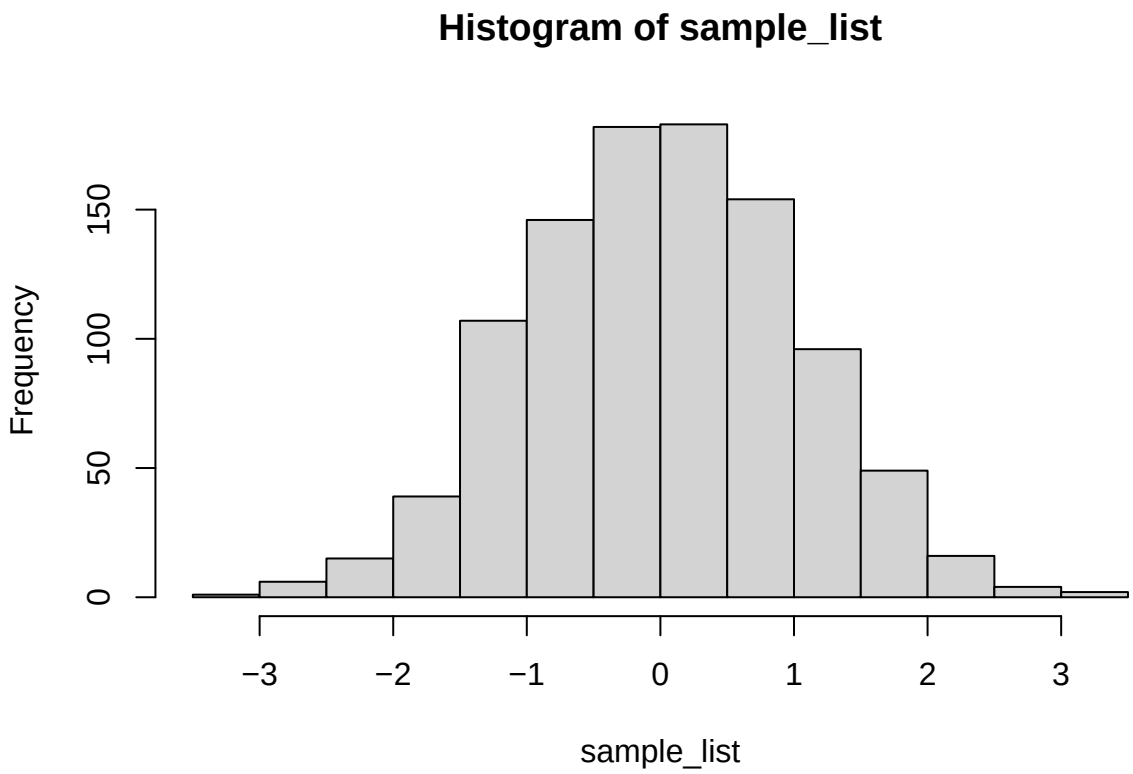


```
curve(oddstoprob, from=1, to=100,ylab = 'P(A)',xlab = 'O(A)',main="Odds2Prob")
```



