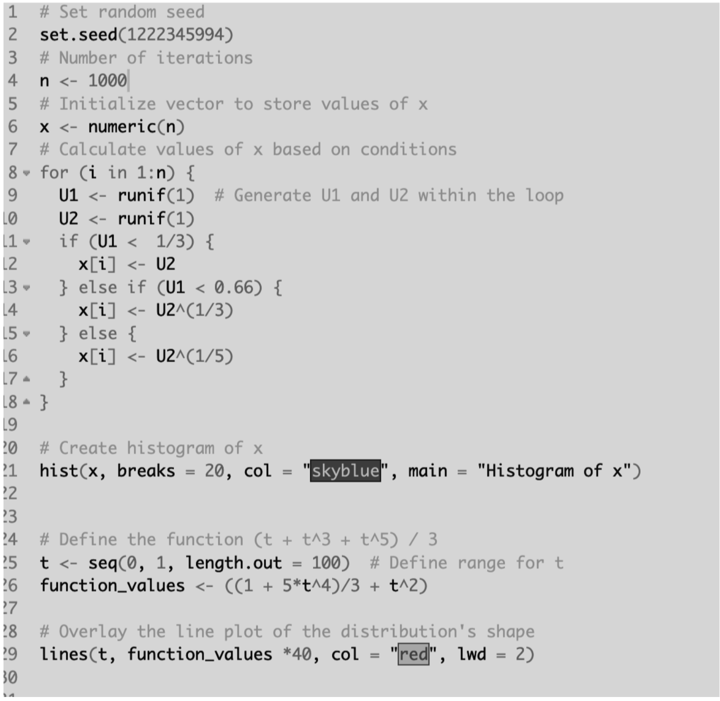
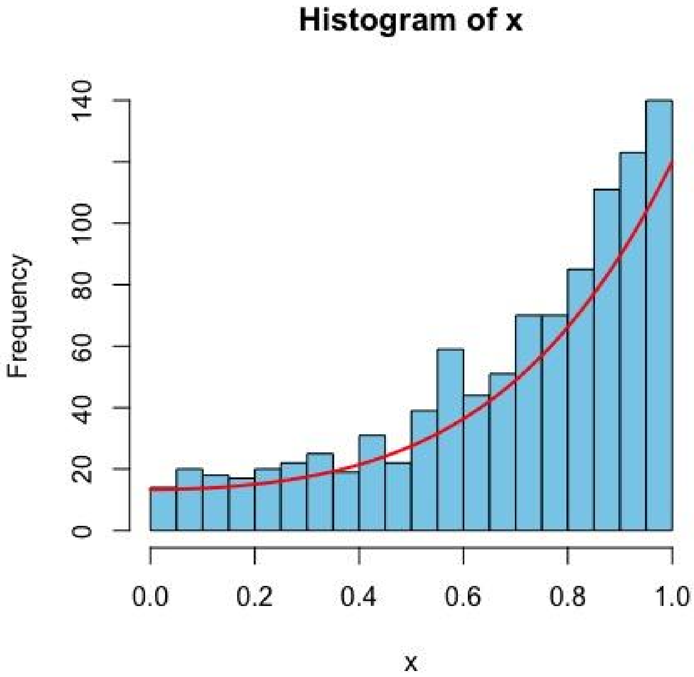
Emiliana Geronimo
Math 666: Simulations of Stochastic Systems
Homework 3

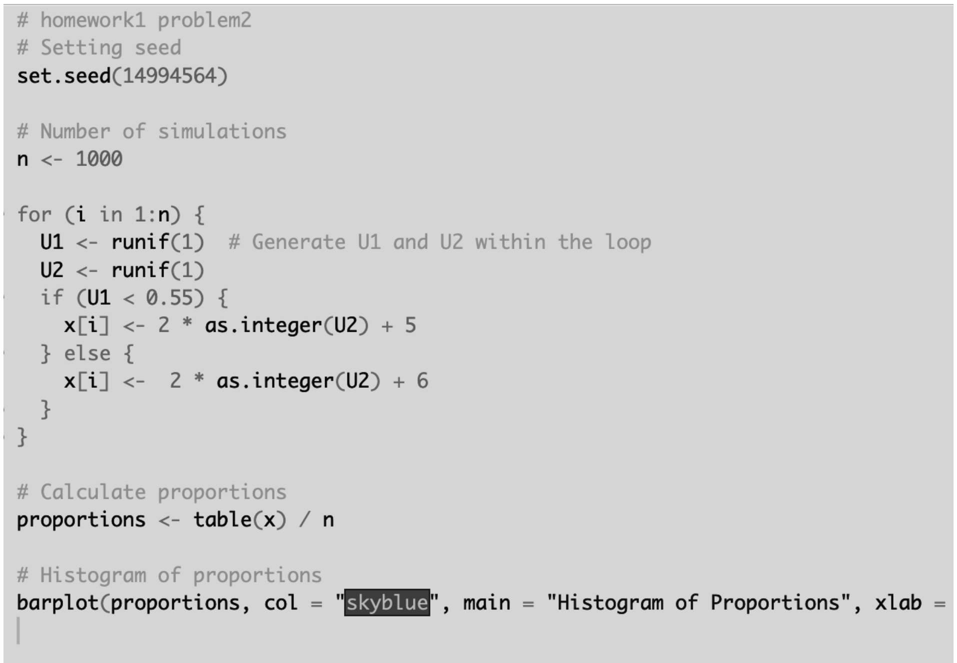
1.



