Lab 1

Mervan Palmér (merpa433)

2023-09-13

Exempeluppgift 1

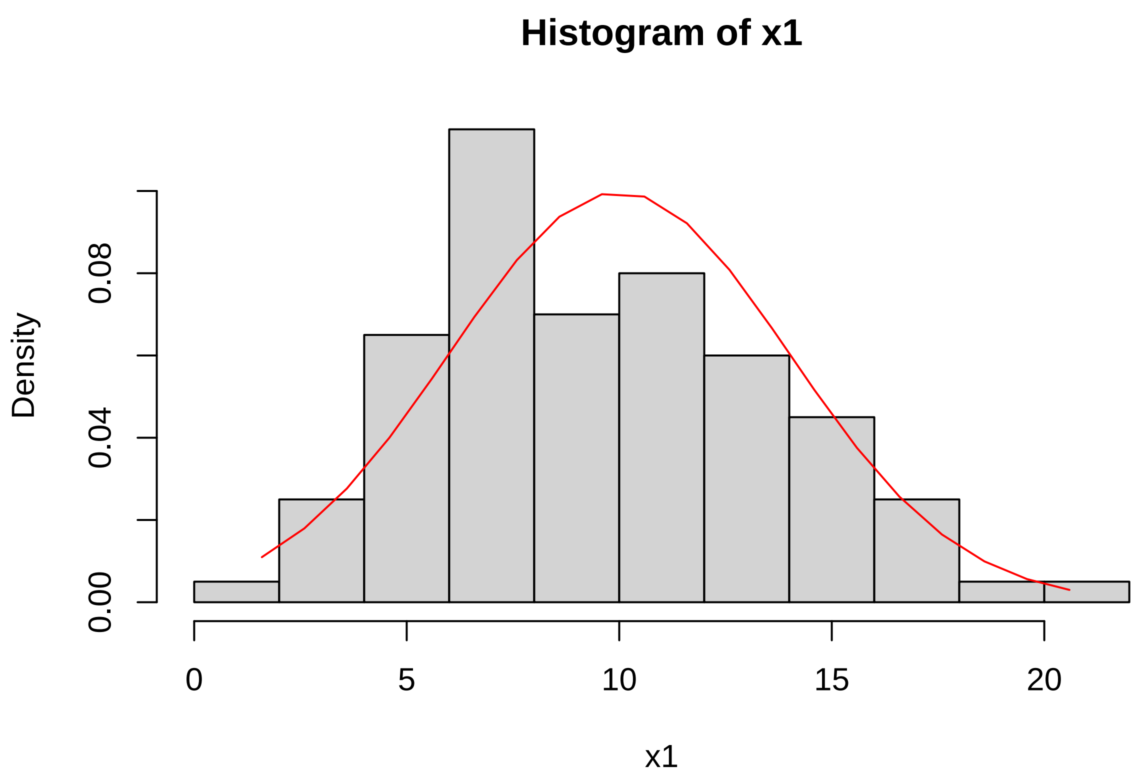
a) Simulera tal från normal- och exponentialfördelningen

Nedan simuleras normalfördelningen med olika antalet dragningar, samt ett histogram av dragningarna från normal-fördelningen.

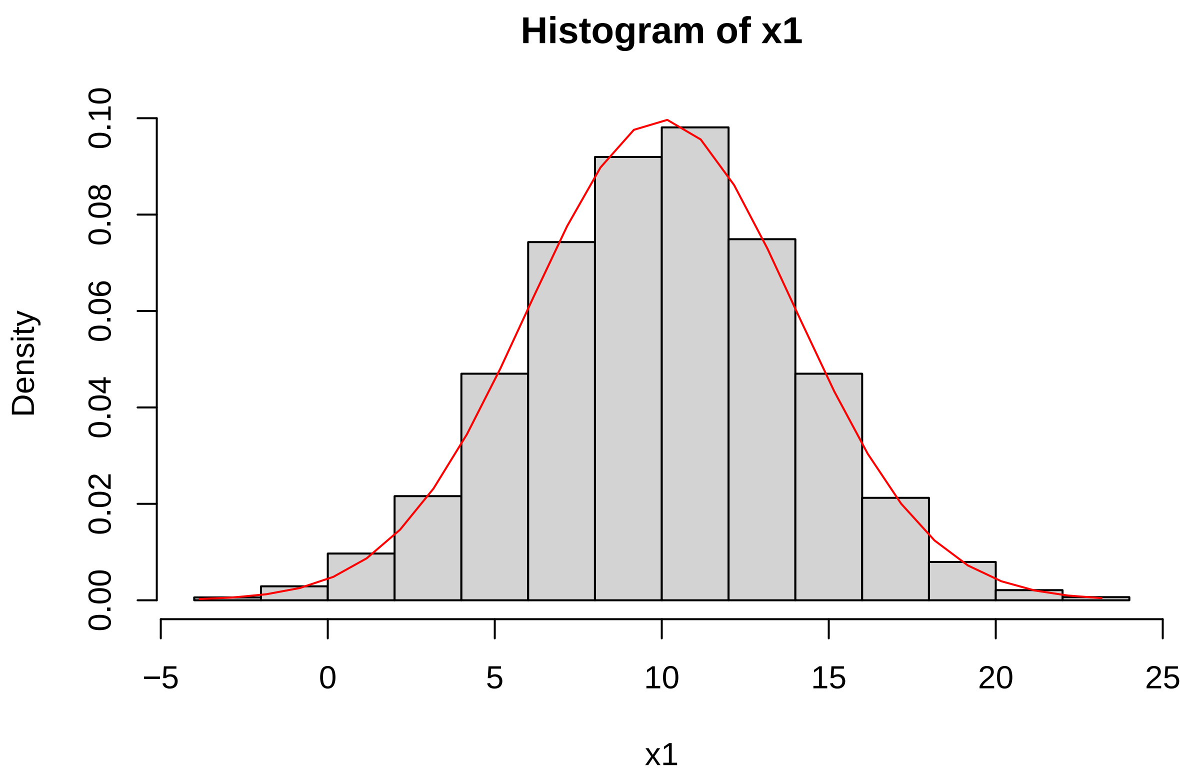
3.1.1

1)100 dragningar

```
x1 <- rnorm(100, mean = 10, sd = 4)
hist(x1, probability = TRUE)
xfit <- seq(min(x1), max(x1), 1)
yfit <- dnorm(xfit, mean = 10, sd = 4)
lines(xfit, yfit, col="red")
```



10000 dragningar

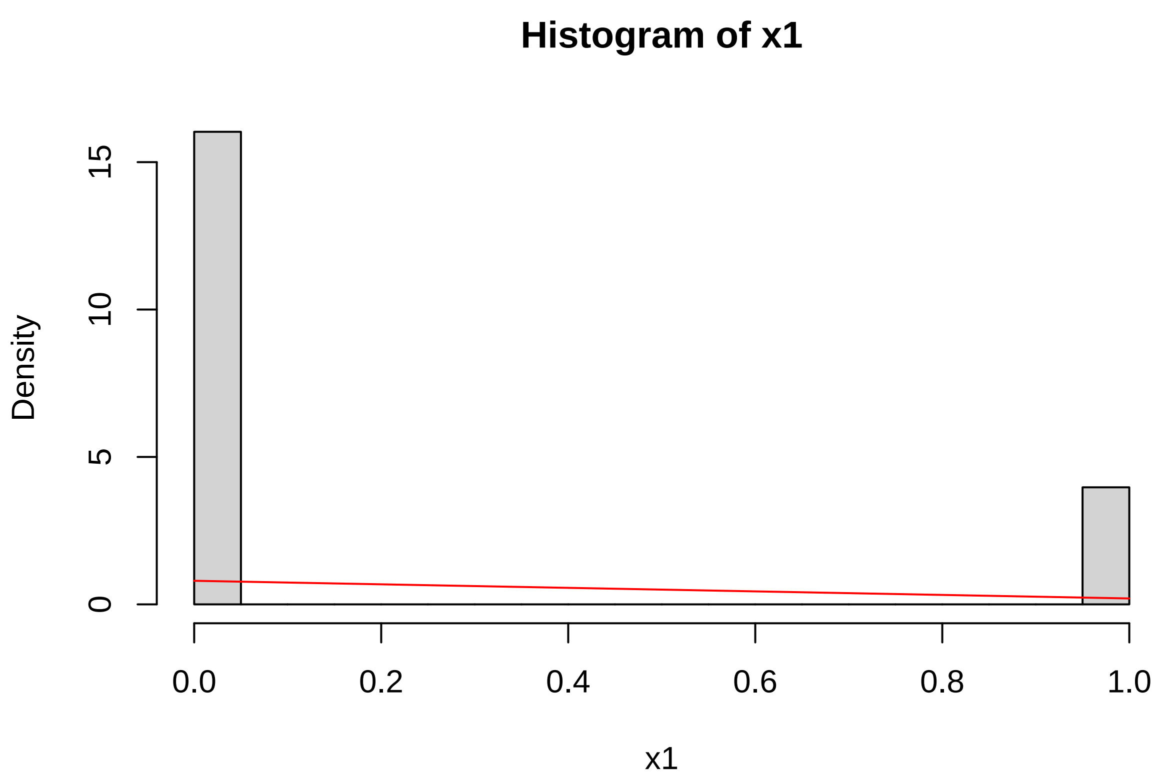


2)

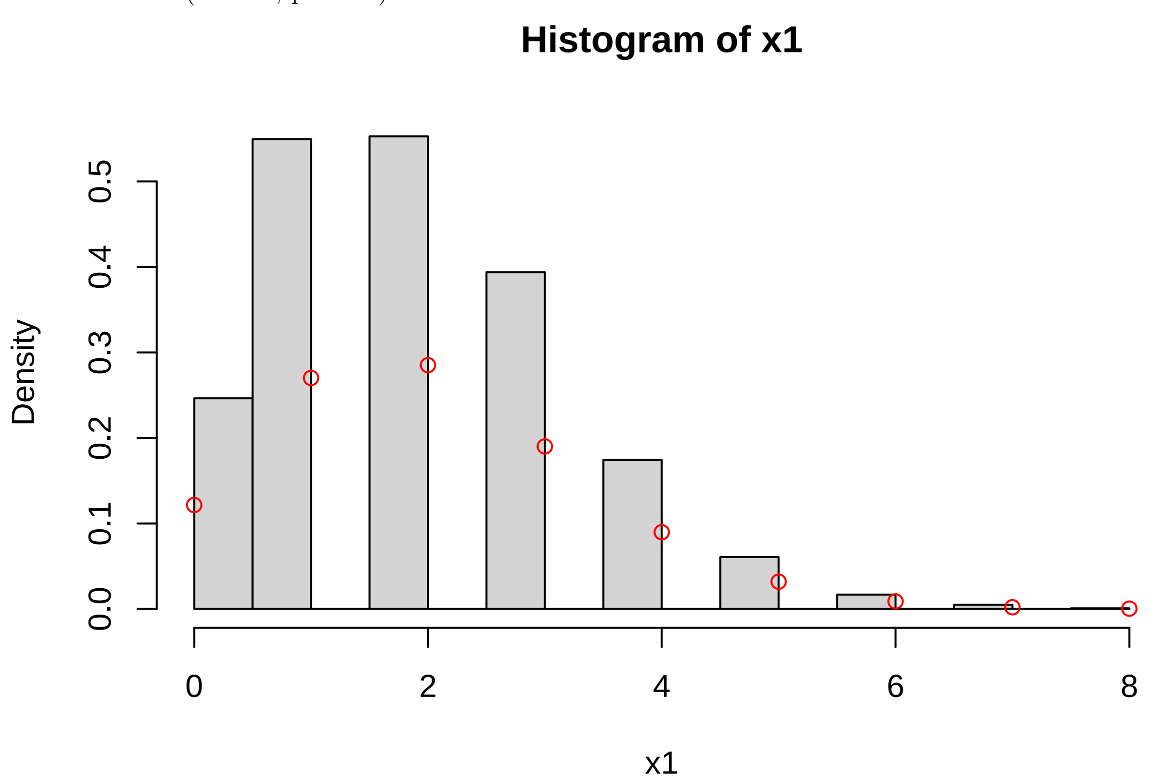
Den med 10000 dragningar är mer lik sin täthetsfunktion. Den varierar dessutom mindre pågrund av att antalet dragningar är större och följer en normalfördelning mycket mer. 100 dragningar är mer sannolik att se annorlunda ut då färre dragningar genomförs

