Nicolai_ex2

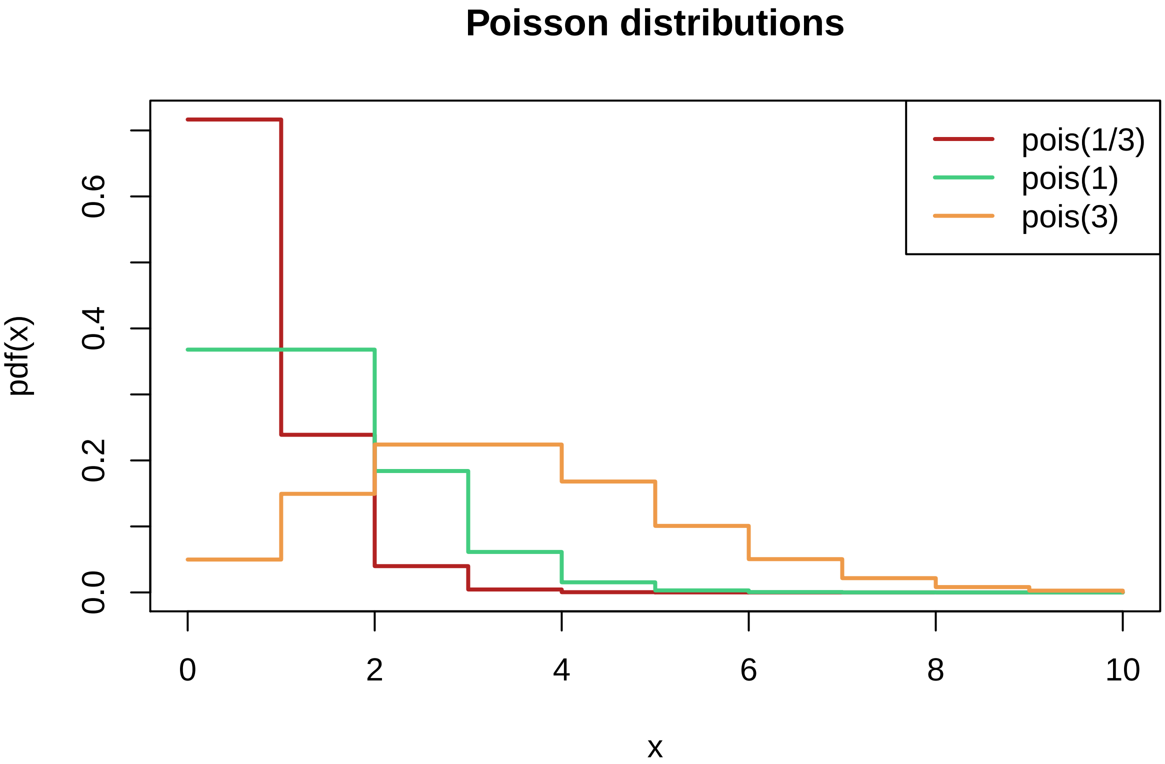
Andrea Nicolai

8/4/2020

Exercise 0

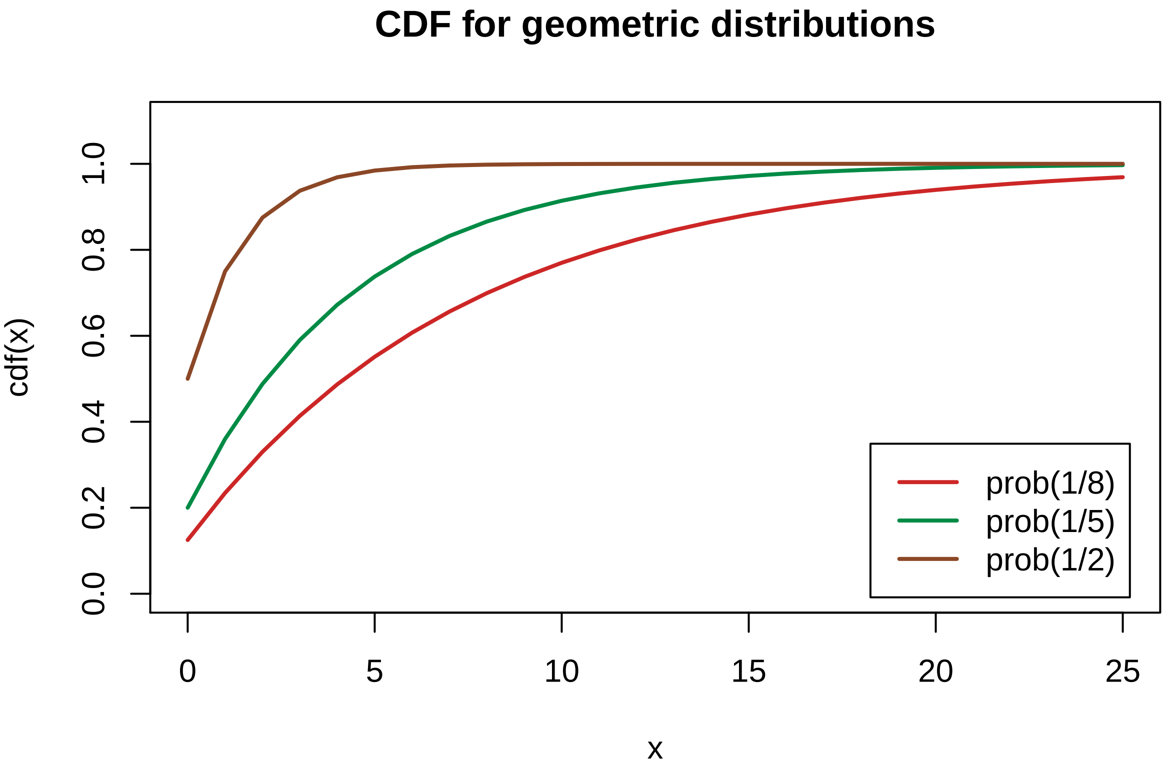
Let's plot events distributing according to a Poisson distribution for different values of the lambda parameter:

```r
x <- 0:10
plot(x, dpois(x, 1/3), col = 'firebrick', main = 'Poisson distributions', xlab = 'x', ylab = 'pdf(x)', 
lines(x, dpois(x, 1), col = 'seagreen3', type = 's', lwd = 2)
lines(x, dpois(x, 3), col = 'tan2', type = 's', lwd = 2)
legend(x = "topright", legend = c("pois(1/3)", "pois(1)", "pois(3)"), col = c("firebrick","seagreen3", 
```