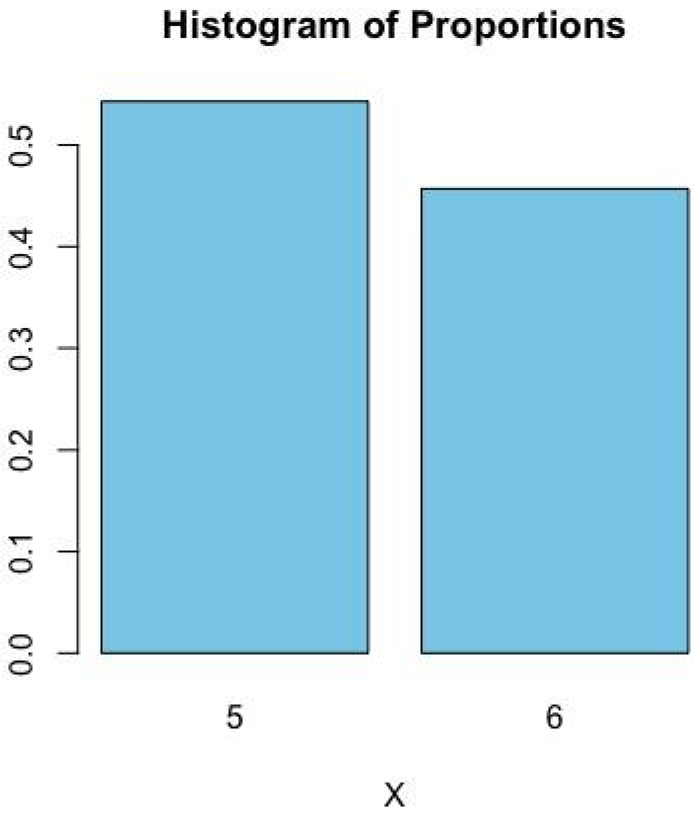
```r
# Set random seed
set.seed(1222345994)
# Number of iterations
n <- 1000
# Initialize vector to store values of x
x <- numeric(n)
# Calculate values of x based on conditions
for (i in 1:n) {
  U1 <- runif(1)  # Generate U1 and U2 within the loop
  U2 <- runif(1)
  if (U1 <  1/3) {
    x[i] <- U2
  } else if (U1 < 0.66) {
    x[i] <- U2^(1/3)
  } else {
    x[i] <- U2^(1/5)
  }
}

# Create histogram of x
hist(x, breaks = 20, col = "skyblue", main = "Histogram of x")


# Define the function (t + t^3 + t^5) / 3
t <- seq(0, 1, length.out = 100)  # Define range for t
function_values <- ((1 + 5*t^4)/3 + t^2)

# Overlay the line plot of the distribution's shape
lines(t, function_values *40, col = "red", lwd = 2)
```



