Lab Tutorial 1: First Steps in R

Complete all of the following questions, adding your inputs as code chunks (enclose within triple accent marks) within Rmarkdown.

The exercises are not marked and will not be factored into your course grade, but it is important to complete them to make sure you have the skills to answer assessment questions. You may consult any resource, including other students and the instructor. Please Knit this document to a PDF and upload your work via Canvas at the end of the session. Solutions will be posted for you to check your own answers.

Basic mathematics and variables

1. Represent the fraction 2/9 as a decimal.

```r
2/9
```

```
## [1] 0.2222222
```

2. Calculate the sum of 3141, 541, and 6588.

```r
3141+541+6588
```

```
## [1] 10270
```

3. Calculate the square root and cube root of 7.

```r
sqrt(7)
```

```
## [1] 2.645751
```

```r
7^(1/3)
```

```
## [1] 1.912931
```

4. Round the value of the mathematical constant e to three decimal places. (Note: e = exp(1))

```r
round(exp(1), 3)
```

```
## [1] 2.718
```

5. Calculate the natural logarithm of the following values: 10, 51, 20.14, π, 0, -1

```r
log(10)
```

```
## [1] 2.302585
```

```r
log(51)
```

```
## [1] 3.931826
```

```r
log(20.14)
```

```
## [1] 3.002708
```

```r
log(pi)
```

```
## [1] 1.14473
```

```r
log(0)
```

```
## [1] -Inf
```

```r
log(-1)
```

```
## Warning in log(-1): NaNs produced
```

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

6. Create two variables a and b, where a = 5.0538 and b = 4.9472. Then, calculate a+b.

```r
a = 5.0538
b = 4.9472
a+b
```

```
## [1] 10.001
```

7. Calculate the value that is twice the sum of a and b.

```r
(a+b)^2
```

```
## [1] 100.02
```

8. Is -1/a greater than -1/b ? (Use a logical expression.)

```r
-1/a > -1/b
```

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

9. Suppose I claim that, for the values of a and b above, a^{90} (a to the 90th power) is greater than b^{91} but less than b^{92}. Is this statement TRUE or FALSE?

```r
(a^90 > b^91) & (a^90 < b^92)
```

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

10. Confirm that the following mathematical statements are all TRUE (according to R):
    a. Two plus two is equal to four.
    b. If n is an odd-numbered integer, then the cosine of π times n is equal to -1.
    c. If x is a real number, then e^{-x} is equal to $1/e^x$. (You do not have to prove these statements for all values of n and x: choose a specific value of each, and confirm that the equality works.)

```r
2+2 ==4
```

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

```r
cos(pi)^3 ==-1
```

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

```r
e <- exp(1)
e^-1==1/e^1
```

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

Vectors

11. Create and print out a vector (call it U) containing the values 6, 9, 5, and 11, in that order. Use `c()`.

```r
u<- c(6,9,5,11)
u
```

```
## [1]  6  9  5 11
```

12. Calculate the square roots of 6, 9, 5, and 11 using a single operation.

```r
sqrt(u)
```

```
## [1] 2.449490 3.000000 2.236068 3.316625
```

13. Calculate the sum of the values in U. Use `sum()`.

```r
sum(u)
```

```
## [1] 31
```

14. Print the elements of this vector in sorted ascending order. Use `sort()`.

```r
sort(u)
```

```
## [1]  5  6  9 11
```

15. Print the elements of this vector in sorted descending order.

```r
rev(sort(u))
```

```
## [1] 11  9  6  5
```

16. Print out a vector that is 30 items long and simply repeats the value 7. Use `rep()`.

```r
rep(7, 30)
```

```
##  [1] 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
```

17. Print out a sequential vector of all the integers between 1 and 10 (inclusive).

```r
v <- seq(1:10)
v
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

18. Create and print out a numeric vector (call it V) where the lowest value is 0, the highest value is 10, and the values in the vector are uniformly spaced with an interval of 0.1. Use `seq()`.