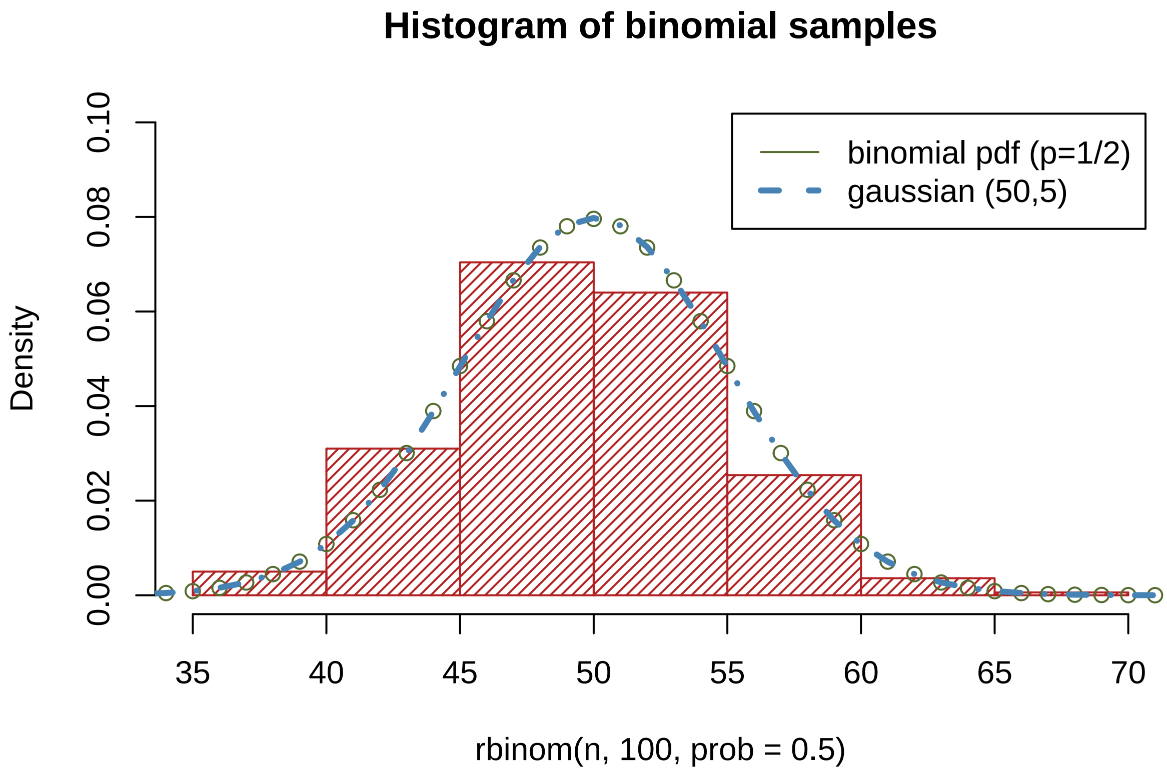
Let's plot some cdfs of geometric distributions for different prob variables:

```r
x <- 0:25
plot(x, pgeom(x, 1/8), col = 'firebrick3', main = 'CDF for geometric distributions', xlab = 'x', ylab =
lines(x, pgeom(x, 1/5), col = 'springgreen4', type = 'l', lwd = 2)
lines(x, pgeom(x, 1/2), col = 'sienna4', type = 'l', lwd = 2)
legend(x = "bottomright", inset = 0.03, legend = c("prob(1/8)", "prob(1/5)", "prob(1/2)"), col = c("fir
```