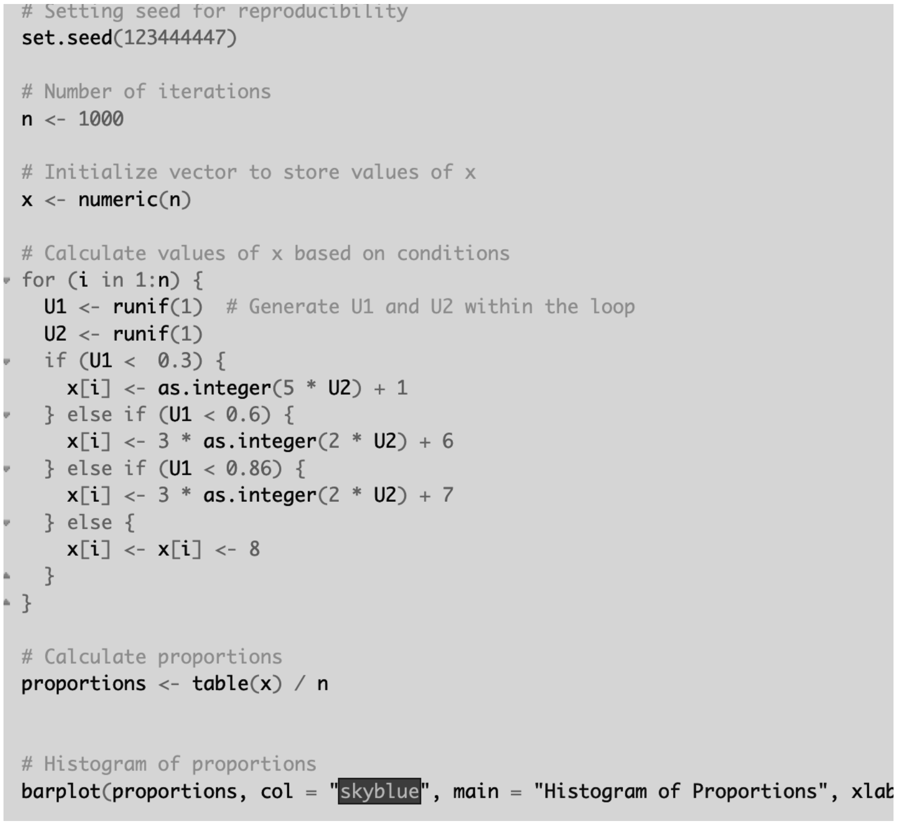
2.



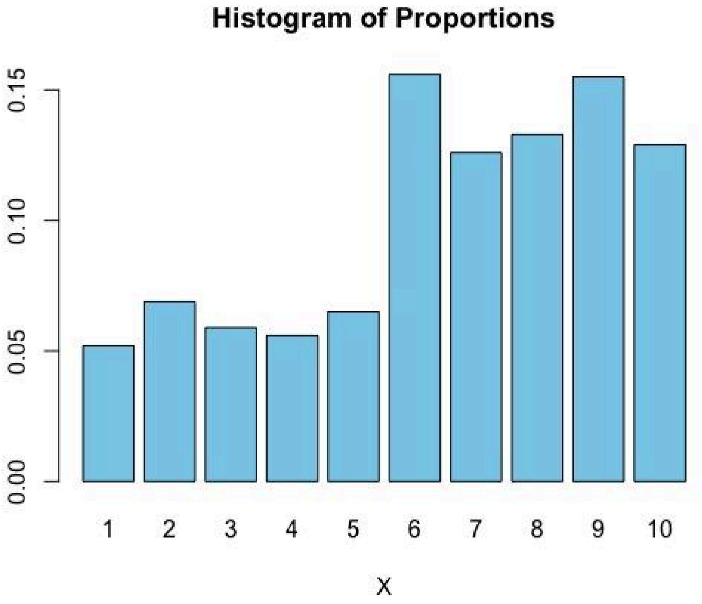
```r
# homework1 problem2
# Setting seed
set.seed(14994564)

# Number of simulations
n <- 1000

for (i in 1:n) {
  U1 <- runif(1)  # Generate U1 and U2 within the loop
  U2 <- runif(1)
  if (U1 < 0.55) {
    x[i] <- 2 * as.integer(U2) + 5
  } else {
    x[i] <-  2 * as.integer(U2) + 6
  }
}

# Calculate proportions
proportions <- table(x) / n

# Histogram of proportions
barplot(proportions, col = "skyblue", main = "Histogram of Proportions", xlab =
```



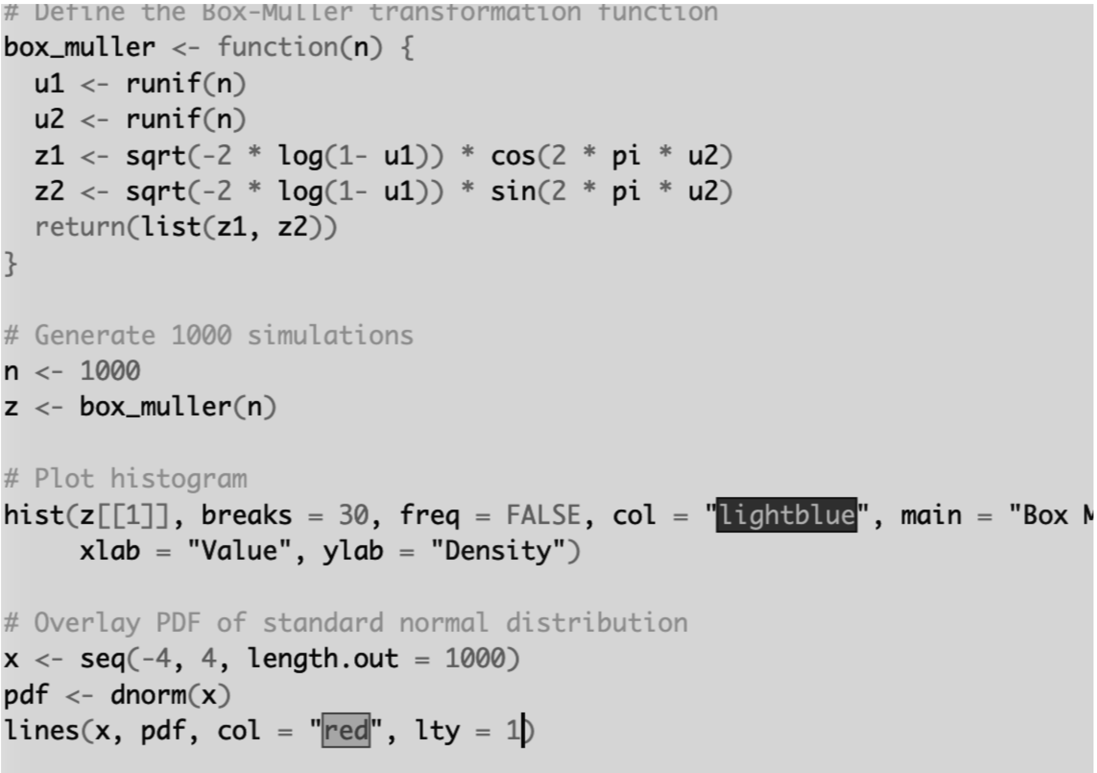
3.