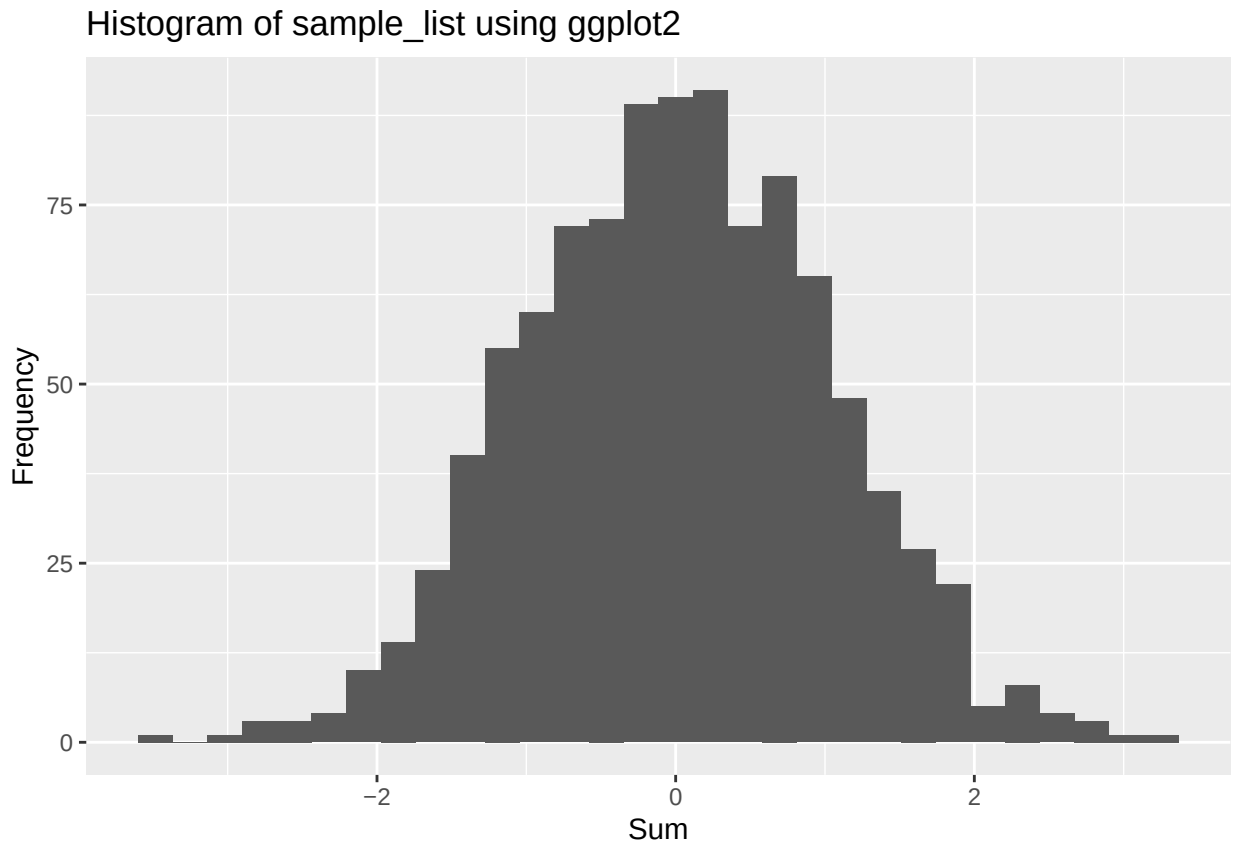
Q8

```
sample_generator <- function(dummy_var){
  results <- runif(12 , min=0,max=1)
  return(sum(results)-6)
}
sample_list <- sapply(1:1000 , sample_generator)
hist(sample_list)
```




```
library(ggplot2)
ggplot(data = data.frame(sample_list), aes(x = sample_list)) +
  geom_histogram() +
  ggtitle("Histogram of sample_list using ggplot2") +
  xlab("Sum") +
  ylab("Frequency")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Q9

```
sample_generator <- function(dummy_var){
  results <- runif(12 , min=0,max=1)
  return(sum(results)-6)
}

normal_sample_gen <- function(){
  results <- sample_generator()
  return(rnorm(1, mean = results, sd=1))
}

single_random_var <- normal_sample_gen()
```

Q10

```
df <- read_csv("https://jlucasmckay.bmi.emory.edu/global/bmi510/simpson_data.csv")
```

```
## Rows: 2498 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (3): Age, Exercise, Cholesterol
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
library(ggplot2)
ggplot(df, aes(x = Exercise, y = Cholesterol, color = factor(Age))) +
  geom_point(alpha = 0.5, size = 0.8) +
  geom_smooth(method = 'lm', show.legend = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```