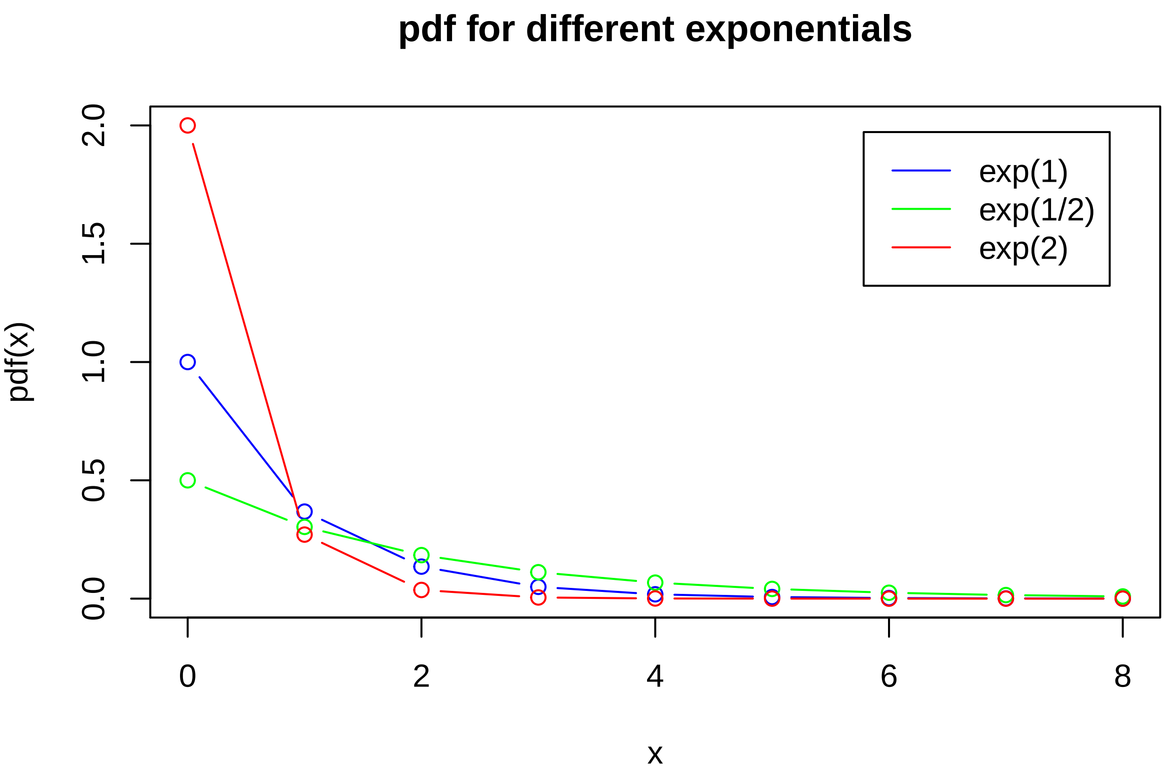
Let's draw some samples (1000) from the simmetric binomial distribution and superimpose the normal one and the pdf:

```
n <- 1000
hist(rbinom(n, 100, prob = 0.5), probability = TRUE, ylim = c(0, 0.10), col = "Firebrick", density =25,
points(0:75, dbinom(0:75, 100, prob = 0.5), col = "Dark Olive Green ")
lines(0:75, dnorm(0:75, mean = 50, sd = 5), col = "Steel Blue", lty = "dotdash", lwd=3)
legend(x = "topright",  legend=c("binomial pdf (p=1/2)", "gaussian (50,5)"), inset = 0.02, col = c("Dar
```



Let's plot some pdfs of exponential distributions for different values of the lambda parameter:

```
x <- 0:8
plot(x, dexp(x, rate = 1), col = 'blue', type = 'b', ylim = c(0, 2), main = "pdf for different exponenti
lines(x, dexp(x, rate = 1/2), col = 'green', type = 'b')
lines(x, dexp(x, rate = 2), col = 'red', type = 'b')
legend(x = "topright", legend = c("exp(1)", "exp(1/2)", "exp(2)"), inset=.05, col =c('blue','green','re
```



Exercise 1

```
x  <- c(15.58, 15.9, 16.0, 16.1, 16.2)
p1 <- c(0.15, 0.21, 0.35, 0.15, 0.14)
p2 <- c(0.14, 0.05, 0.64, 0.08, 0.09)
```

Evaluate the expected values, E[X]

```
E.1  <- sum(p1*x)
E.2  <- sum(p2*x)
```

And the variance, Var(X):

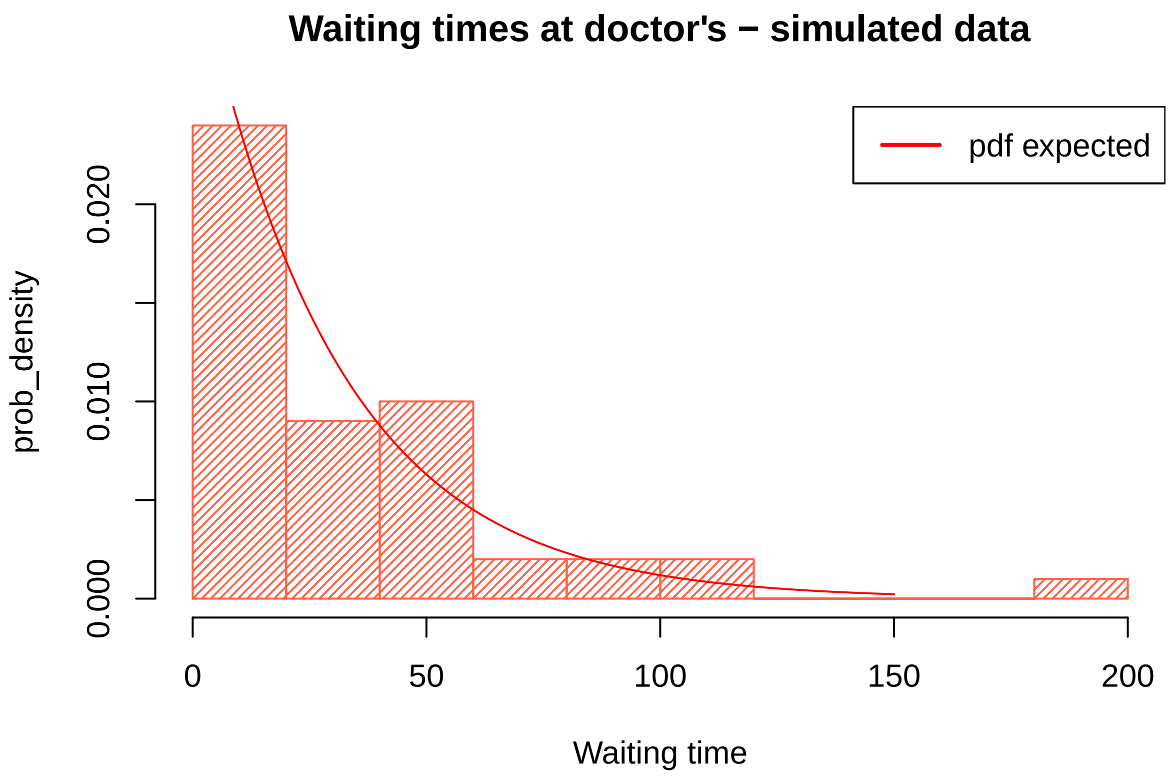
```
E2.1 <- sum(p1*x^2)
var1 <- E2.1 - E.1^2


E2.2 <- sum(p2*x^2)
var2 <- E2.2 - E.2^2
```