```r
v= seq(0, 10, 0.1)
v
```

```
##   [1]  0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0  1.1  1.2  1.3  1.4
##  [16]  1.5  1.6  1.7  1.8  1.9  2.0  2.1  2.2  2.3  2.4  2.5  2.6  2.7  2.8  2.9
##  [31]  3.0  3.1  3.2  3.3  3.4  3.5  3.6  3.7  3.8  3.9  4.0  4.1  4.2  4.3  4.4
##  [46]  4.5  4.6  4.7  4.8  4.9  5.0  5.1  5.2  5.3  5.4  5.5  5.6  5.7  5.8  5.9
##  [61]  6.0  6.1  6.2  6.3  6.4  6.5  6.6  6.7  6.8  6.9  7.0  7.1  7.2  7.3  7.4
##  [76]  7.5  7.6  7.7  7.8  7.9  8.0  8.1  8.2  8.3  8.4  8.5  8.6  8.7  8.8  8.9
##  [91]  9.0  9.1  9.2  9.3  9.4  9.5  9.6  9.7  9.8  9.9 10.0
```

19. How long is this vector? (How many elements does it have?) Use `length()`.

```r
length(v)
```

```
## [1] 101
```

20. What is the value of the 1st element of this vector?

```r
v[1]
```

```
## [1] 0
```

21. What is the value of the 3rd element of this vector?

```r
v[3]
```

```
## [1] 0.2
```

22. What are the values of the 31st through 37th elements of this vector?

```r
v[31:37]
```

```
## [1] 3.0 3.1 3.2 3.3 3.4 3.5 3.6
```

23. What are the values of the 10th and 20th elements of this vector? (Use a single expression.)

```r
v[c(10, 20)]
```

```
## [1] 0.9 1.9
```

Logical operations on vectors and conditional subscripting

24. Create the new vector W as a copy of V. Confirm that the first five elements of W are equal to the first five elements of V.

```r
w= v
w[1:5]==v[1:5]
```

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

Note: if you make a mistake in problems 25-32 you may need to return here and re-run the chunk above.

25. Change the value of the first element of W to -1. Then confirm that the first element of W is now *not* equal to the first element of V.

```r
w[1]=-1
w[1] == v[1]
```

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

26. One of the elements of W is equal to 1.5. Which is it (what is its index)? Use `which()` and a logical expression.

```r
which(w> 1.5)
```

```
##  [1]  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35
## [20]  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
## [39]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73
## [58]  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92
## [77]  93  94  95  96  97  98  99 100 101
```

27. Change the element of W that currently has a value of 1.5 so that it has a value of 2.5 instead.

```r
w[w==1.5]=2.5
which(w== 1.5)
```

```
## integer(0)
```

28. Confirm that there are now two different elements of W which have a value of 2.5.

```r
which(w==2.5)
```

```
## [1] 16 26
```

```r
w[16]
```

```
## [1] 2.5
```

29. What are the subscripts (indices) of the elements in the vector W for which 6.55 < W < 6.75? Use `which()`.

```r
w[which(w>6.55) &(w<6.75 )]
```

```
## Warning in which(w > 6.55) & (w < 6.75): longer object length is not a multiple
## of shorter object length
```

```
##  [1] -1.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0  1.1  1.2  1.3  1.4
## [16]  2.5  1.6  1.7  1.8  1.9  2.0  2.1  2.2  2.3  2.4  2.5  2.6  2.7  2.8  2.9
## [31]  3.0  3.1  3.2  3.3  3.4  3.5  3.6  3.7  3.8  3.9  4.0  4.1  4.2  4.3  4.4
## [46]  4.5  4.6  4.7  4.8  4.9  5.0  5.1  5.2  5.3  5.4  5.5  5.6  5.7  5.8  5.9
## [61]  6.0  6.1  6.2  6.3  6.4  6.5  6.6  6.7
```

30. Change all of the values of W which are currently between 6.55 and 6.75 to be equal to zero instead.

```r
w[(w>=6.55) &(w==6.75 )]=0
w[w==6.55]
```

```
## numeric(0)
```

31. Change all the values of W which are greater than 8 to be equal to zero instead.