3.1.2

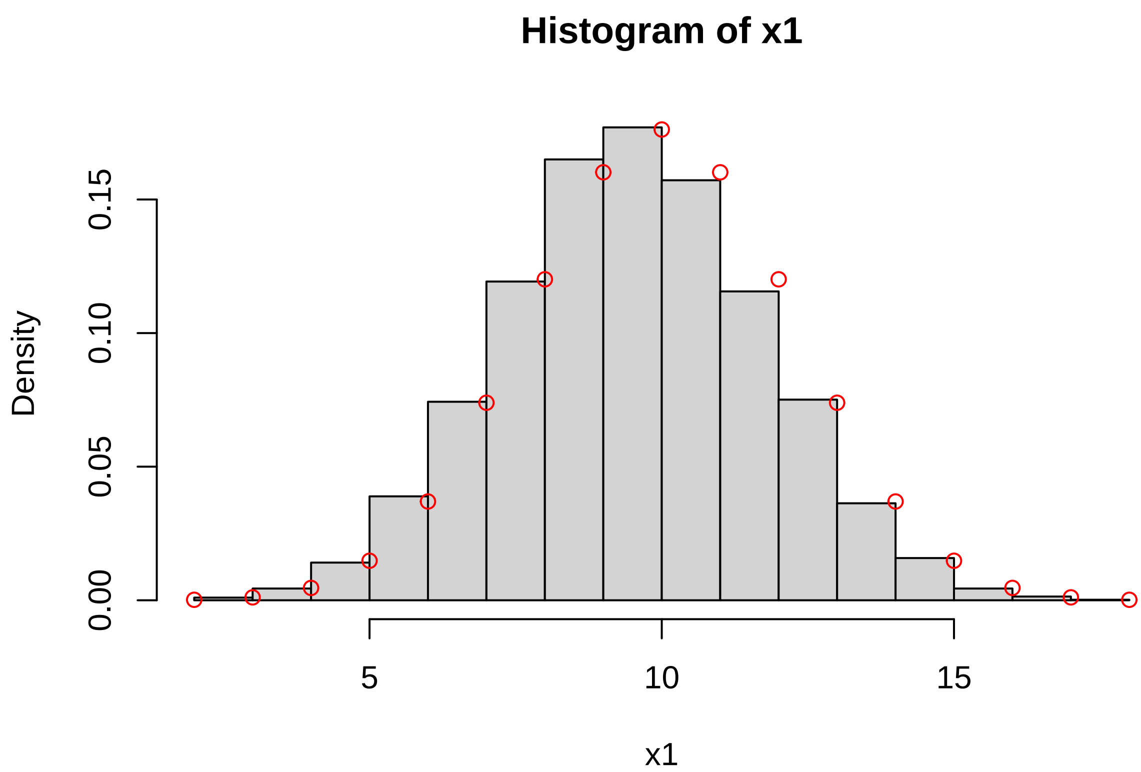
1) x1 Bernoulli(p = 0.2)



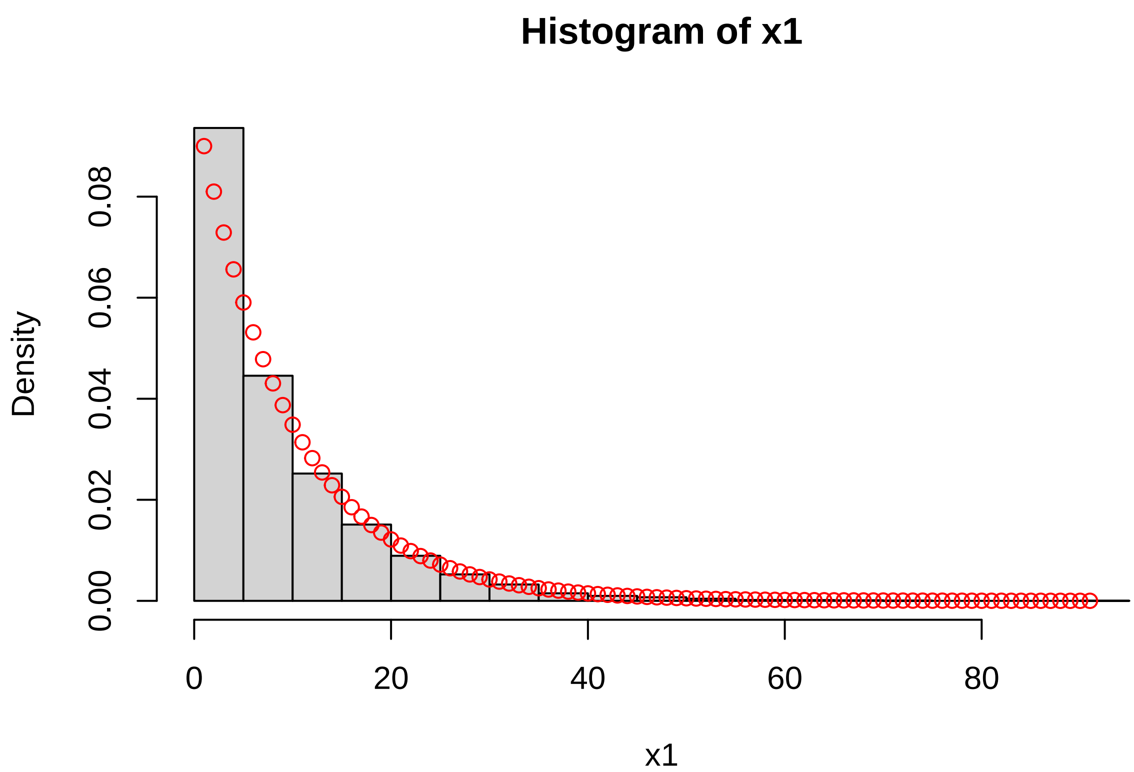
X2 = Binomial(n = 20, p = 0.1)



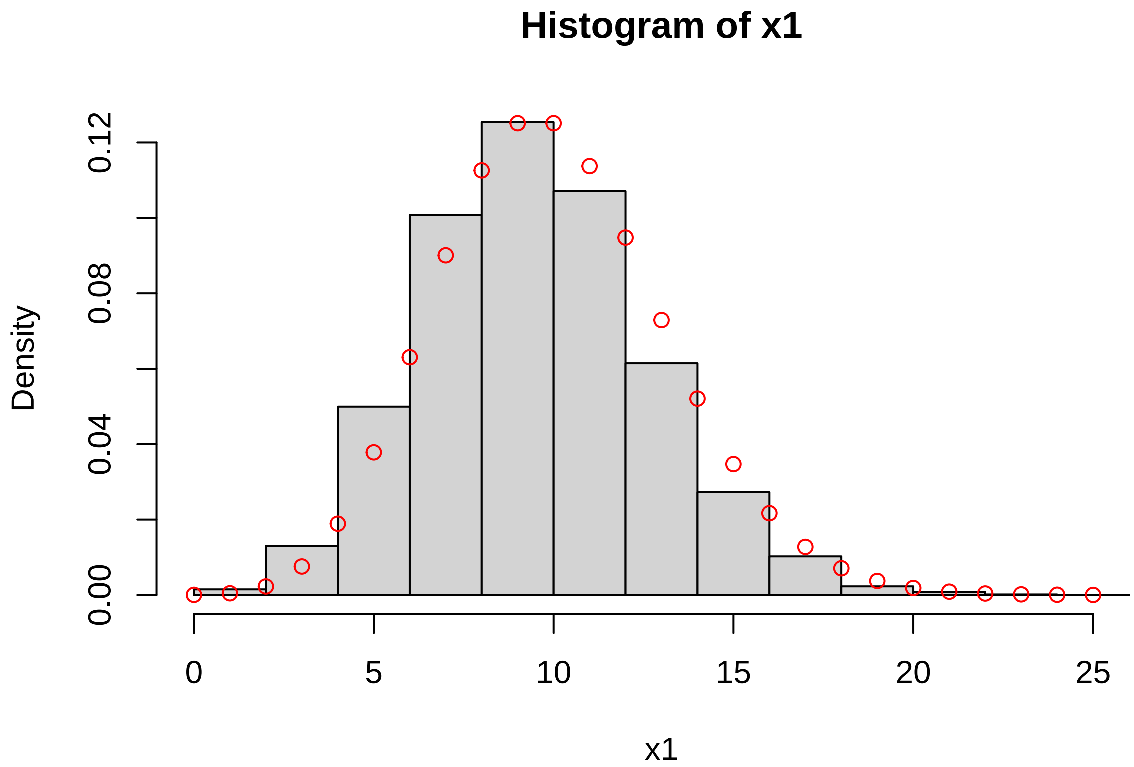
X3 = Binomial(n = 20, p = 0.5)



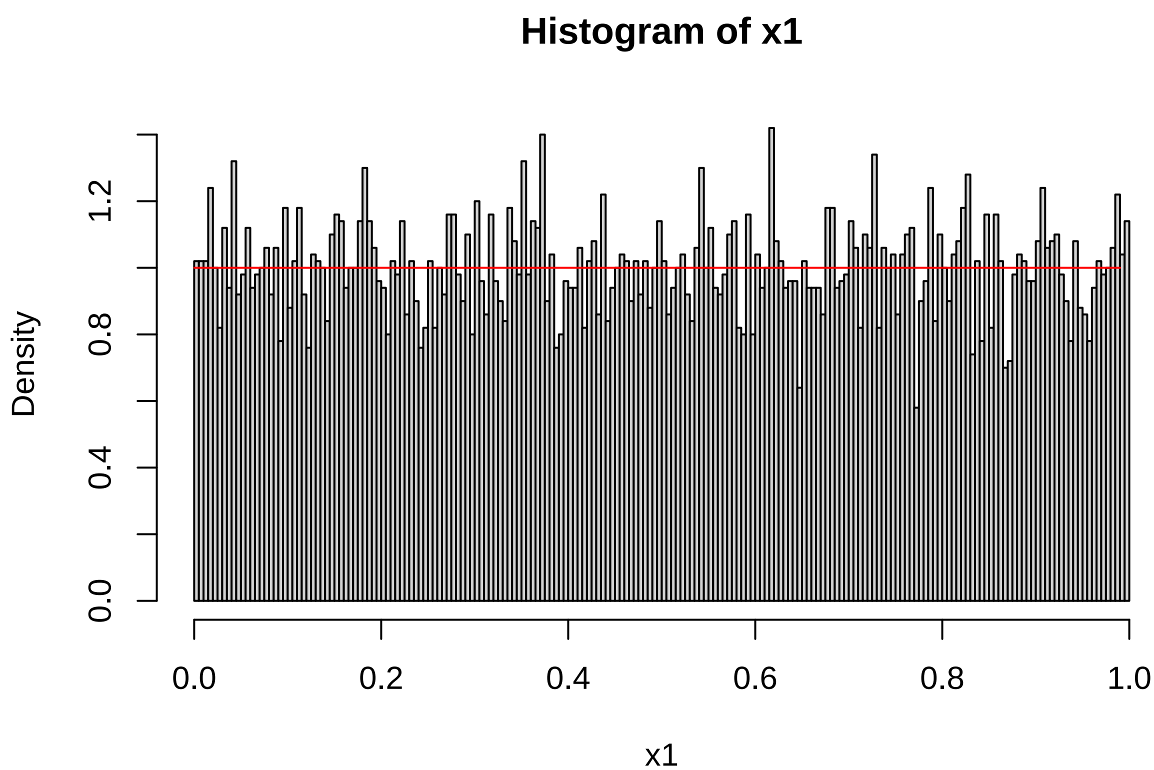
X4 = Geometrisk(p = 0.1)



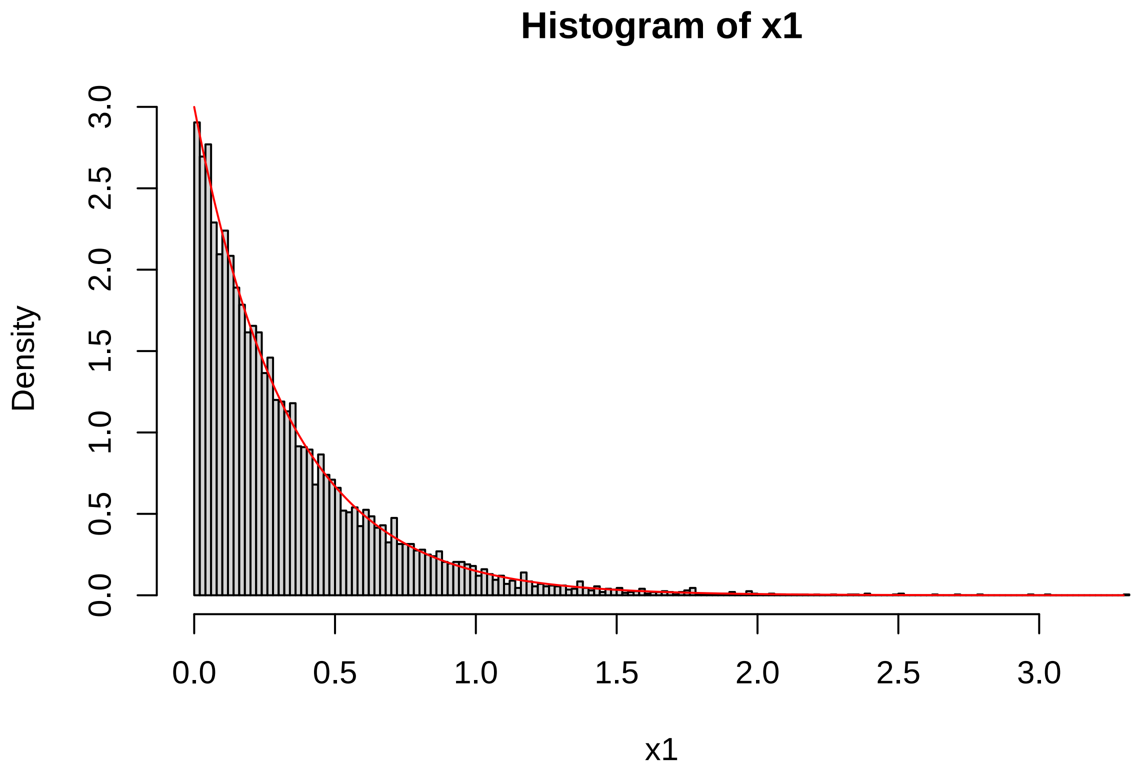
X5 = Poisson(lambda = 10)