Exercise 2: The waiting time, in minutes, at the doctor's is about 30 minutes, and the distribution follows an exponential pdf with rate 1/30. Simulate the waiting time for 50 people at the doctor's office and plot the relative histogram:

```
x <- 0:150
exp_doctor <- dexp(x, rate = 1/30)

sim_waiting_times <- rexp( n = 50 , rate = 1/30 )
sim_waiting_times <- sort(sim_waiting_times , decreasing = TRUE)
hist(sim_waiting_times, main = "Waiting times at doctor's - simulated data", col = "tomato", density = 
lines(x, exp_doctor, col = 'red')
legend(x = "topright", c("pdf expected"), lwd=2, col="red")
```



What is the probability that a person will wait for less than 10 minutes ?

```
pexp( q = 10, rate = 1/30 )
```

```
## [1] 0.2834687
```

Evaluate the average waiting time from the simulated data and compare it with the expected value (calculated from theory and by manipulating the probability distributions using R):

```
mean.sim <- mean(sim_waiting_times)
mean.integral <- integrate(function (x) {(dexp(x,1/30)*x)}, 0, +Inf)$value

message(sprintf("Mean time by using simulated data is: mean = %.1f min", mean.sim))
```

```
## Mean time by using simulated data is: mean = 31.6 min
```

```
message(sprintf("Mean time by computing it from distributions is: mean = %.1f min", mean.integral))
```

```
## Mean time by computing it from distributions is: mean = 30.0 min
```

What is the probability for waiting more than one hour before being received ?

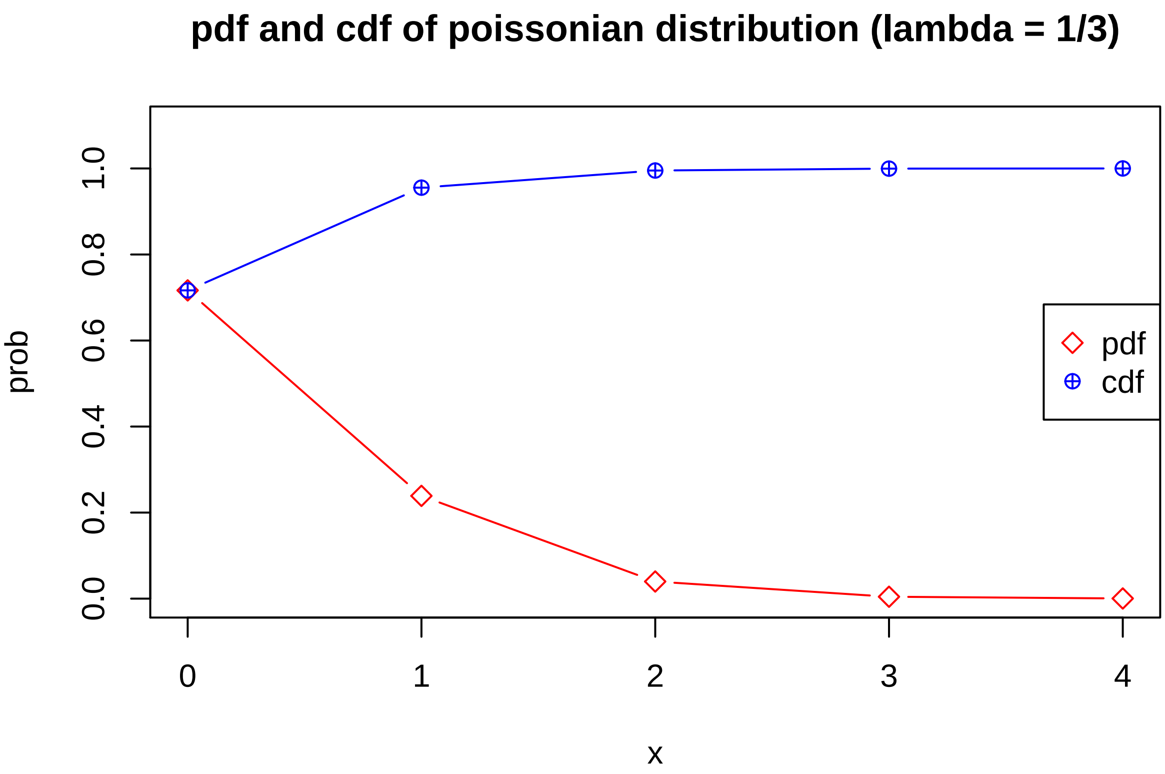
```
1 - pexp( q = 60, rate = 1/30 )
```

```
## [1] 0.1353353
```