```
w[w>8]=0
```

32. Print all the values of W and confirm they are what you expect: the values that were formerly 1.5, between 6.55 and 6.75, and above 8.0 should now be set to 2.5, 0.0, and 0.0 respectively.

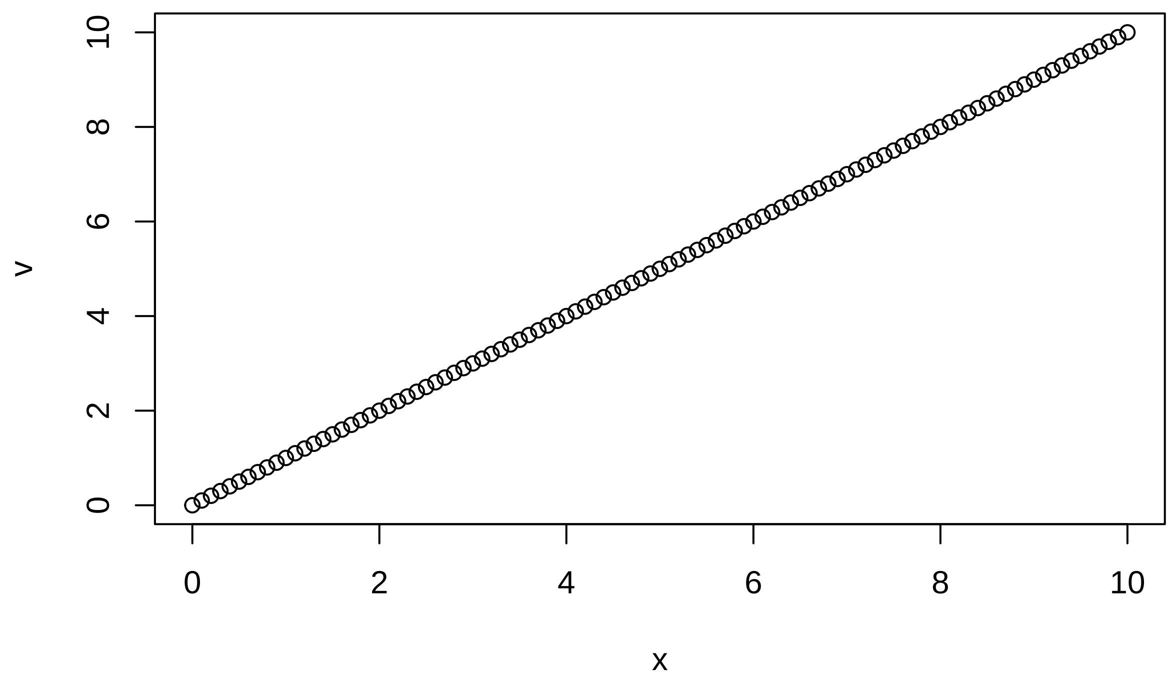
```
w
```

```
##   [1] -1.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0  1.1  1.2  1.3  1.4
##  [16]  2.5  1.6  1.7  1.8  1.9  2.0  2.1  2.2  2.3  2.4  2.5  2.6  2.7  2.8  2.9
##  [31]  3.0  3.1  3.2  3.3  3.4  3.5  3.6  3.7  3.8  3.9  4.0  4.1  4.2  4.3  4.4
##  [46]  4.5  4.6  4.7  4.8  4.9  5.0  5.1  5.2  5.3  5.4  5.5  5.6  5.7  5.8  5.9
##  [61]  6.0  6.1  6.2  6.3  6.4  6.5  6.6  6.7  6.8  6.9  7.0  7.1  7.2  7.3  7.4
##  [76]  7.5  7.6  7.7  7.8  7.9  8.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
##  [91]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