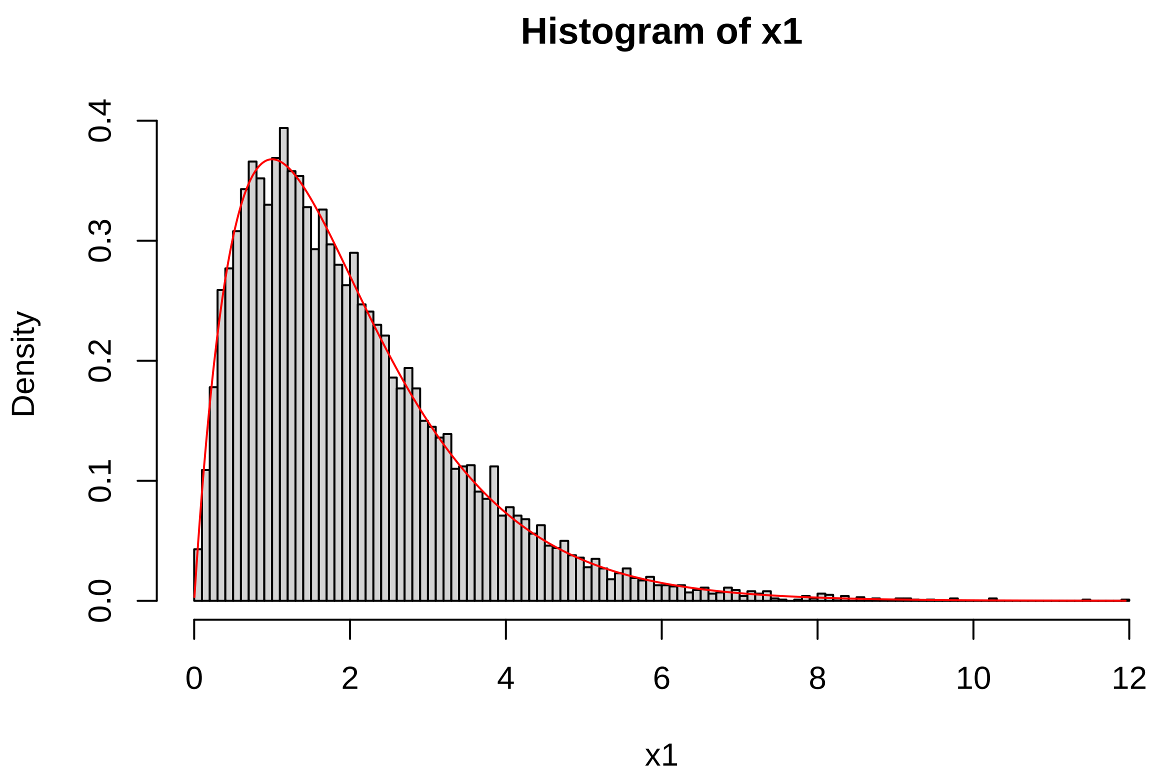
Y1 = Likformig(min = 0, max = 1)



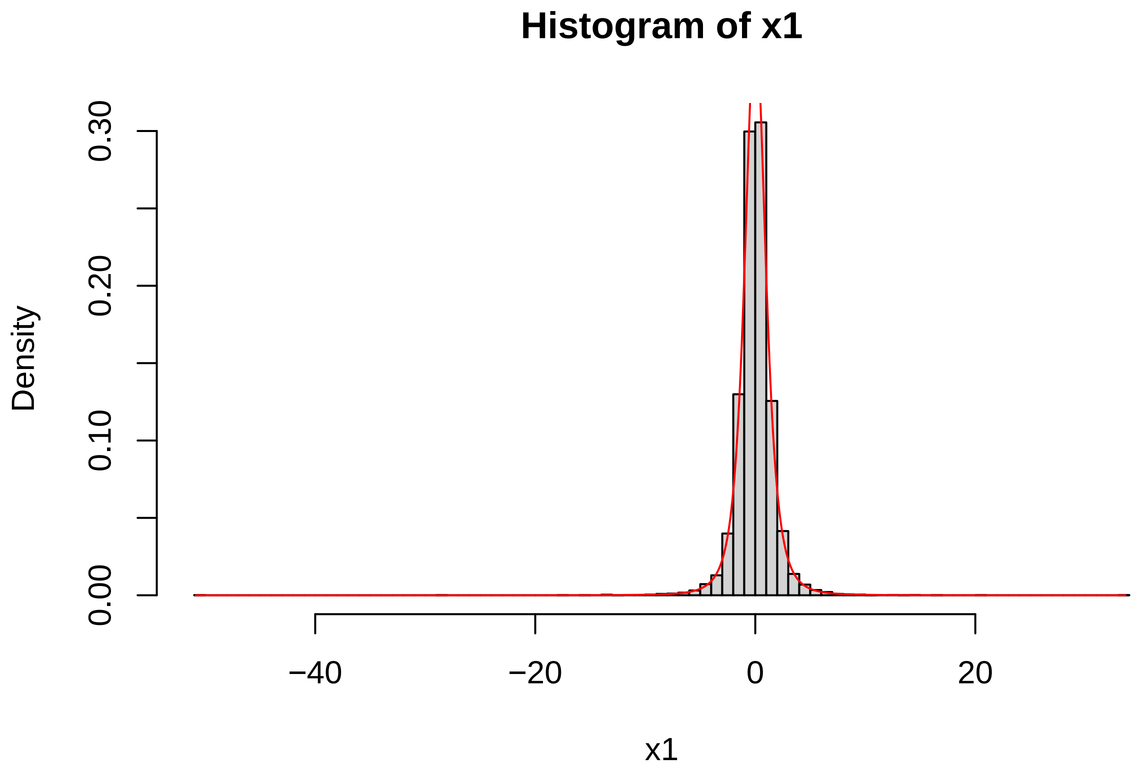
Y2 = Exp(theta = 3)



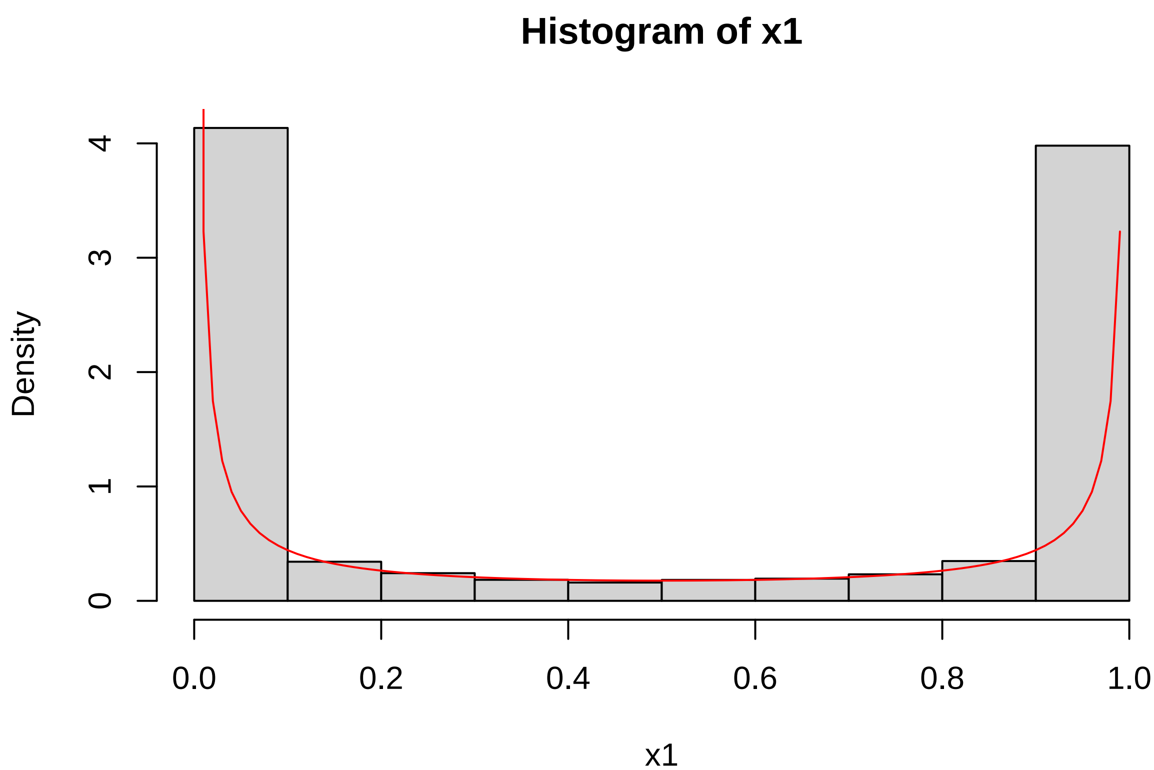
Y3 = Gamma(alfa = 2, beta = 1)



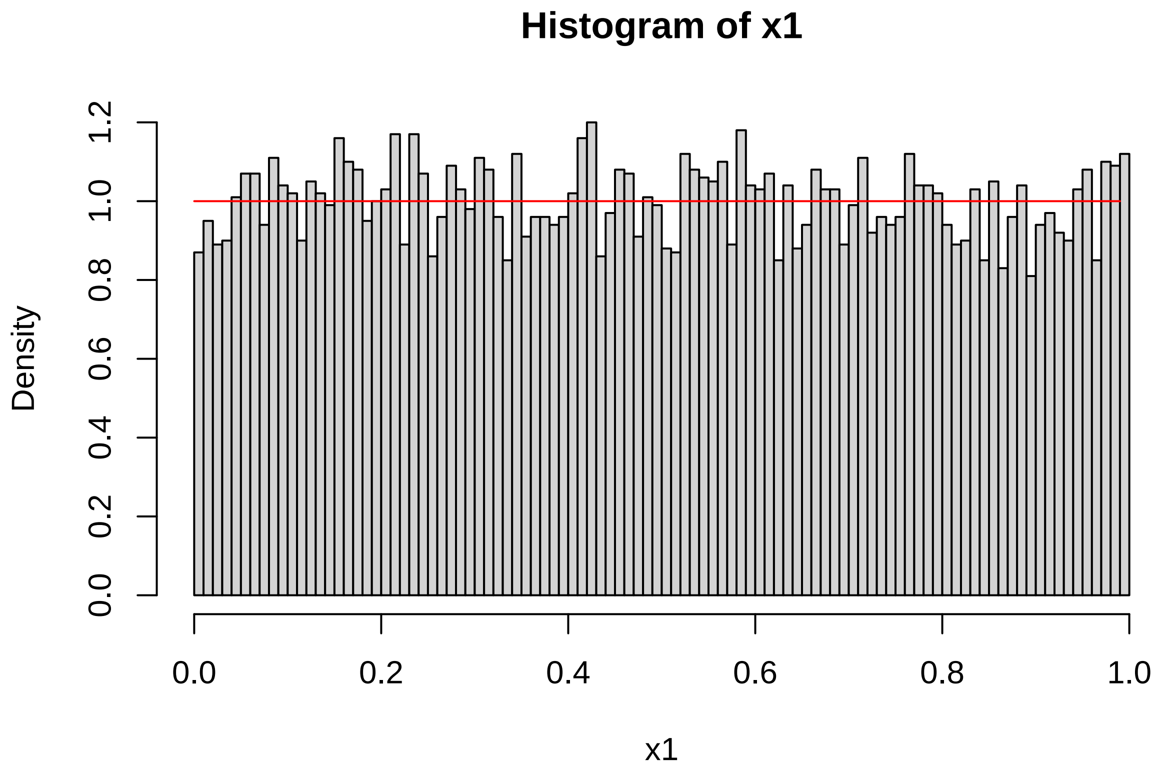
Y4 = Student-t(v = 3)



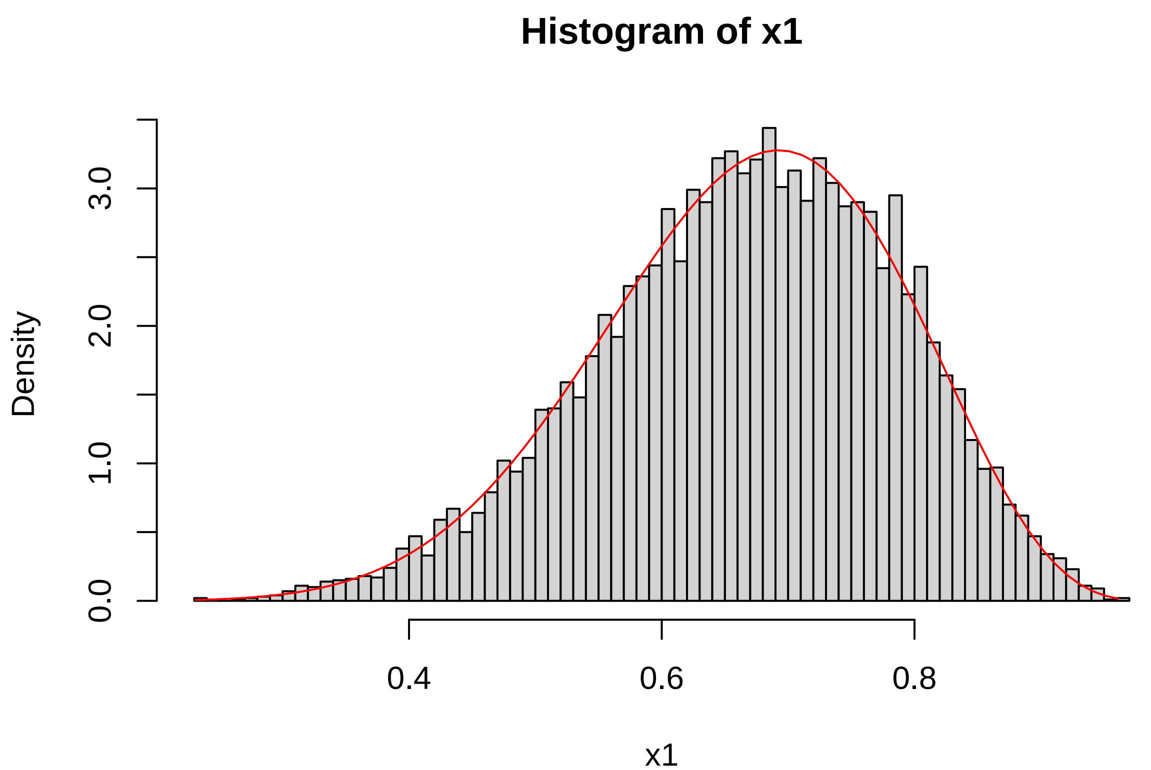
Y5 = Beta(alfa = 0.1, beta = 0.1)