Exercise 3

Let's suppose that on a book, on average, there is one typo error every three pages. If the number of errors follows a Poisson distribution, plot the pdf and cdf:

```
x <- 0:4
plot(x, dpois(x, lambda = 1/3), main = "pdf and cdf of poissonian distribution (lambda = 1/3)", col = "
points(x, ppois(q = x, lambda = 1/3), col = "blue", pch = 10, type = 'b')
legend( x = "right", c("pdf", "cdf") , col = c("red", "blue"), pch = c(5,10))
```



Calculate the probability that there is at least one error on a specific page of the book:

```
1 - ppois(0, lambda = 1/3 )
```

```
## [1] 0.2834687
```

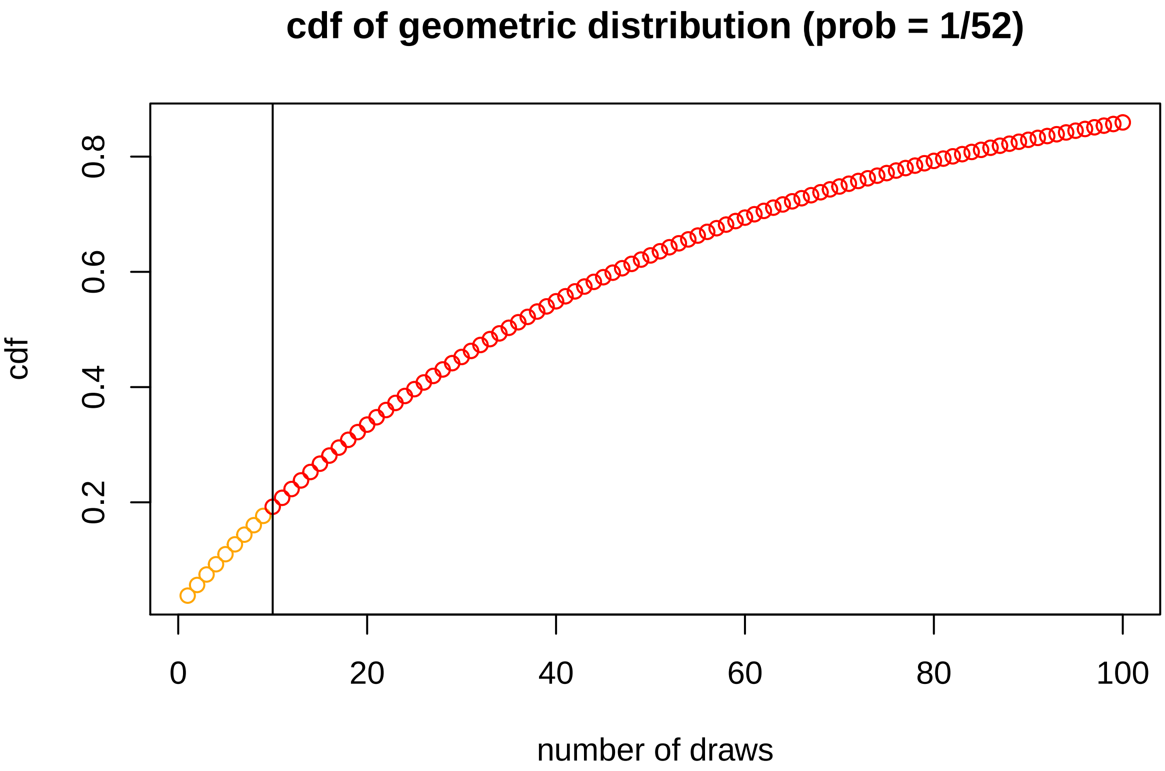
```
#or equivalently
1 - dpois(0, lambda = 1/3 )
```

```
## [1] 0.2834687
```

Exercise 4

We randomly draw cards from a deck of 52 cards, with replacement, until one ace is drawn. Calculate the probability that at least 10 draws are needed. Distribution that follows our event is the geometric one.

```
x <- 1:100
plot(x, pgeom(x, prob = 1/52), main = "cdf of geometric distribution (prob = 1/52)", col = "orange", yl
points(10:100, pgeom(10:100, prob = 1/52), col = "red")
abline(v = 10)
```



```
message(sprintf("Probability that at least 10 draws are needed is: %.2f", 1 - pgeom(9 , prob = 1/52)))
```

```
## Probability that at least 10 draws are needed is: 0.82
```