Plots of functions

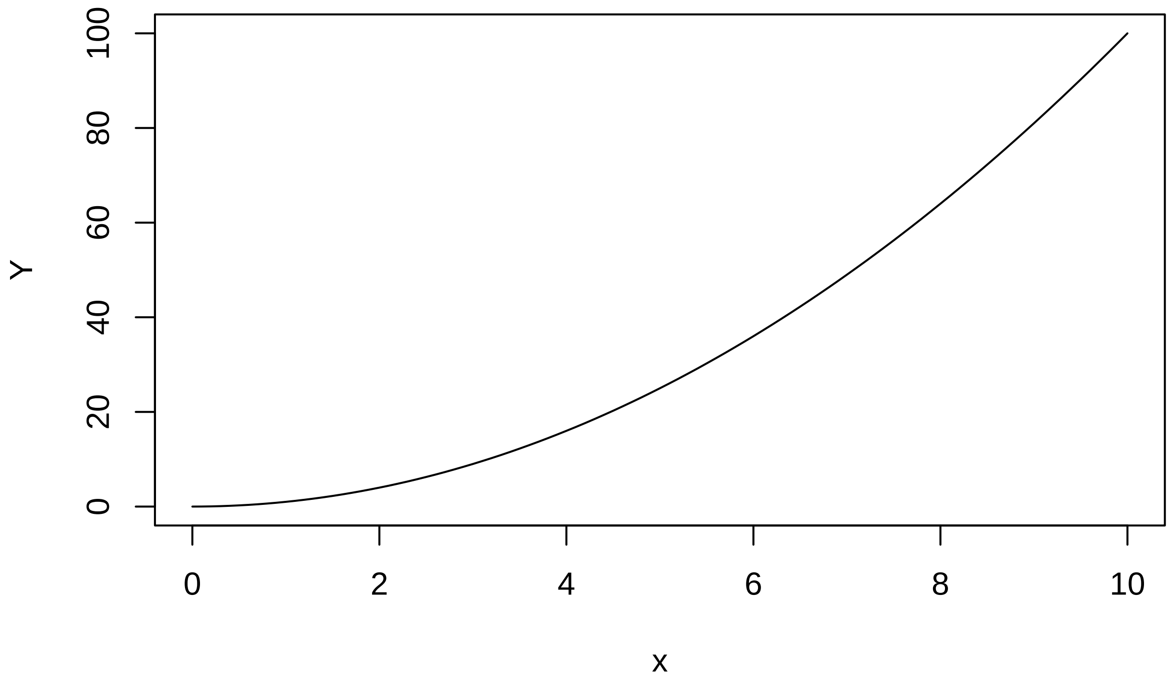
33. Create the new vector X as a copy of V. Plot X versus V as a line plot and confirm that it forms a diagonal straight line. Use `plot()`.

```
x=v
plot(x,v)
```



34. Create the new vector Y, where the elements of Y are equal to the corresponding elements of X, squared. Then plot X (x-axis) versus Y (y-axis) as a line plot (specify `type='l'`)

```
Y= x^2
plot(x, Y, type = 'l')
```