Y6 = Beta(alfa = 1, beta = 1)



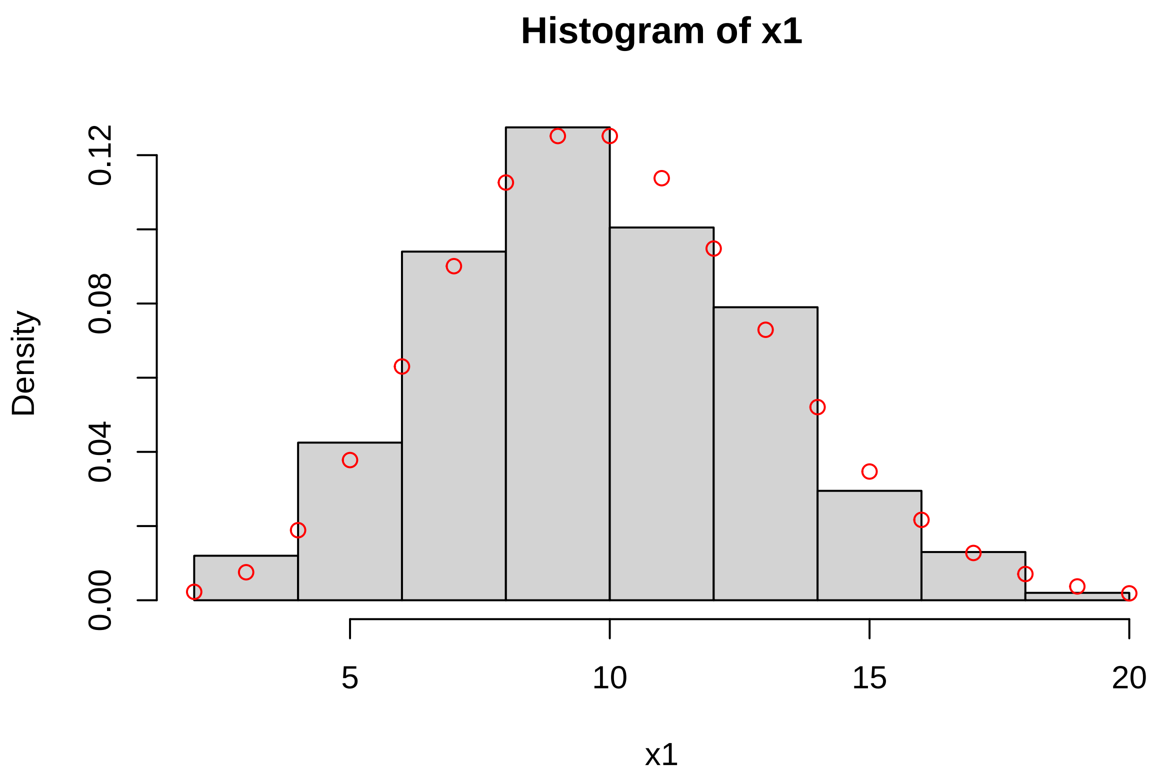
Y7 = Beta(alfa = 10, beta = 5)



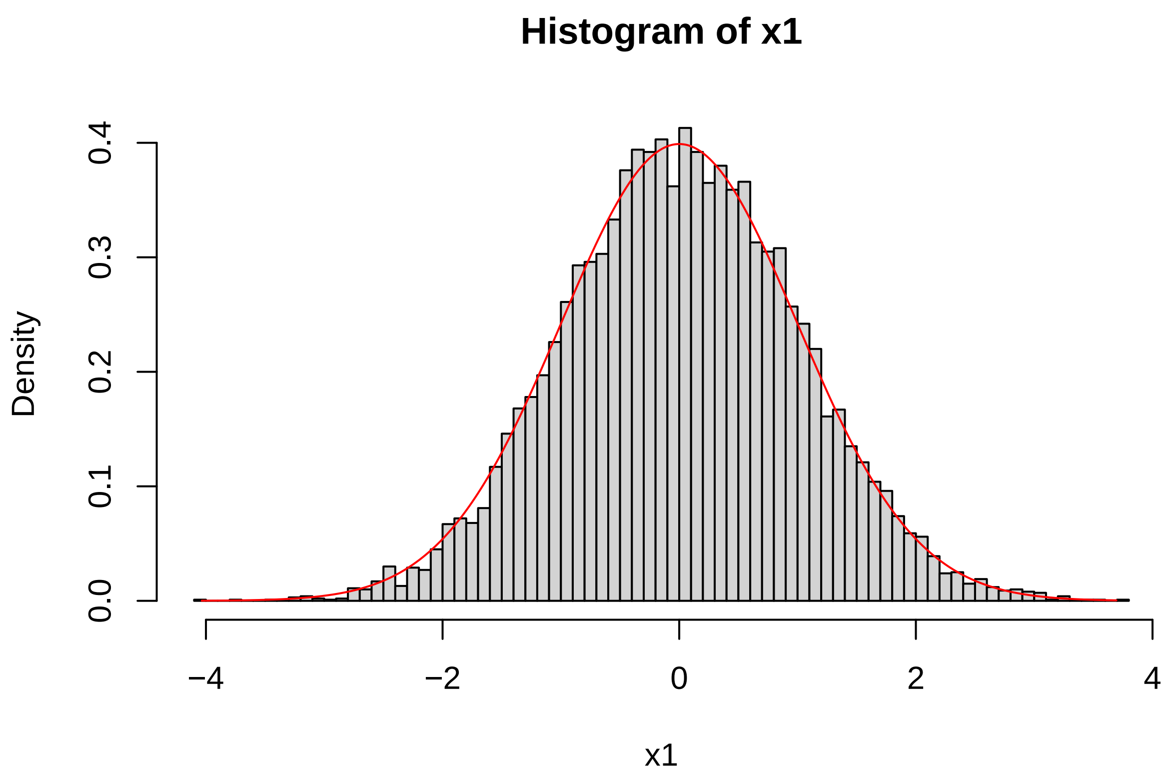
3.1.3

1)

X = Binomial(n = 10000, p = 0.001)



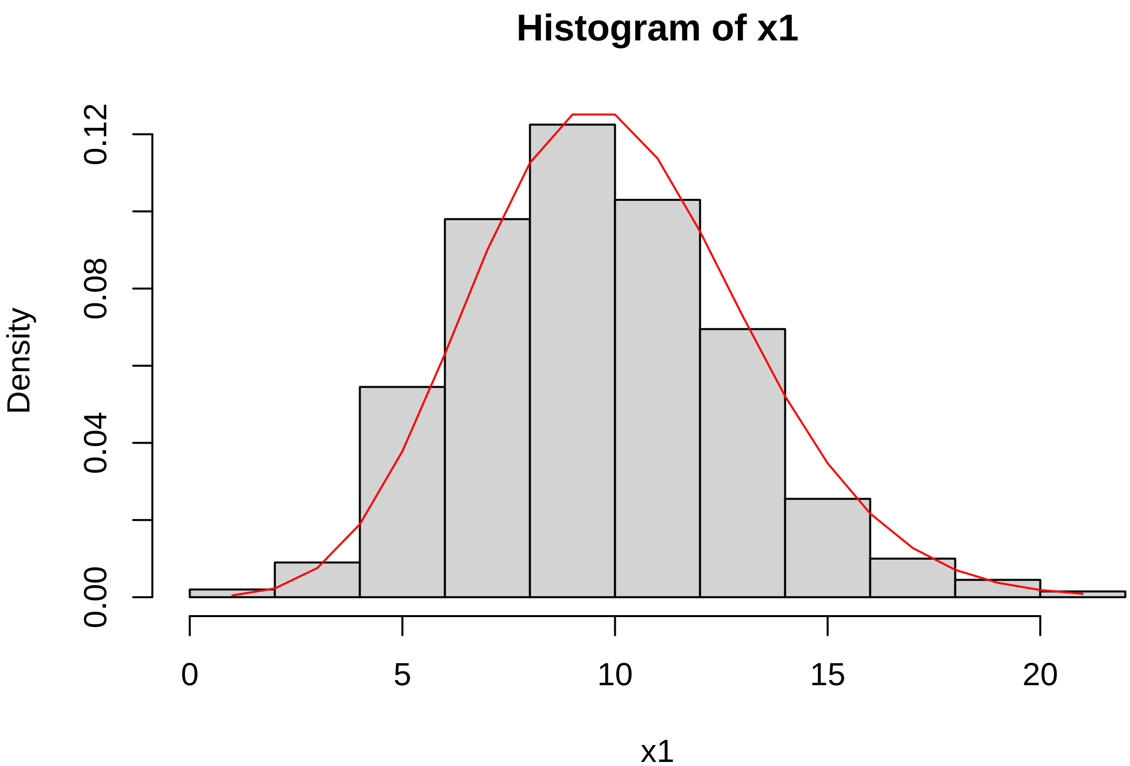
Y = Student-t(v = 10000)



2) Binomial fördelningar konvergar mot Poisson fördelningar. Student-t konvergerar till normal fördelning.

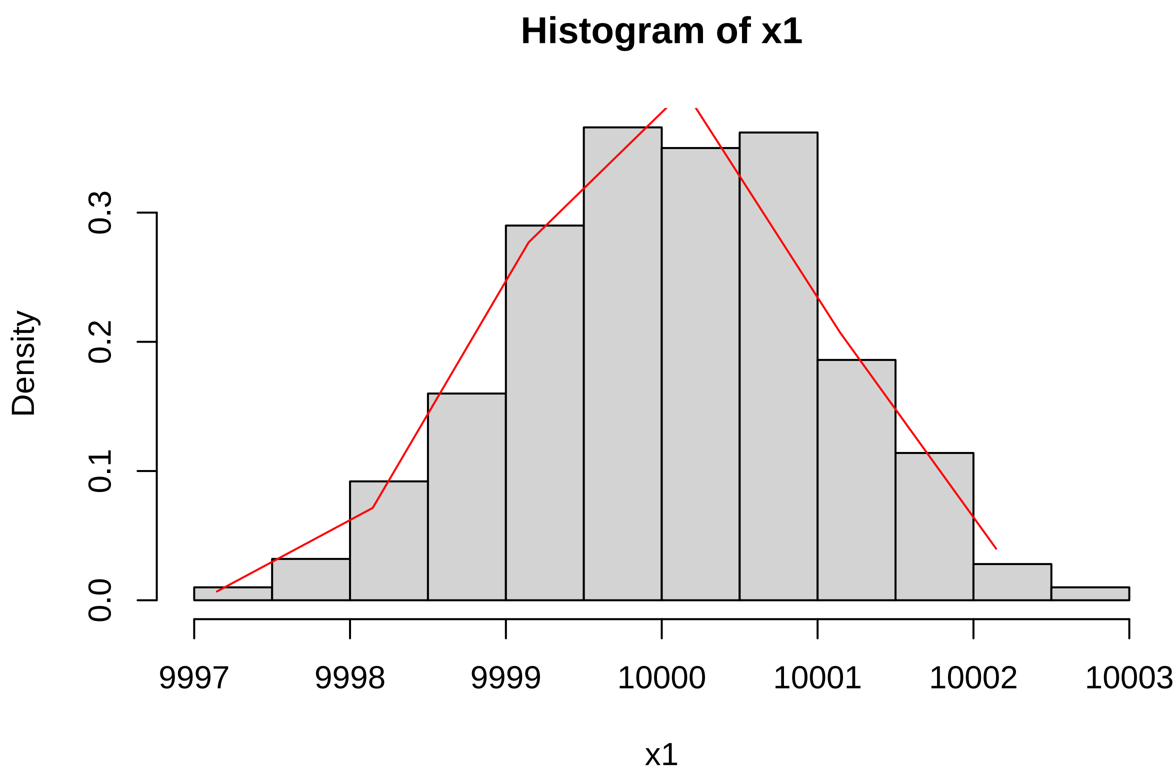
3)

De är liknande om man tar liknande parameterar och Här är motsvarande för binomialfördelning fast med poissonfördelning:



Även denna är lik sin motsvarighet. Här är motsvarande för Student-t fördelning fast med normalfördelning istället:

```
x1 <- rnorm(1000, mean = 10000, sd = 1)
hist(x1, probability = TRUE)
xfit <- seq(min(x1), max(x1), 1)
yfit <- dnorm(xfit, mean = 10000, sd = 1)
lines(xfit, yfit, col="red")
```



3.1.4

1)

```
y <- rbinom(10000, 10, 0.1)
count_where_y_is_0 <- sum(y == 0)
share_y_0 <- count_where_y_is_0 / 10000
print(share_y_0)
```

```
## [1] 0.3466
```

2.

```
p1 <- pnorm(0, 0, 1)
print(p1)
```

```
## [1] 0.5
```

```
p2 <- pnorm(1, 0, 1) - pnorm(-1, 0, 1)
print(p2)
```

```
## [1] 0.6826895
```

```
p3 <- 1 - pnorm(1.96, 0, 1)
print(p3)
```

```
## [1] 0.0249979
```

```
p4 <- pbinom(10, 10, 0.1) - pbinom(0, 10, 0.1)
print(p4)
```

```
## [1] 0.6513216
```

```
p5 <- pbinom(0 + 0.0001, 10, 0.1) - pbinom(0 - 0.0001, 10, 0.1)
print(p5)
```

```
## [1] 0.3486784
```

```r
p6 <- p4 + p5
print(p6)
```

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

3.

```r
norm_func <- rnorm(10000, 0, 1)
binom_func <- rbinom(10000, 10, 0.1)

p1sum <- sum(norm_func < 0) / 10000
print(p1sum)
```

```
## [1] 0.5024
```

```r
p2sum <- (sum(norm_func < 1) - sum(norm_func <= -1)) / 10000
print(p2sum)
```

```
## [1] 0.6908
```

```r
p3sum <- sum(norm_func > 1.96) / 10000
print(p3sum)
```

```
## [1] 0.0247
```

```r
p4sum <- (sum(binom_func < 10) - sum(binom_func <= 0)) / 10000
print(p4sum)
```

```
## [1] 0.6523
```

```r
p5sum <- sum(binom_func == 0) / 10000
print(p5sum)
```

```
## [1] 0.3477
```

```r
p6sum <- (sum(binom_func <= 10) - sum(binom_func < 0)) / 10000
print(p6sum)
```

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

3.1.5

1.

```r
old <- rbinom(n = 10000, size = 337, p = 0.1)
print(sum(old)/10000)
```

```
## [1] 33.7128
```

```r
prob <- sum(runif(n = 10000, min = 0.02, max = 0.16))/10000
new <- rbinom(n = 10000, size = 337, p = prob)
print(sum(new)/10000)
```

```
## [1] 30.4349
```

2.

```r
print(sum(new < old)/10000)
```

```
## [1] 0.6411
```

3.

```r
sum(old > 50)/10000
```

```
## [1] 0.0017
```

```r
sum(new > 50)/10000
```

```
## [1] 1e-04
```

3.2.1

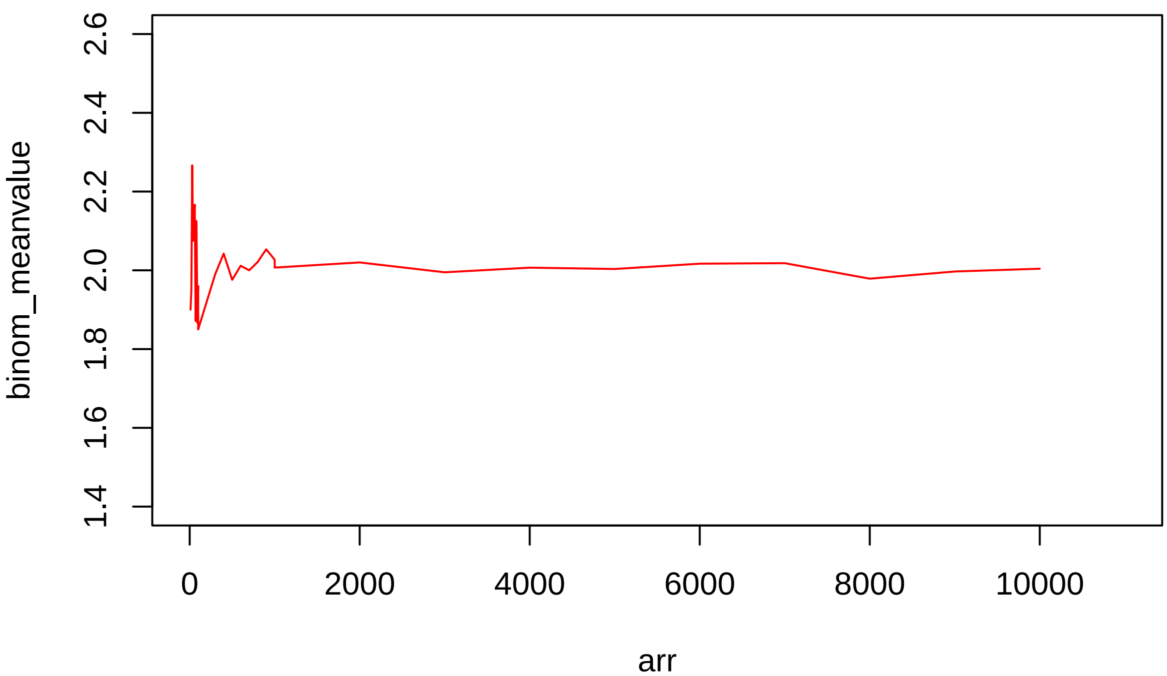
1.

E(Y) = 2 / 2 = 1

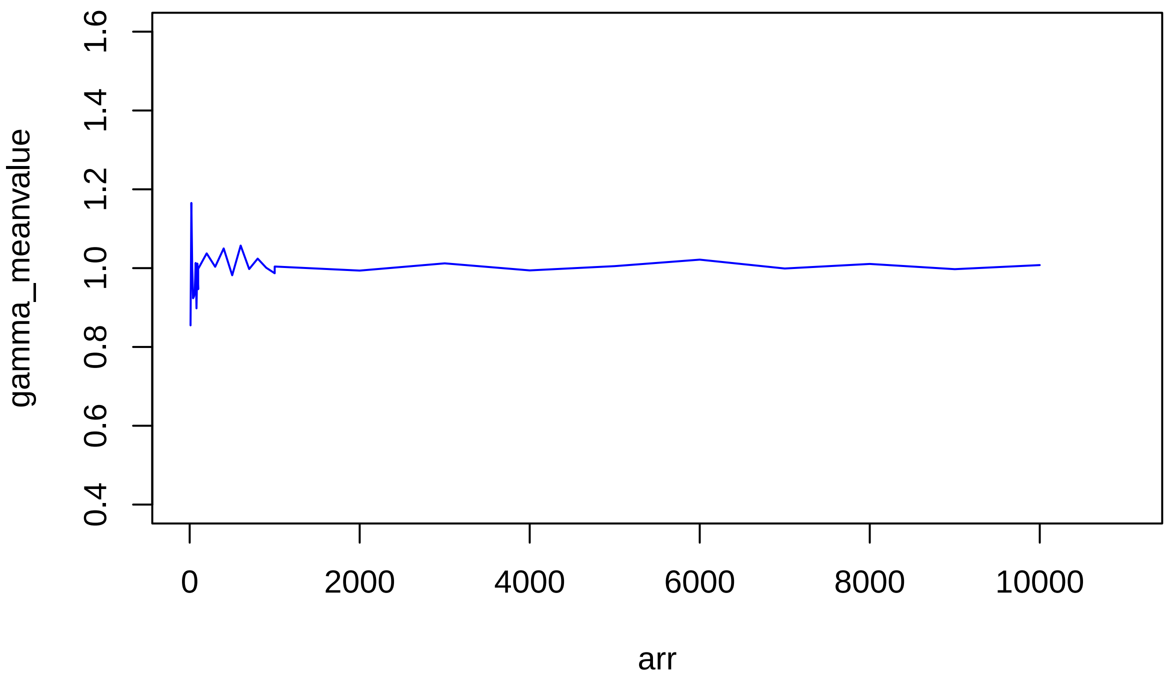
E(X) = 10 * 0.2 = 2

2.

```r
arr <- c(seq(10, 100, 10), seq(100, 1000, 100), seq(1000, 10000, 1000))
binom_meanvalue <- numeric(length(arr))
gamma_meanvalue <- numeric(length(arr))
for (i in 1:length(arr)) {
  n <- arr[i]
  binom_meanvalue[i] <- mean(rbinom(n, 10, 0.2))
  gamma_meanvalue[i] <- mean(rgamma(n, 2, 2))
}
plot(arr, binom_meanvalue, ylim=c(1.4,2.6), xlim=c(0, 11000), col="red", type="l")
```




```r
plot(arr, gamma_meanvalue, ylim=c(0.4,1.6), xlim=c(0, 11000), col="blue", type="l")
```



3.3.1

1.

E(X) = 1 / 10 = 0.1

E(Y) = 3

Var(X) = 1 / (10^2) = 0.01

Var(Y) = 3

2.

```
exp_value <- rexp(10000, 10)
print(mean(exp_value))
```

```
## [1] 0.1017785
```

```
print(var(exp_value))
```

```
## [1] 0.01041849
```

```
pois_value <- rpois(10000, 3)
print(mean(pois_value))
```

```
## [1] 2.9813
```

```
print(var(pois_value))
```

```
## [1] 2.951645
```

3.

E(3) = 3

E(3X + 2) = E(3X) + E(2) = 3 * E(X) + 2 = 0.3 + 2 = 2.3

E(X + Y) = E(X) + E(Y) = 0.1 + 3 = 3.1

E(X * Y) = E(X) * E(Y) = 0.1 * 3 = 0.3

E(3X + 2Y - 3) = 3E(X) + 2E(Y) - 3 = 0.3 + 6 - 3 = 3.3

Var(2X - 5) = 2^2 * Var(X) = 4 * 0.01 = 0.04

Var(X + Y) = Var(X) + Var(Y) = 0.01 + 3 = 3.01

4.

```
print(mean(3))
```

```
## [1] 3
```

```
print(mean(3*exp_value + 2))
```

```
## [1] 2.305336
```

```
print(mean(exp_value + pois_value))
```

```
## [1] 3.083079
```

```
print(mean(exp_value * pois_value))
```

```
## [1] 0.3033164
```

```
print(mean(3*exp_value + 2*pois_value - 3))
```

```
## [1] 3.267936
```

```
print(var(2*exp_value - 5))
```

```
## [1] 0.04167397
```

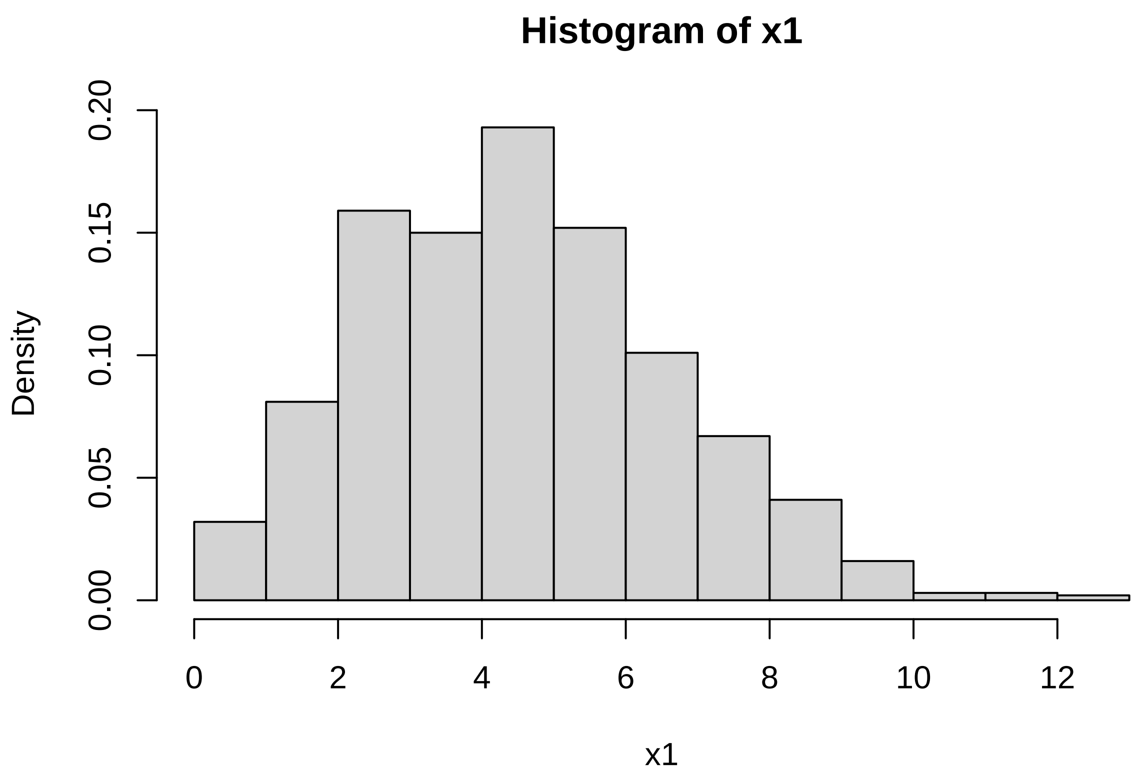
```
print(var(exp_value + pois_value))
```

```
## [1] 2.961832
```

