```
# Function for toss coin
library(ggplot2)
coinTossOutcomes <- function(N, p) {
  # Define the sample space and probabilities
  sample_space <- c("H", "T")
  probabilities <- c(p, 1 - p)

  # Generate all possible outcomes for N coin tosses
  all_outcomes <- expand.grid(replicate(N, sample_space, simplify = FALSE))

  # Initialize a vector to store the probabilities of each outcome
  outcome_probabilities <- numeric(nrow(all_outcomes))

  # Calculate the probability of each outcome
  for (i in 1:nrow(all_outcomes)) {
    outcome <- as.character(unlist(all_outcomes[i, ]))
    prob <- prod(ifelse(outcome == "H", probabilities[1], probabilities[2]))
    outcome_probabilities[i] <- prob
  }

  # Add the probabilities to the data frame
  all_outcomes$Probability <- outcome_probabilities

  # Combine the toss outcomes into a single string for easier viewing
  all_outcomes$X <- do.call(paste0, all_outcomes[, 1:N])

  # Create the final table
  final_table <- all_outcomes[, c("X", "Probability")]

  # Normalize table
  return(final_table)

}

# 1. N = 5
N <- 5
p <- 0.8
result <- coinTossOutcomes(N, p)
x_axis = seq(0,1,0.001) ;print(x_axis)
y = c()
for (x in x_axis) {
  events = sum(result$Probability > x) # >= (1-x)
  y_axis = log2(events)
```

```
  y = c(y,y_axis)

}
final_result = data.frame(y,x_axis)
final_result$normal = final_result$y/N
a = final_result$normal

# 2. N = 10
rm(final_table)
rm(result)
N <- 10
p <- 0.8
result <- coinTossOutcomes(N, p)
x_axis = seq(0,1,0.001) ;print(x_axis)
y = c()
for (x in x_axis) {
  events = sum(result$Probability > x) # >= (1-x)
  y_axis = log2(events)
  y = c(y,y_axis)

}
final_result = data.frame(y,x_axis)
final_result$normal = final_result$y/N
b = final_result$normal



# 3. N = 15
rm(final_table)
rm(result)
N <- 15
p <- 0.8
result <- coinTossOutcomes(N, p)
x_axis = seq(0,1,0.001) ;print(x_axis)
y = c()
for (x in x_axis) {
  events = sum(result$Probability > x) # >= (1-x)
  y_axis = log2(events)
  y = c(y,y_axis)

}
final_result = data.frame(y,x_axis)
final_result$normal = final_result$y/N
c = final_result$normal
```

```
# 4. N = 20
rm(final_table)
rm(result)
N <- 20
p <- 0.8
result <- coinTossOutcomes(N, p)
x_axis = seq(0,1,0.001) ;print(x_axis)
y = c()
for (x in x_axis) {
  events = sum(result$Probability > x) # >= (1-x)
  y_axis = log2(events)
  y = c(y,y_axis)

}
final_result = data.frame(y,x_axis)
final_result$normal = final_result$y/N
d = final_result$normal




# 3. Combine all
df <- data.frame(x_axis = final_result$x_axis, "5" = a, "10" = b, "15" = c, "20" = d)

# Reshape the data to long format
df_long <- tidyr::gather(df, variable, value, -x_axis)

# Generate the plot
ggplot(df_long, aes(x = x_axis, y = value, color = variable)) +
  geom_line() +
  labs(title = "Q3.e) Essential Bits given delta",
       x = "delta",
       y = "Normalised Essential Bit Content") +
  scale_color_discrete(name = "N")
```