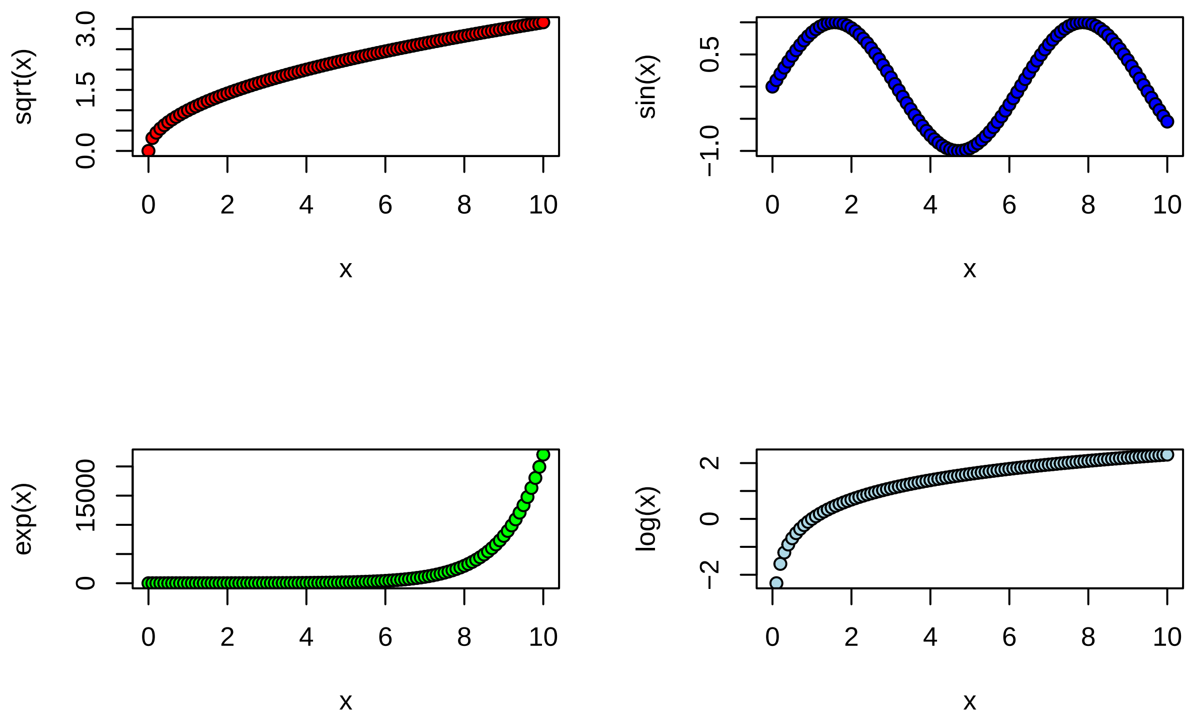


35. Make a four-panel (2x2) grid of line plots of X versus the following functions:

    a. The square root of X.
    b. The sine of X.
    c. The exponential of X.
    d. The base-10 logarithm of X. Use `par(mfrow=)`.

(Make sure to set the plot style back to 1x1 when you're done.)

```
par(mfrow=c(2,2))
plot(x, sqrt(x), pch=21,bg='red')
plot(x, sin(x), pch=21,bg='blue')
plot(x, exp(x), pch=21,bg='green')
plot(x, log(x), pch=21,bg='lightblue')
```



36. Create another new vector Z with the same length as X. The value Z takes depends on the value of the corresponding paired value of X. Set the values of Z as follows, using logical subscripts:

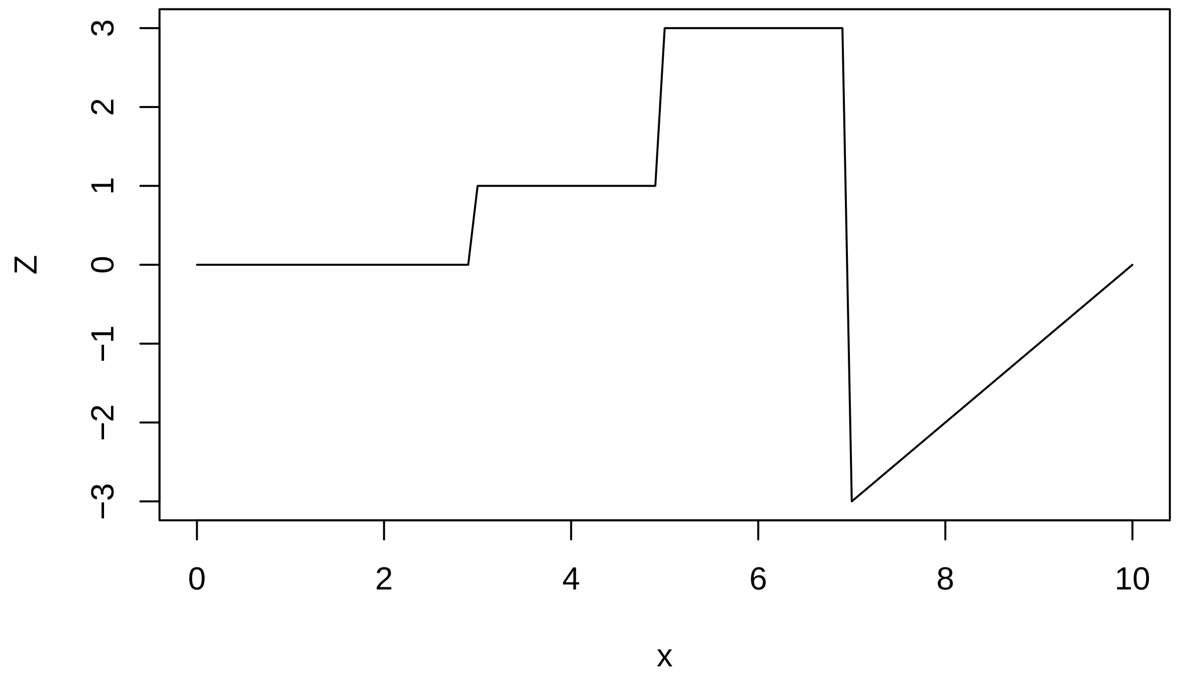
    - For values of X less than 3, set the corresponding values of Z to 0.