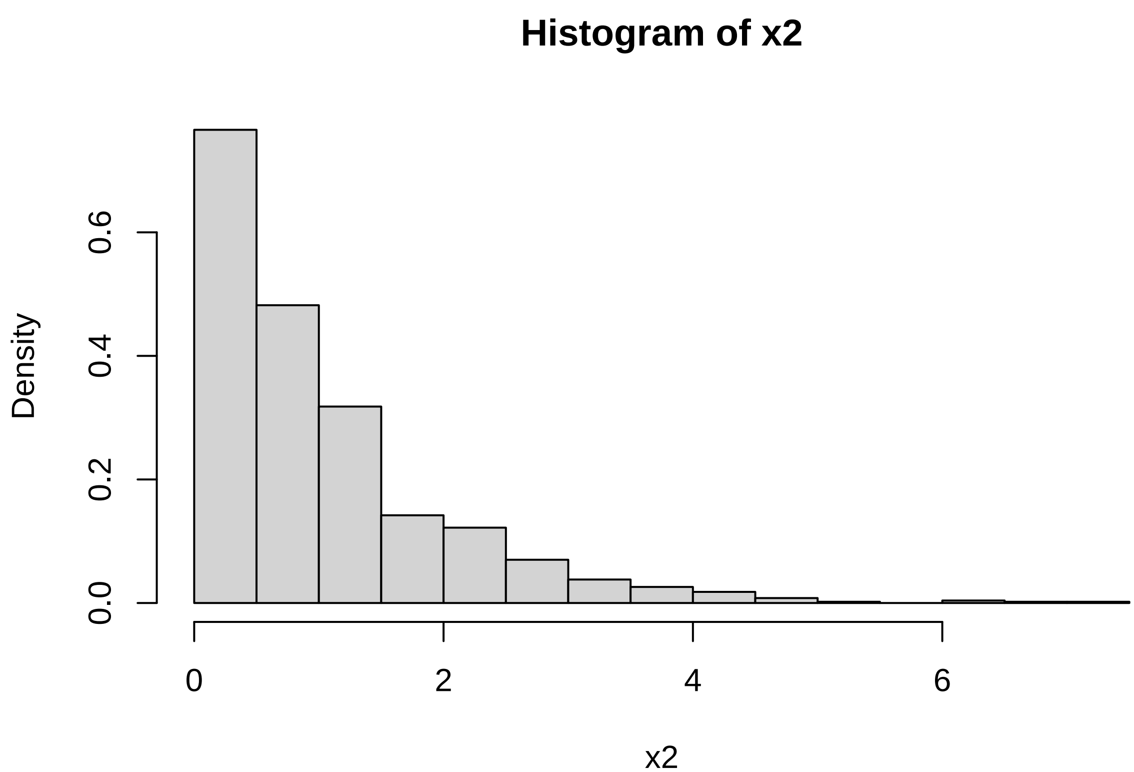
3.4.1

1.

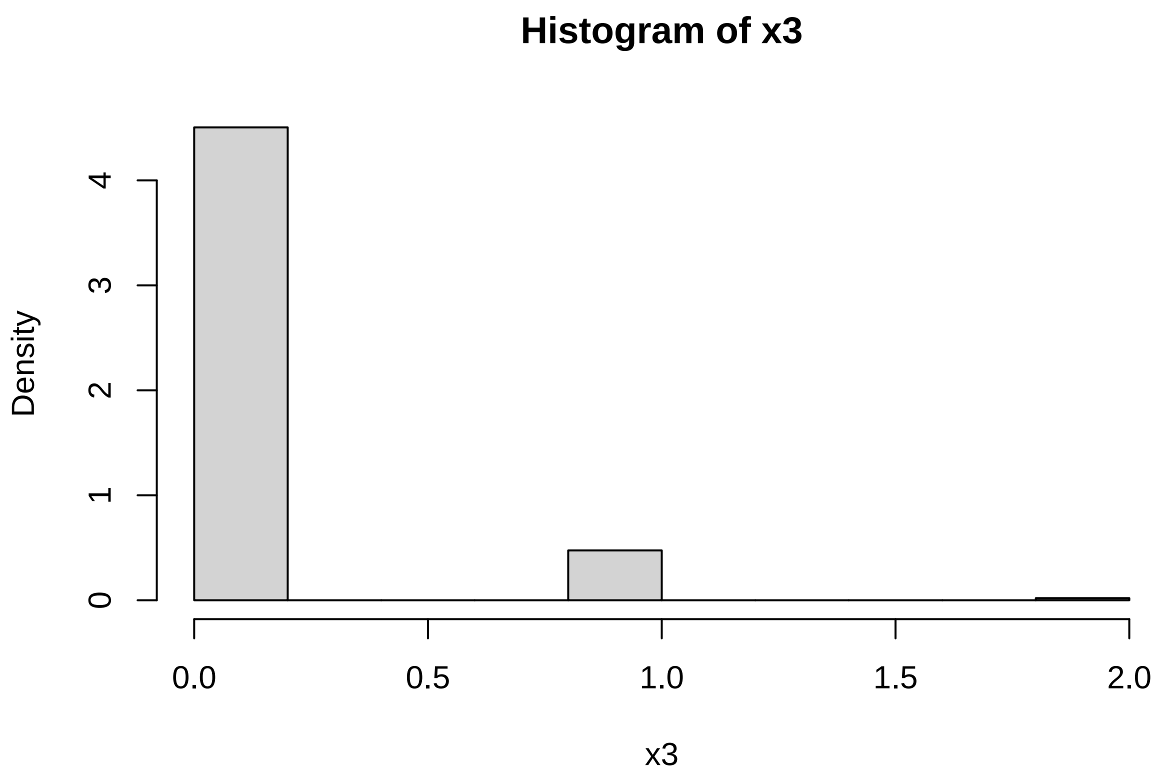
```
x1 <- rpois(1000, 5)
x2 <- rexp(1000, 1)
x3 <- rbinom(1000, 10, 0.01)

hist(x1, probability = T)
```




```
hist(x2, probability = T)
```

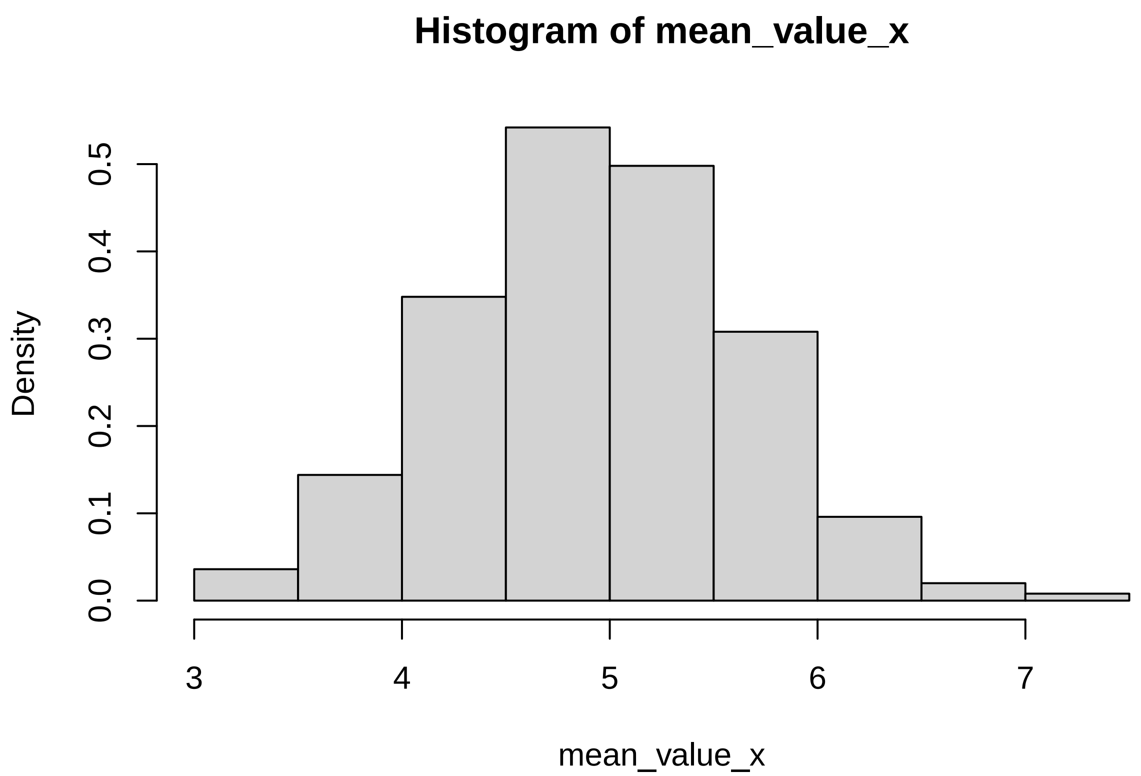



```
hist(x3, probability = T)
```

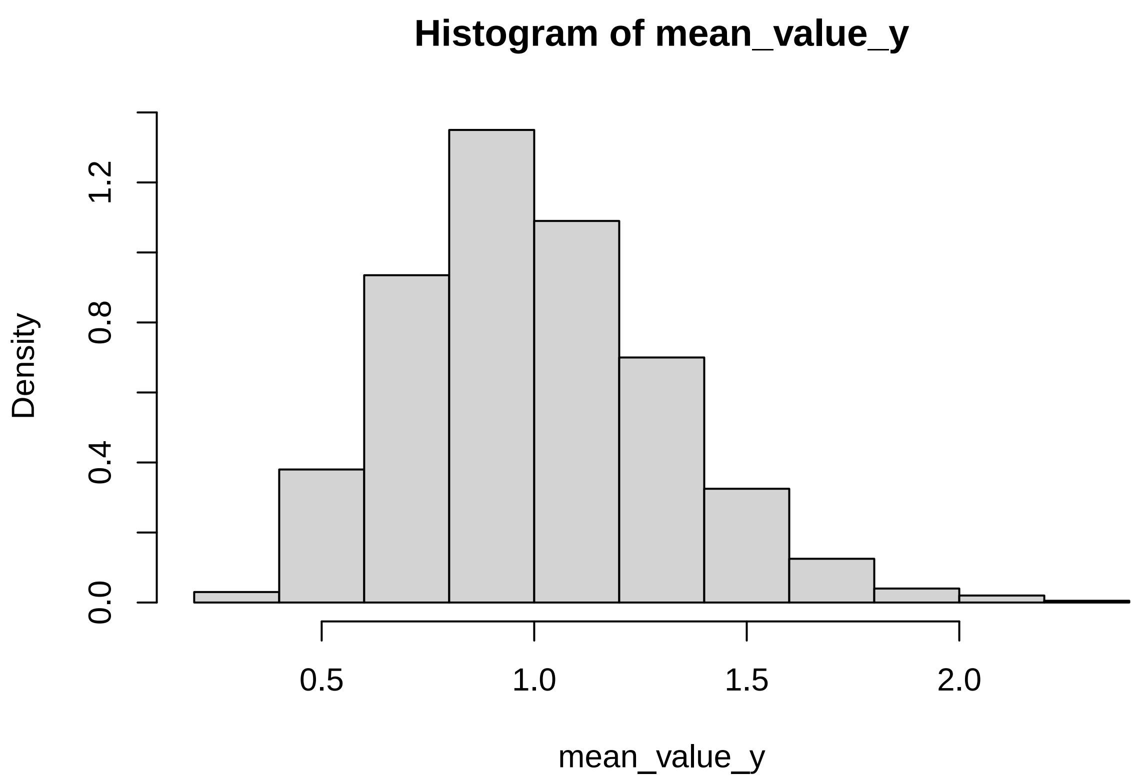


2.

```
mean_value_x <- numeric(0)
mean_value_y <- numeric(0)
for (i in 1:1000) {
  mean_value_x <- c(mean_value_x, mean(rpois(10, 5)))
  mean_value_y <- c(mean_value_y, mean(rexp(10, 1)))
}
hist(mean_value_x, probability = TRUE)
```