Exercise 4 Open R and import the file in a tibble or data.frame:

```
library("tibble")
library("tidyverse")
```

```
## -- Attaching packages ------------------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## v purrr   0.3.3
```

```
## -- Conflicts ---------------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

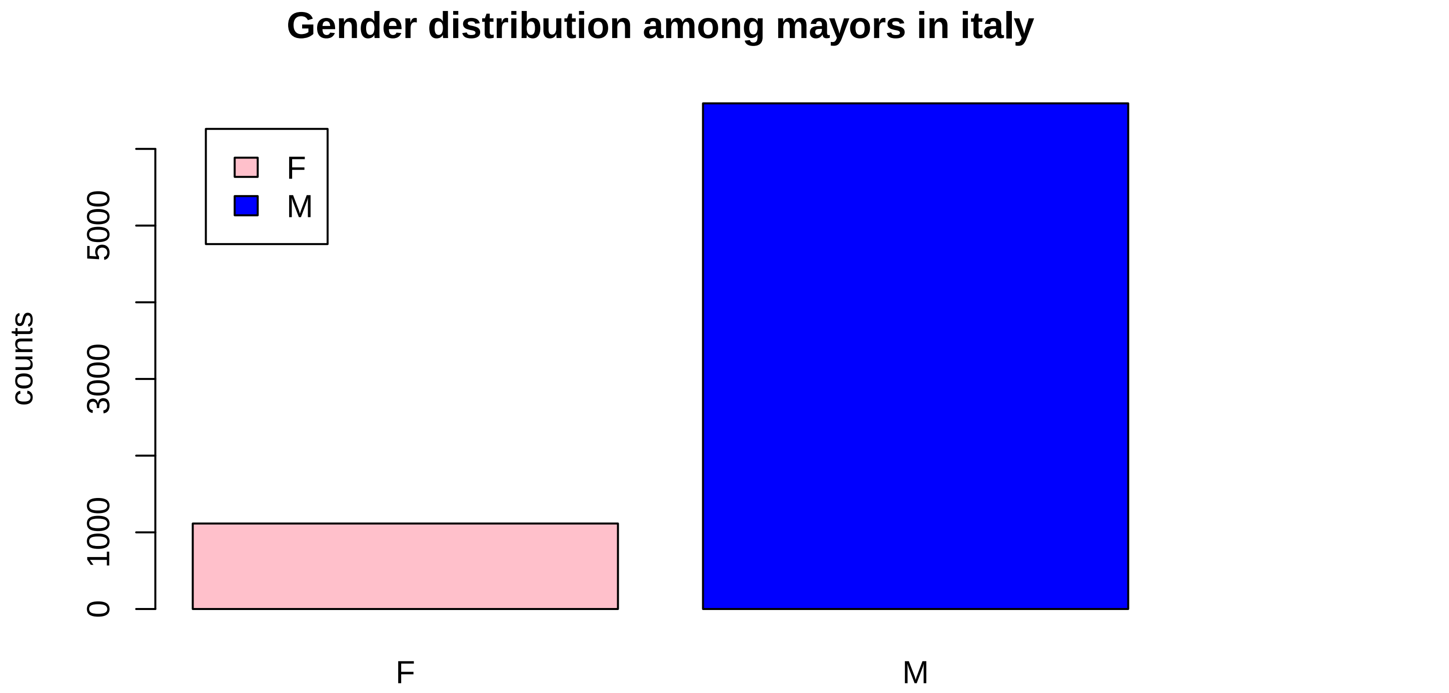
```
mayors <- read_csv2( "sindaciincarica.csv" , skip = 2)
```

```
## Using ',' as decimal and '.' as grouping mark. Use read_delim() for more control.
```

```
## Parsed with column specification:
## cols(
##   codice_regione = col_character(),
##   codice_provincia = col_character(),
##   codice_comune = col_character(),
##   denominazione_comune = col_character(),
##   sigla_provincia = col_character(),
##   popolazione_censita = col_double(),
##   titolo_accademico = col_character(),
##   cognome = col_character(),
##   nome = col_character(),
##   sesso = col_character(),
##   data_nascita = col_character(),
##   luogo_nascita = col_character(),
##   descrizione_carica = col_character(),
##   data_elezione = col_character(),
##   data_entrata_in_carica = col_character(),
##   partito = col_character(),
##   titolo_studio = col_character(),
##   professione = col_character()
## )
```

Plot the gender distribution among the mayors (column name sesso)

```
barplot(table(mayors$sesso), col = c('pink','blue'), main = "Gender distribution among mayors in italy"
```



Plot the number of towns grouped per province (codice provincia) and per region (codice regione):

```
province_region <- data.frame(mayors$codice_provincia, mayors$codice_regione) %>% group_by( mayors.codi
colnames(province_region) <- c("region", "province", "counts")

province_region$region<- as.character(province_region$region)
province_region$region[province_region$region == "01"] <- "Piemonte"
province_region$region[province_region$region == "02"] <- "Valle d'Aosta"
province_region$region[province_region$region == "03"] <- "Lombardia"
province_region$region[province_region$region == "04"] <- "Trentino Alto Adige"
province_region$region[province_region$region == "05"] <- "Veneto"
province_region$region[province_region$region == "06"] <- "FVG"
province_region$region[province_region$region == "07"] <- "Liguria"
province_region$region[province_region$region == "08"] <- "Emilia Romagna"
province_region$region[province_region$region == "09"] <- "Toscana"
province_region$region[province_region$region == "10"] <-"Umbria"
province_region$region[province_region$region == "11"] <-"Marche"
```