    - For values of X greater than or equal to 3 and less than 5, set the corresponding values of Z to 1.
    - For values of X greater than or equal to 5 and less than 7, set the corresponding values of Z to 3.
    - For values of X greater than or equal to 7, set the corresponding values of Z to 10 minus the corresponding values of X. (Be careful with this one!) Then, make a line plot of X (the vector above) versus Z.

```
Z= x
Z[Z<3]=0
Z[Z>=3 & Z<5]=1
Z[Z>=5 & Z<7]=3
Z[Z>=7]=Z[Z>=7]-10
Z
```

```
##   [1]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
##  [16]  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
##  [31]  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
##  [46]  1.0  1.0  1.0  1.0  1.0  3.0  3.0  3.0  3.0  3.0  3.0  3.0  3.0  3.0  3.0
##  [61]  3.0  3.0  3.0  3.0  3.0  3.0  3.0  3.0  3.0  3.0 -3.0 -2.9 -2.8 -2.7 -2.6
##  [76] -2.5 -2.4 -2.3 -2.2 -2.1 -2.0 -1.9 -1.8 -1.7 -1.6 -1.5 -1.4 -1.3 -1.2 -1.1
##  [91] -1.0 -0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1  0.0
```

```r
plot(x, Z, type = 'l')
```



Plotting Data

37. The following crime statistics on some common types of crime in 2015 and 2016 come from www.ons.gov.uk and document the number of offenses tabulated in England and Wales each calendar year:

Crime	2015	2016
Assault	936281	1117969
Fraud	617112	641539
Vandalism	530234	556077
Vehicular	361296	389371
Shoplifting	332891	358235

Create two vectors: one containing the names of the top 3 crime types (remember to enclose the names in quotation marks), and another one containing the 2016 numbers for those crime types. (You can enter the data by hand—next week we will learn how to read data from an external file into R as a table.)

```r
name= c("Assault", "Fraud", "Vandalism")
numbers=c(1117969,641539, 556077)
name
```

```
## [1] "Assault"   "Fraud"     "Vandalism"
```

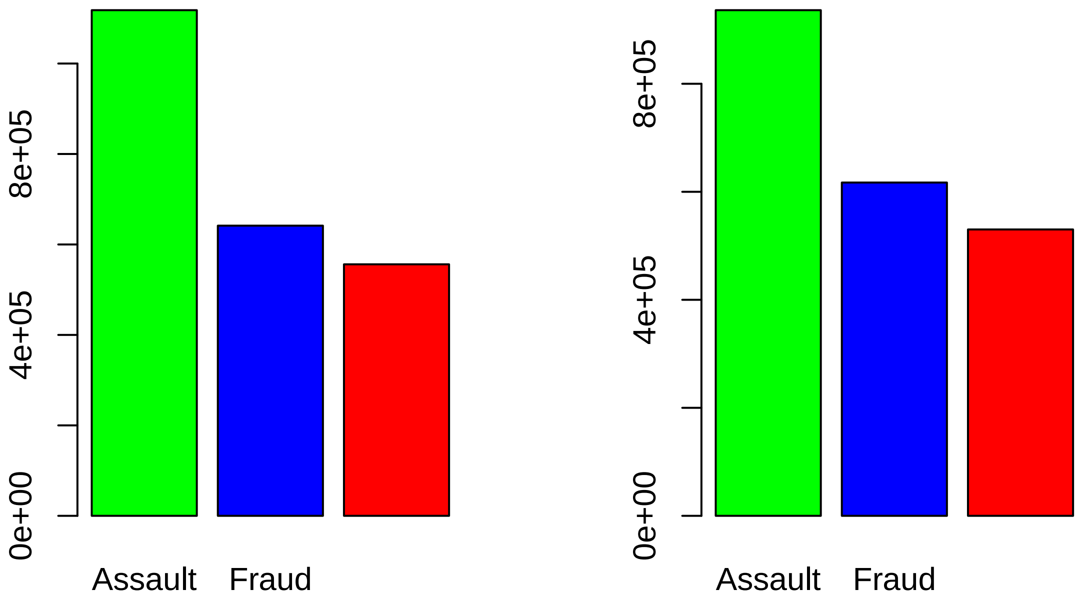
```r
numbers
```

```
## [1] 1117969  641539  556077
```

38. Plots of categorical data (independent variable) versus quantitative data (dependent variable) are often plotted as bar graphs. Make a colourful bar graph showing the numbers of assaults in 2015 and 2016. Use `barplot()`.

```r
numbers_2015= c(936281, 617112, 530234)
par(mfrow=c(1,2))
barplot(numbers,names.arg=name, col=c('green','blue', 'red'))

barplot(numbers_2015,names.arg=name, col=c('green','blue', 'red'))
```