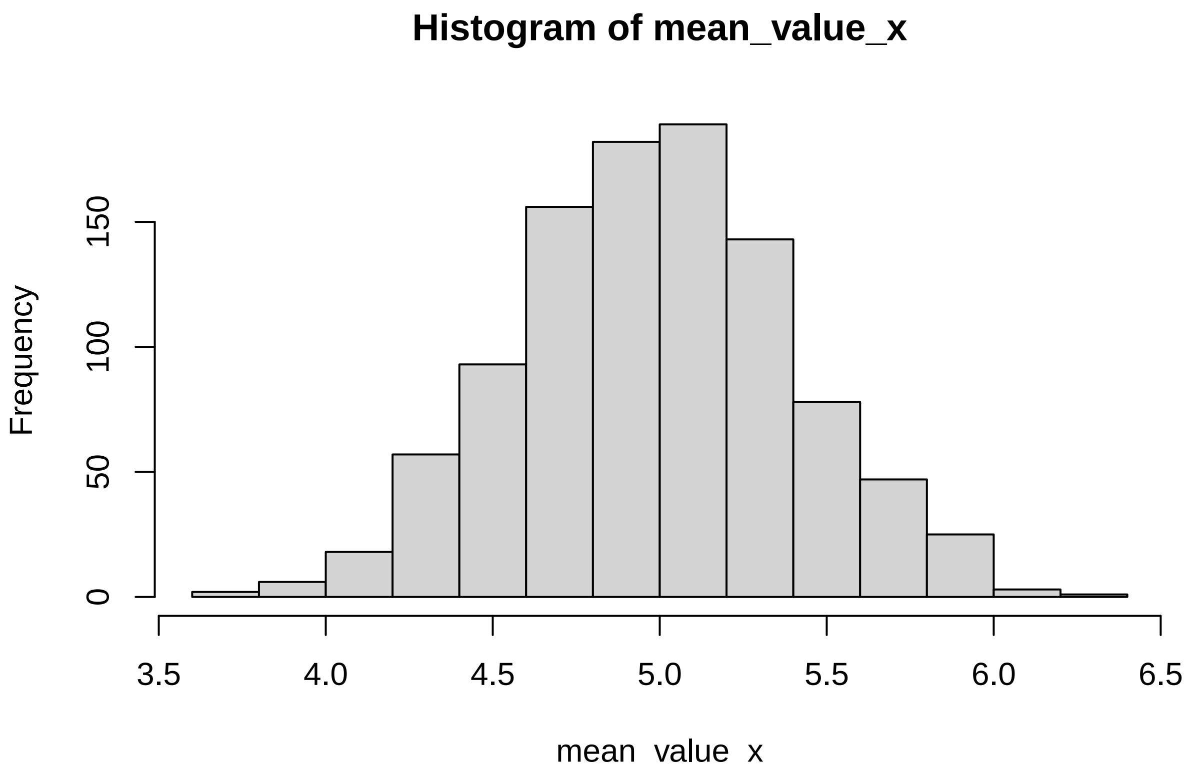

```
hist(mean_value_y, probability = TRUE)
```



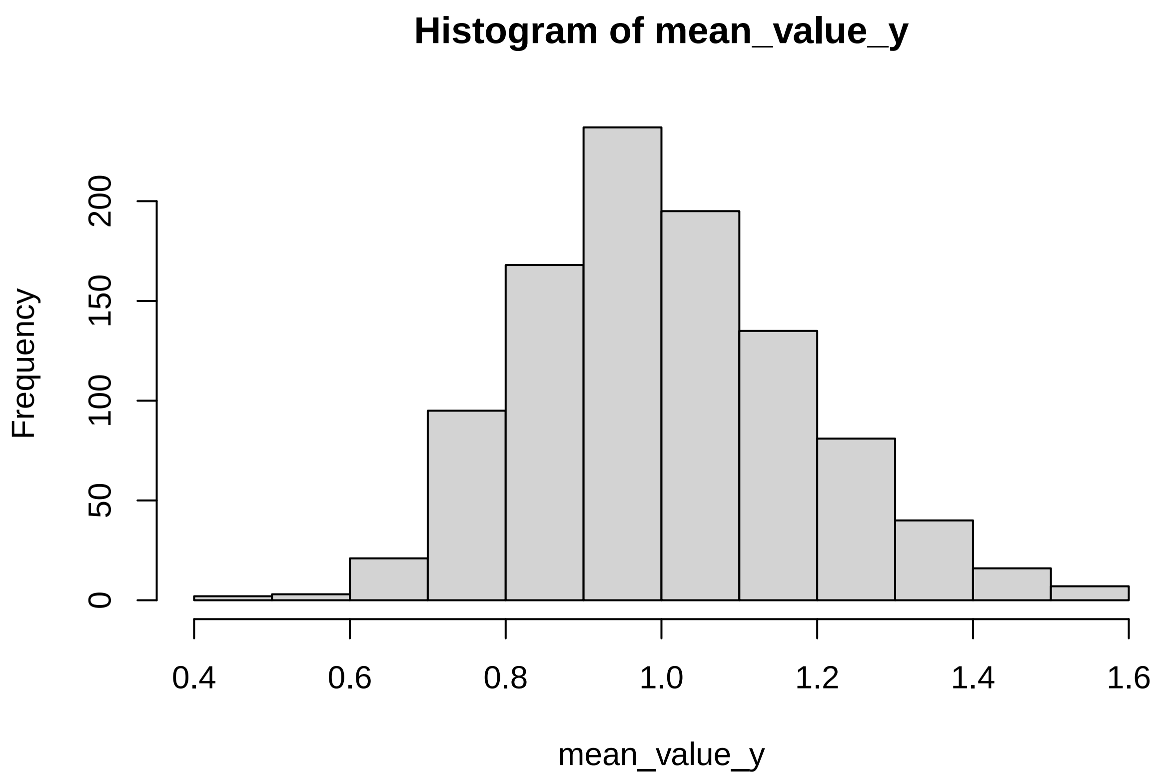
3.

för 30:

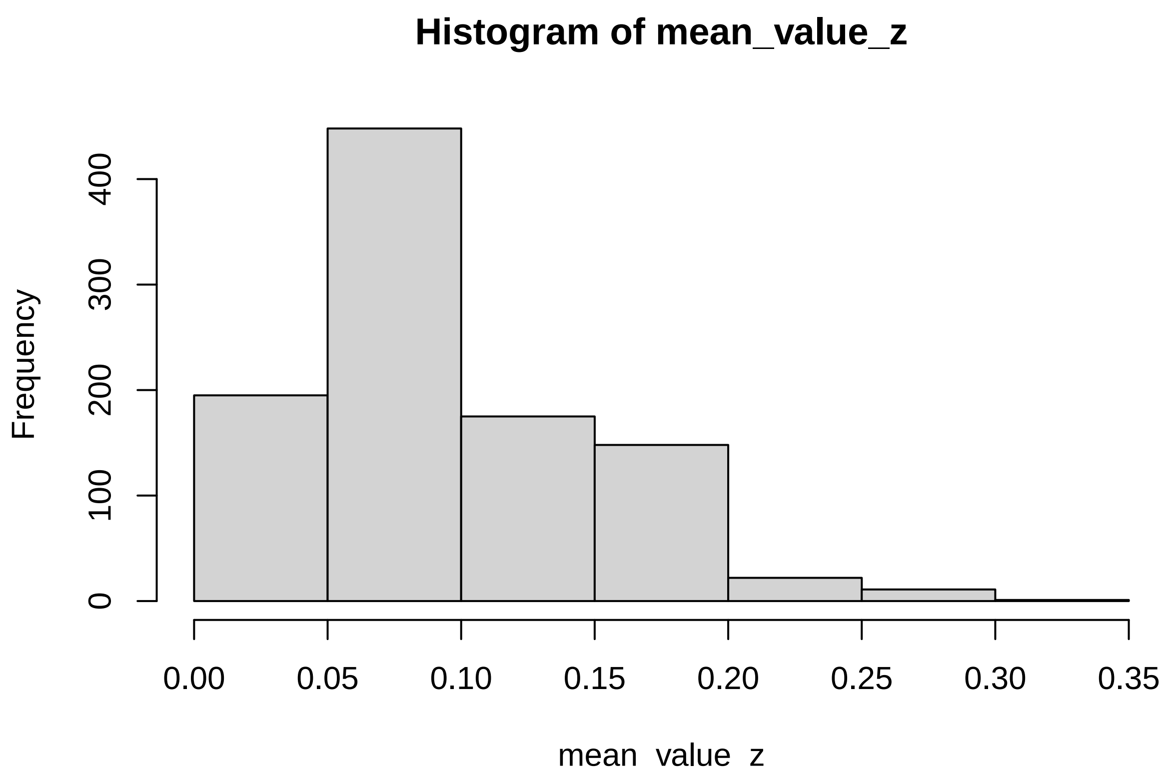
```r
mean_value_x <- numeric(0)
mean_value_y <- numeric(0)
mean_value_z <- numeric(0)
for (i in 1:1000) {
  mean_value_x <- c(mean_value_x, mean(rpois(30, 5)))
  mean_value_y <- c(mean_value_y, mean(rexp(30, 1)))
  mean_value_z <- c(mean_value_z, mean(rbinom(30, 10, 0.01)))
}
hist(mean_value_x)
```




```r
hist(mean_value_y)
```



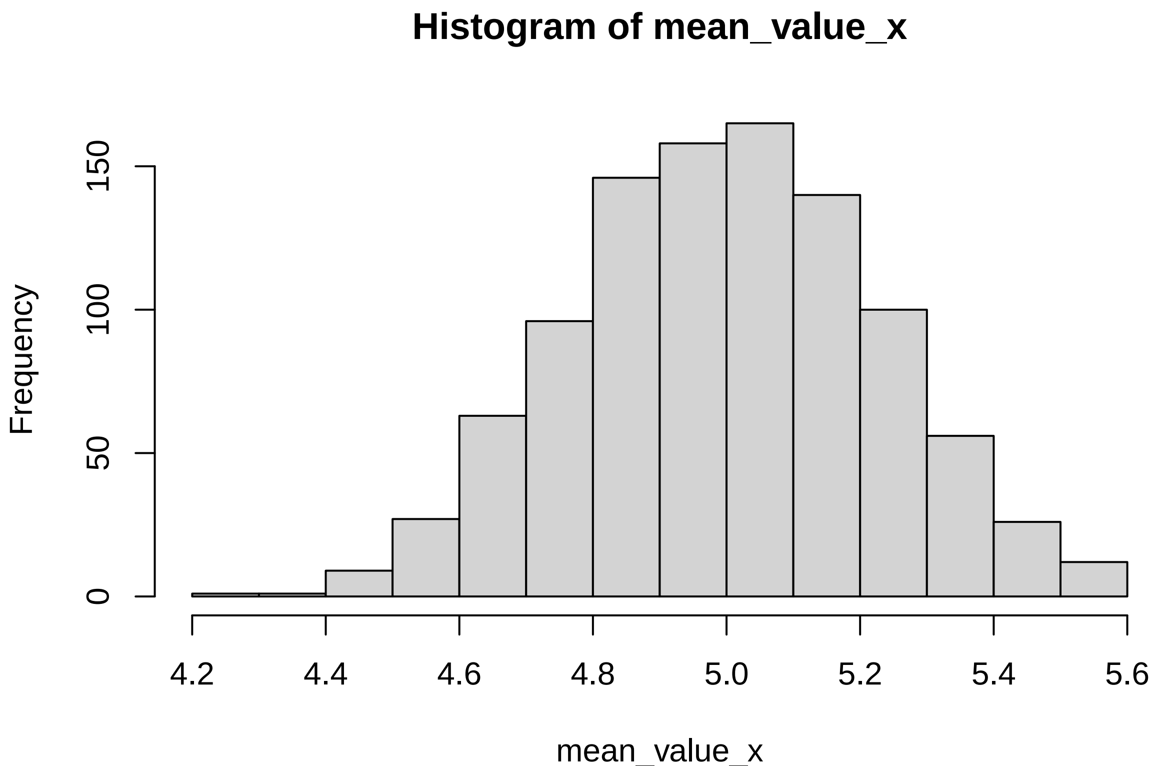

```
hist(mean_value_z)
```



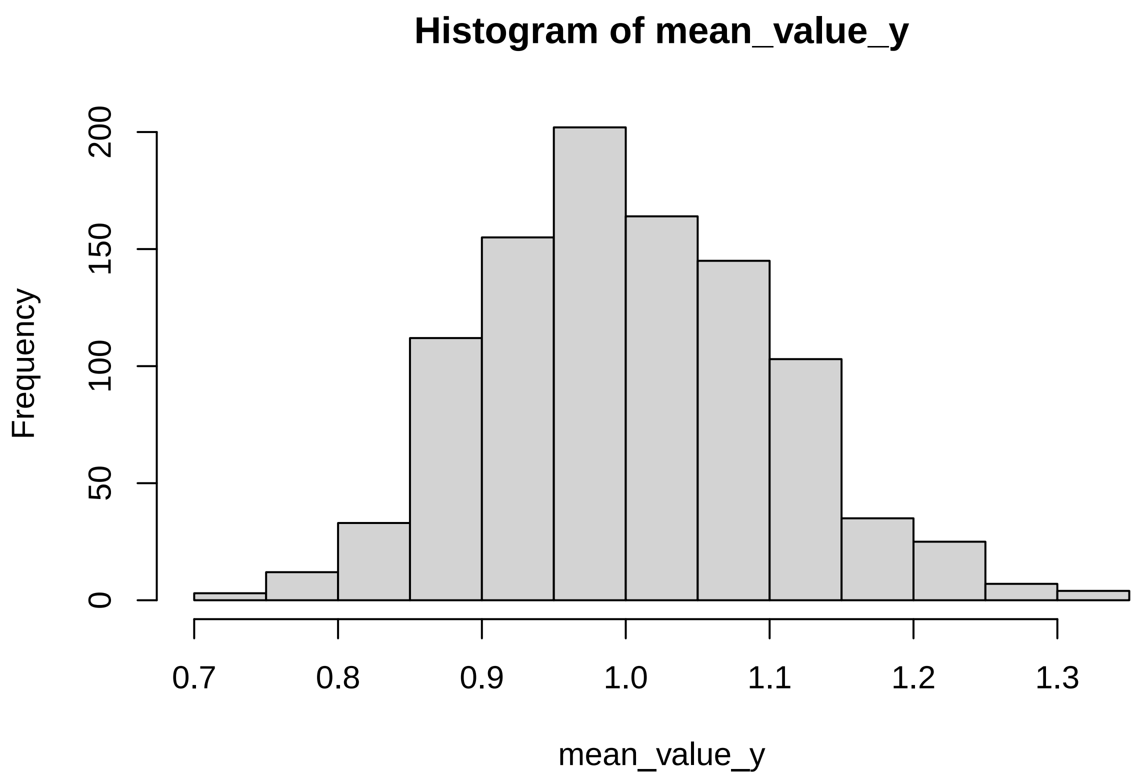
för 100:

```
mean_value_x <- numeric(0)
mean_value_y <- numeric(0)
```