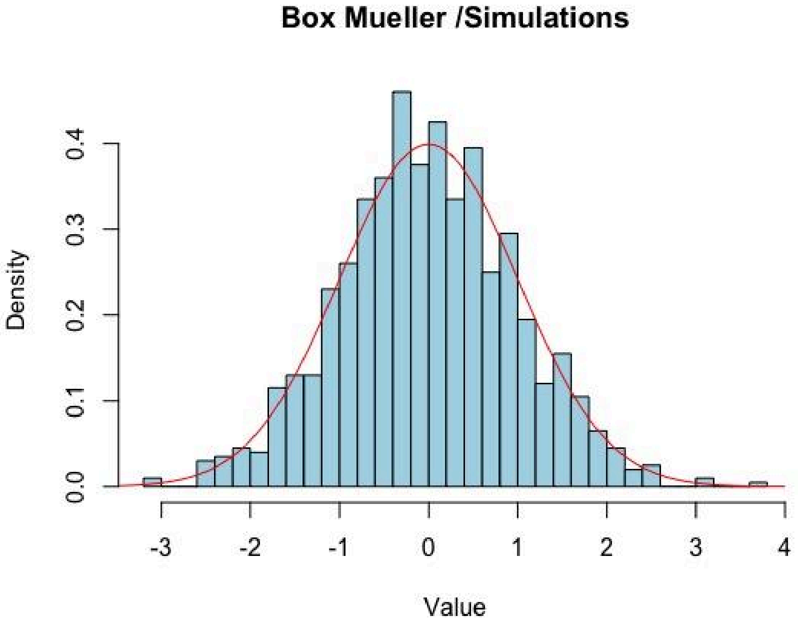
```
# Setting seed for reproducibility
set.seed(123444447)

# Number of iterations
n <- 1000

# Initialize vector to store values of x
x <- numeric(n)

# Calculate values of x based on conditions
for (i in 1:n) {
  U1 <- runif(1)  # Generate U1 and U2 within the loop
  U2 <- runif(1)
  if (U1 <  0.3) {
    x[i] <- as.integer(5 * U2) + 1
  } else if (U1 < 0.6) {
    x[i] <- 3 * as.integer(2 * U2) + 6
  } else if (U1 < 0.86) {
    x[i] <- 3 * as.integer(2 * U2) + 7
  } else {
    x[i] <- x[i] <- 8
  }
}

# Calculate proportions
proportions <- table(x) / n


# Histogram of proportions
barplot(proportions, col = "skyblue", main = "Histogram of Proportions", xlab
```



4.



```
# Define the Box-Muller transformation function
box_muller <- function(n) {
  u1 <- runif(n)
  u2 <- runif(n)
  z1 <- sqrt(-2 * log(1- u1)) * cos(2 * pi * u2)
  z2 <- sqrt(-2 * log(1- u1)) * sin(2 * pi * u2)
  return(list(z1, z2))
}

# Generate 1000 simulations
n <- 1000
z <- box_muller(n)

# Plot histogram
hist(z[[1]], breaks = 30, freq = FALSE, col = "lightblue", main = "Box M
     xlab = "Value", ylab = "Density")

# Overlay PDF of standard normal distribution
x <- seq(-4, 4, length.out = 1000)
pdf <- dnorm(x)
lines(x, pdf, col = "red", lty = 1)
```


Box Mueller /Simulations
Density
Value