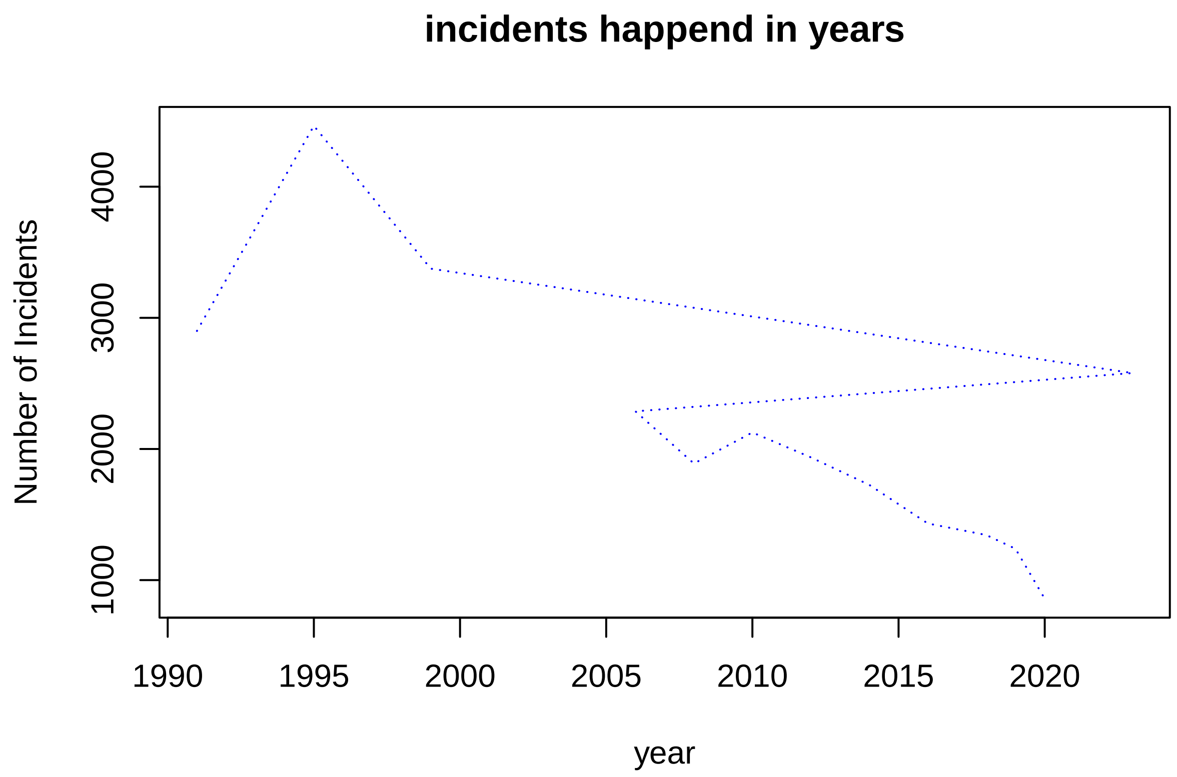


39. Consider the following data on violent crime by year from the same website:

year	incidents (thousands)
1991	2900
1995	4464
1999	3375
2003	2579
2006	2287
2008	1889
2010	2126
2012	1936
2014	1726
2016	1432
2018	1344
2019	1239
2020	858

Plots of an ordered sequential quantity such as time (independent variable) versus another continuous variable (dependent variable) are traditionally plotted as line graphs. Produce a line graph of crime rate versus time. Make sure to label both axes with appropriate units (specify `xlab=` and `ylab=`) and provide a title at the top of the plot as well (specify `main=`).

```r
year=c(1991, 1995, 1999, 2023, 2006, 2008, 2010, 2012, 2014, 2016, 2018, 2019, 2020)
inc_num= c(2900, 4464,3375, 2579, 2287, 1889, 2126, 1936, 1726, 1432, 1344, 1239, 858)
plot(year, inc_num,
     main='incidents happend in years',
     xlab='year',
     ylab = 'Number of Incidents',
     type = 'l',
     lty=3,
     lwd=1,
     col='blue')
```



40. The following data table gives heights (in inches) and weights (in lb) for a group of students:

Height	Weight	Gender
63	154	F
68	132	F
65	145	F
68	133	F
61	144	F
64	112	F
74	180	M
76	209	M
73	198	M
66	242	M
70	174	M
71	154	M
72	212	M