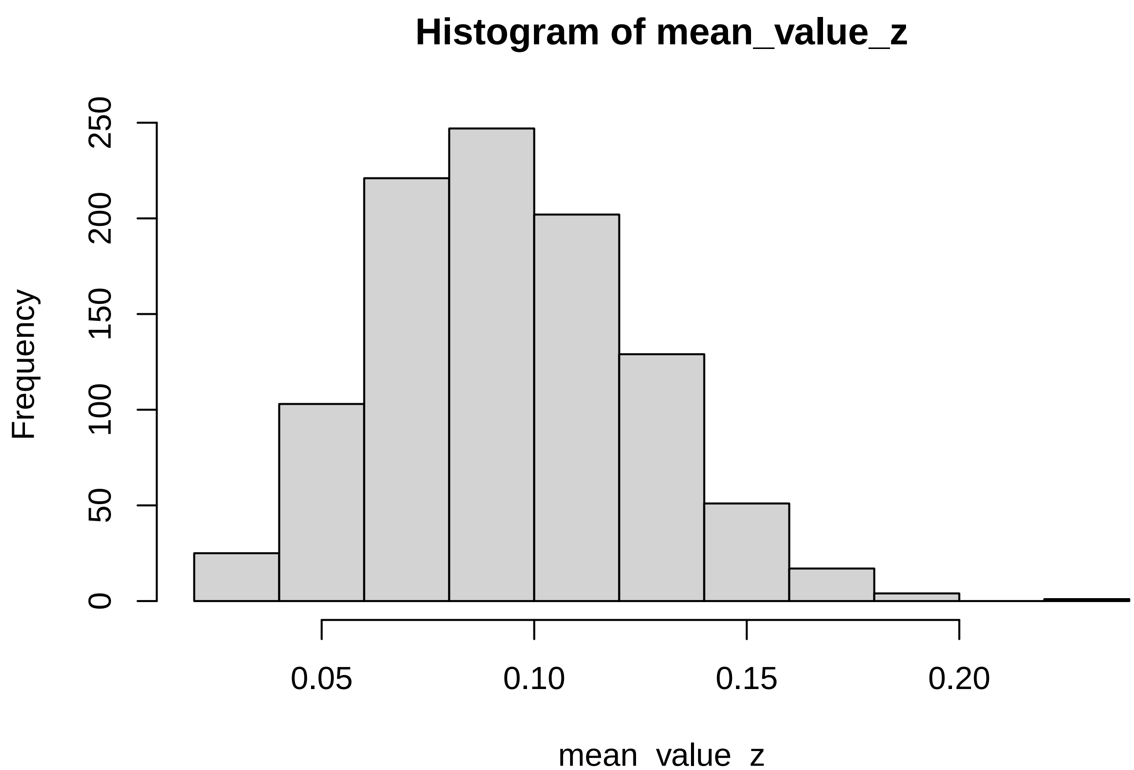
```
mean_value_z <- numeric(0)
for (i in 1:1000) {
  mean_value_x <- c(mean_value_x, mean(rpois(100, 5)))
  mean_value_y <- c(mean_value_y, mean(rexp(100, 1)))
  mean_value_z <- c(mean_value_z, mean(rbinom(100, 10, 0.01)))
}
hist(mean_value_x)
```




```
hist(mean_value_y)
```



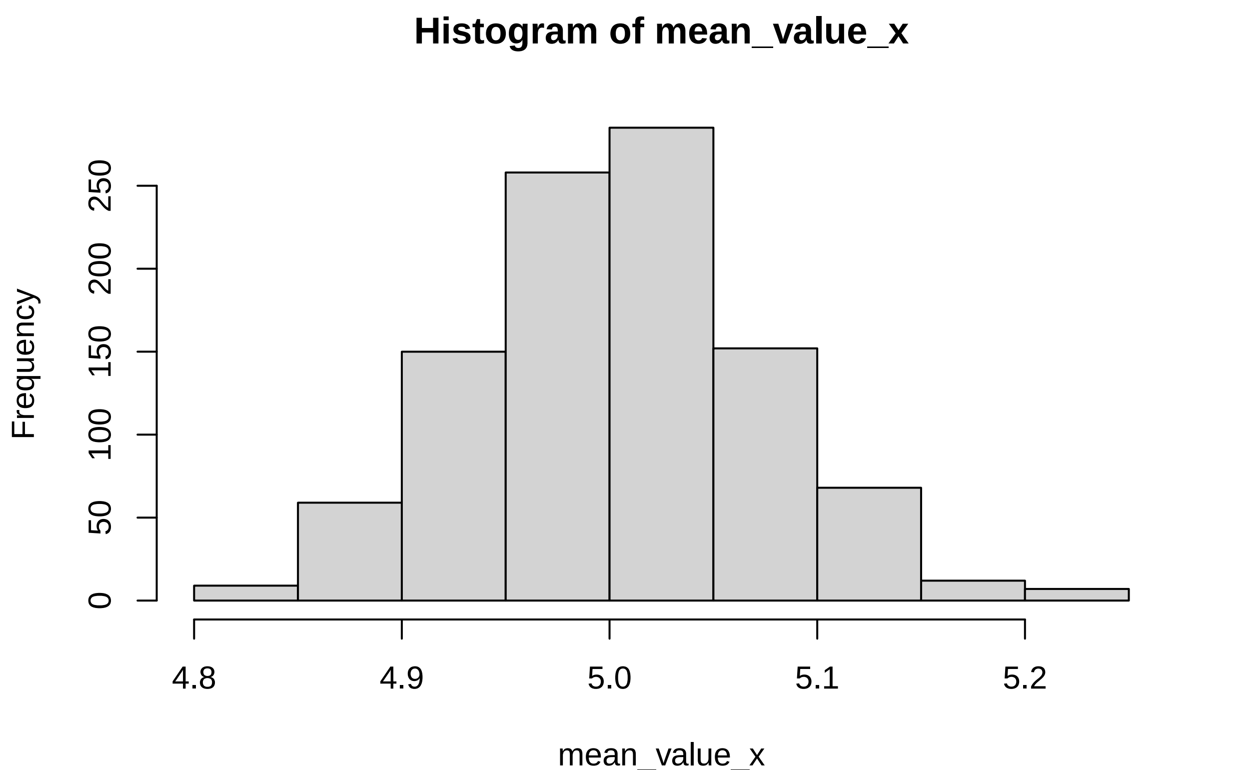

```
hist(mean_value_z)
```



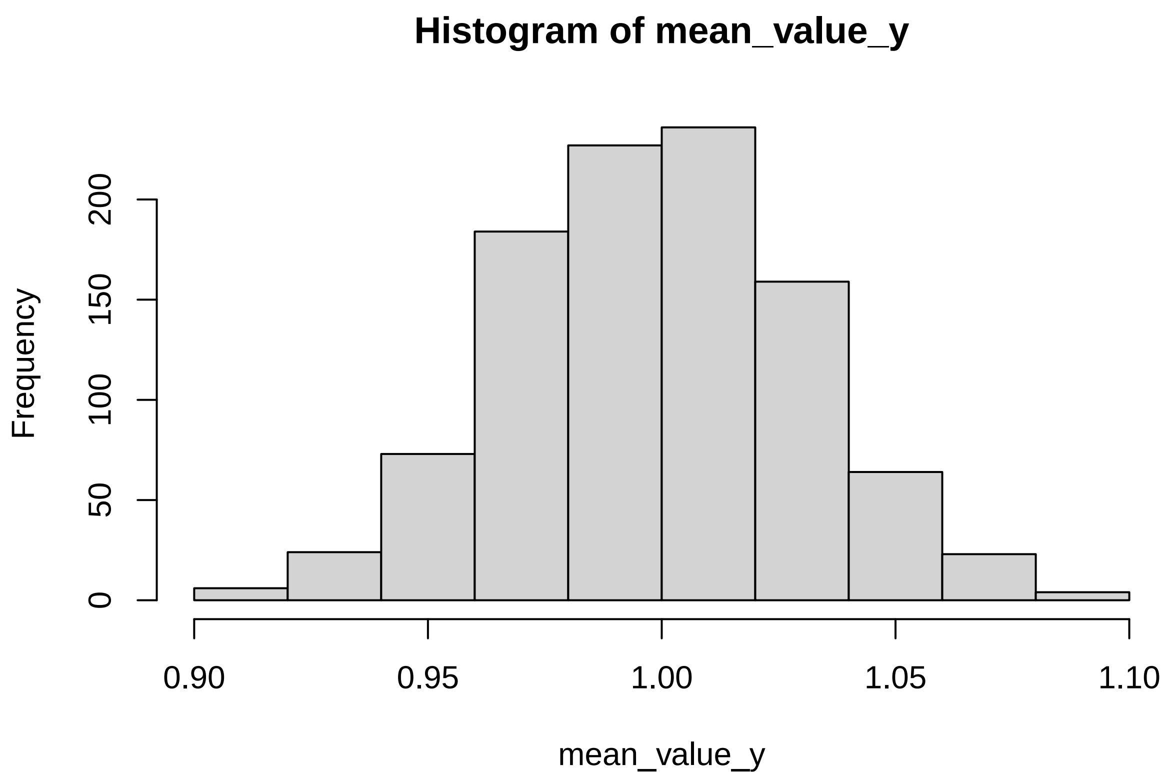
för 1000:

```
mean_value_x <- numeric(0)
mean_value_y <- numeric(0)
```

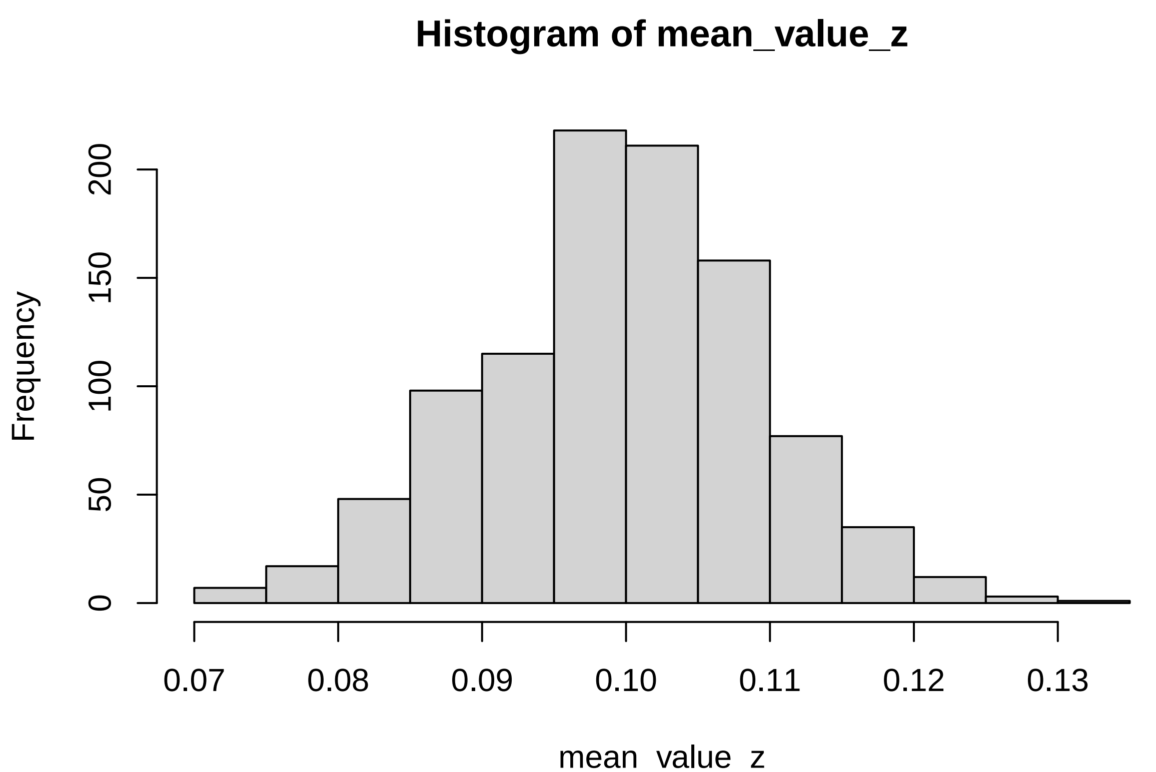
```r
mean_value_z <- numeric(0)
for (i in 1:1000) {
  mean_value_x <- c(mean_value_x, mean(rpois(1000, 5)))
  mean_value_y <- c(mean_value_y, mean(rexp(1000, 1)))
  mean_value_z <- c(mean_value_z, mean(rbinom(1000, 10, 0.01)))
}
hist(mean_value_x)
```




```r
hist(mean_value_y)
```




```
hist(mean_value_z)
```



Centrala gränsvärdessatsen får det att se ut som att medelvärdena verkar närma sig en normalfördelning, och detta verkar faktiskt stämma. Om man kollar på bilderna verkar det också inträffa baserade på mina simuleringar