```
province_region$region[province_region$region == "12"] <-"Lazio"
province_region$region[province_region$region == "13"] <-"Abruzzo"
province_region$region[province_region$region == "14"] <-"Molise"
province_region$region[province_region$region == "15"] <-"Campania"
province_region$region[province_region$region == "16"] <-"Puglia"
province_region$region[province_region$region == "17"] <-"Basilicata"
province_region$region[province_region$region == "18"] <-"Calabria"
province_region$region[province_region$region == "19"] <-"Sicilia"
province_region$region[province_region$region == "20"] <-"Sardegna"

province_region$region <- as.factor(province_region$region)
```

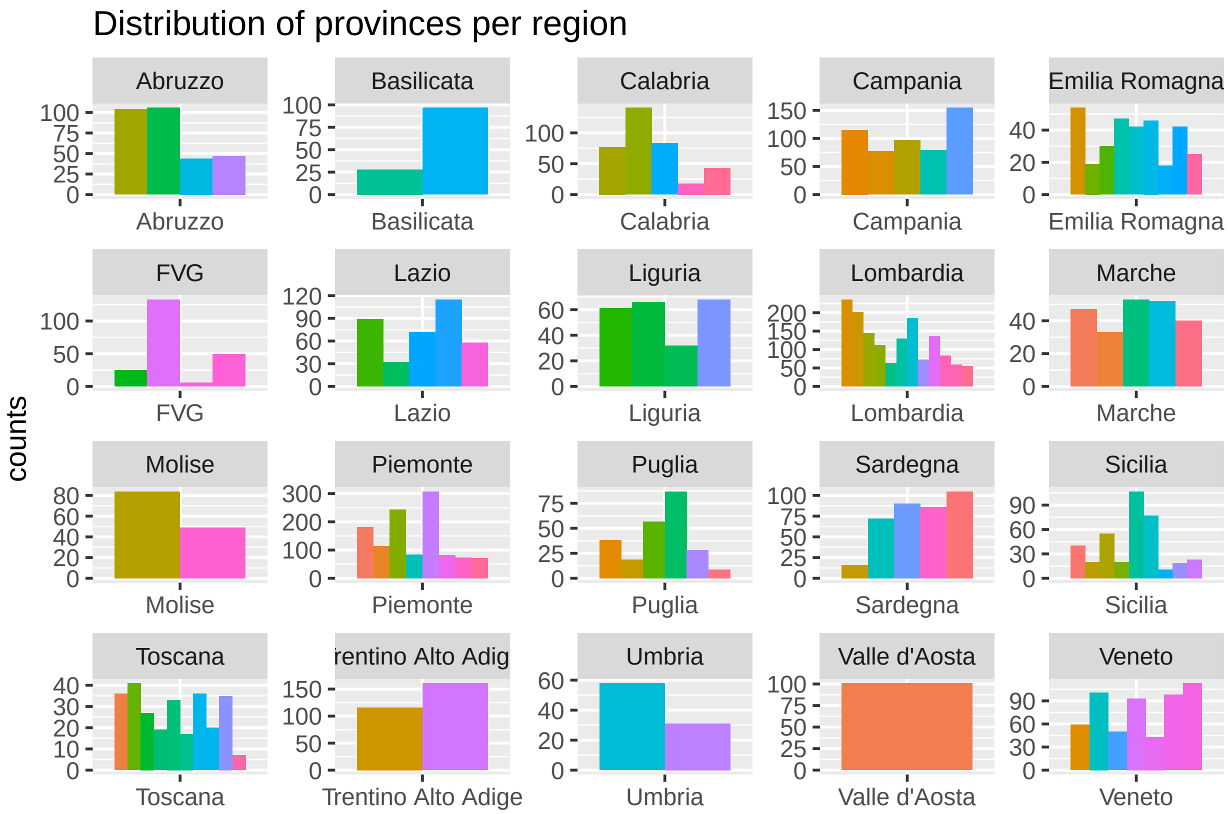
Numbers of town for different regions are the different

```
library("ggplot2")
library(viridis)
```

```
## Loading required package: viridisLite
```

```
province_region_plot <- ggplot(province_region, aes(fill=province, y=counts, x=region)) + geom_bar(posi

province_region_plot
```

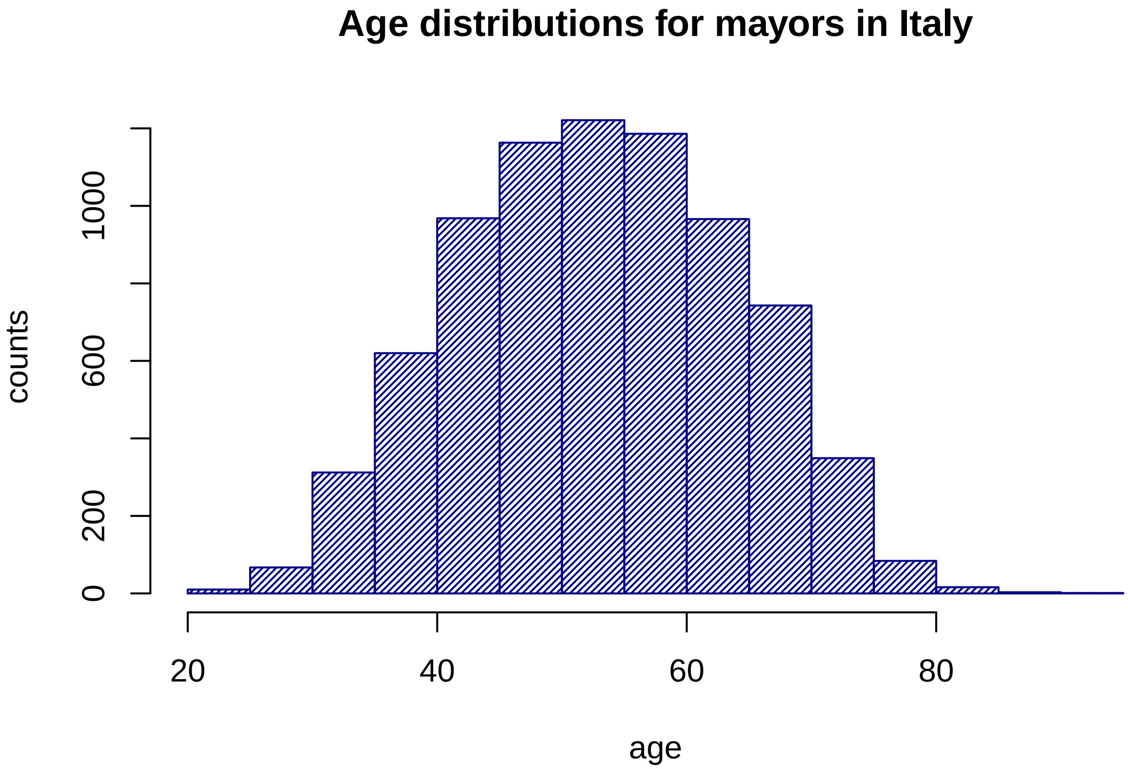


Plot a distributions of the age (years only) of the mayors. In the data nascita column the birthday is available:

```
library("lubridate")
```

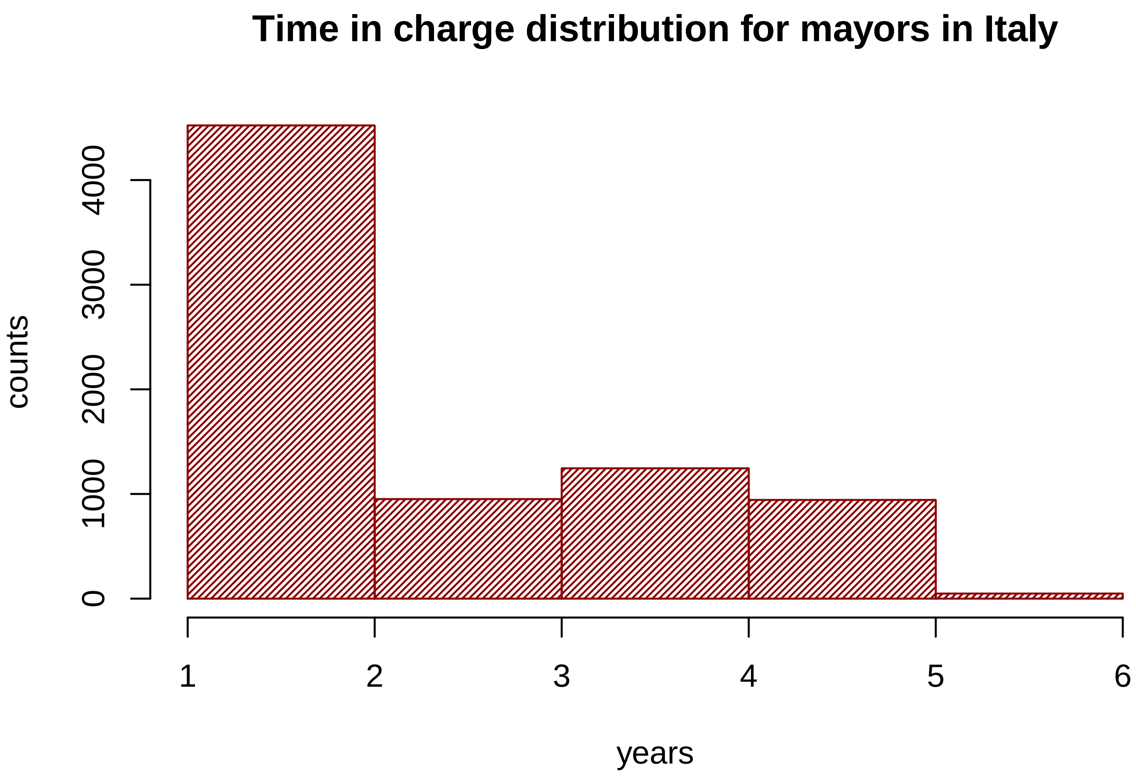
```
##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##     date
```

```
year_birth <- format(as.Date(mayors$data_nascita, "%d/%m/%Y"), "%Y")
year_today <- format(lubridate::today(), "%Y")
age <- as.integer(year_today) - as.integer(year_birth)
hist(age, main = "Age distributions for mayors in Italy", xlab = "age", ylab = "counts", col = "darkblu
```



Plot a distribution of the time the mayor is in charge. The starting date is in column data elezione.

```
year_charge <- format(as.Date(mayors$data_elezione, "%d/%m/%Y"), "%Y")
time_in_charge <- as.integer(year_today) - as.integer(year_charge)
hist(time_in_charge, main = "Time in charge distribution for mayors in Italy", xlab = "years", ylab = "
```



Since elections happen every 5 years, how many of them are going to complete their mandate this year ?

```
duration_of_duty <- 5
length(time_in_charge[time_in_charge >= duration_of_duty])
```

```
## [1] 991
```

```
#there are some mayors that have been in charge for 6 years...we think that they will partecipate to el
```

And how many in 2021 ?

```
duration_of_duty <- 5
length(time_in_charge[time_in_charge == (duration_of_duty - 1)])
```

```
## [1] 1245
```