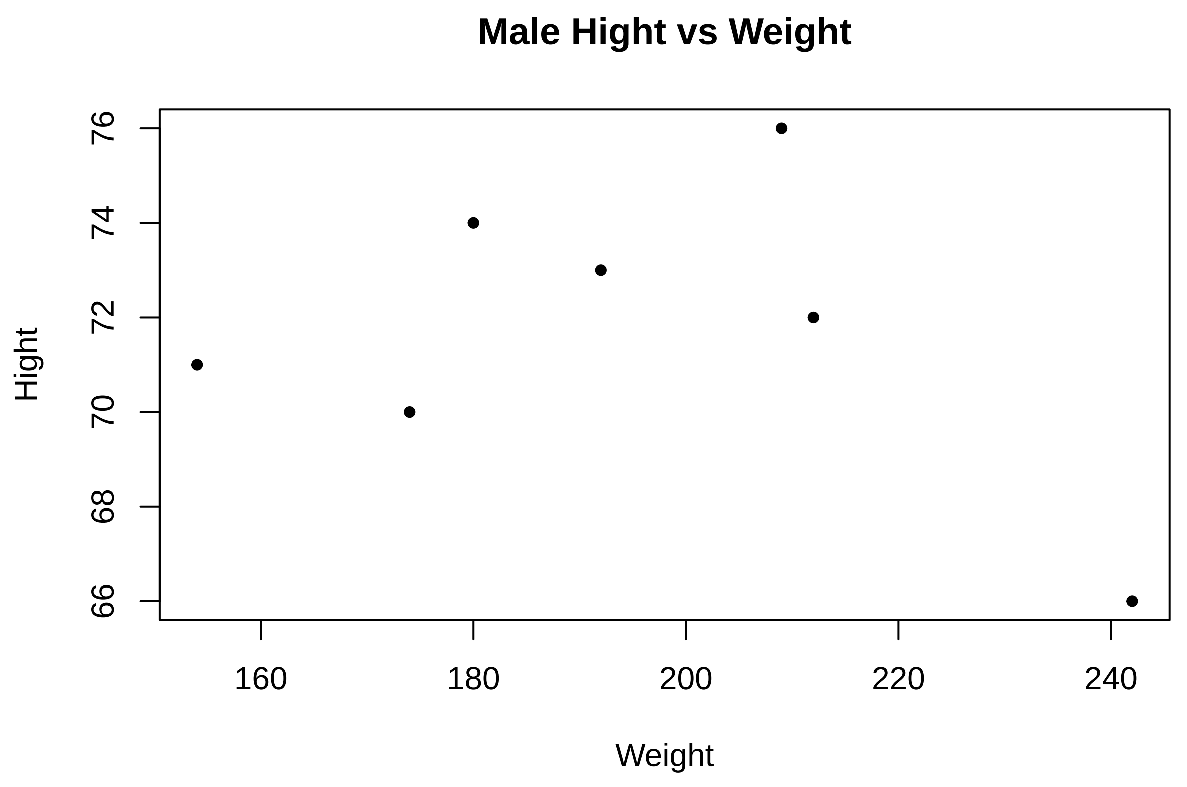
Plots of two continuous measurements where we expect there to be intrinsic scatter (one is not a simple function of the other) are called scatter plots. They are among the most common type of plot in statistics and often are the first step to insight into connections between variables.

Make a scatter plot of the data for male students. Make sure to label your axes and spruce it up by adding a colour of your choice inside the circles for each entry (specify `pch=` and `bg=`).

```
height_all=c(74, 76, 73,66, 70, 71, 72, 63,68,65, 68, 61,64)
weight_all=c(180, 209, 192, 242, 174, 154, 212,154, 132, 145,133, 144, 112)

height=c(74, 76, 73,66, 70, 71, 72)
weight=c(180, 209, 192, 242, 174, 154, 212)
plot(weight, height,
     main='Male Hight vs Weight',
     xlab='Weight',
     ylab = 'Hight',

     pch=20,
     lty=3,
     lwd=1,
     bg='blue')
```



41. Make another scatter plot showing the same data, but this time convert everything to metric by expressing the height in units of cm and the weight in units of kg. Specify the units on both axis labels.

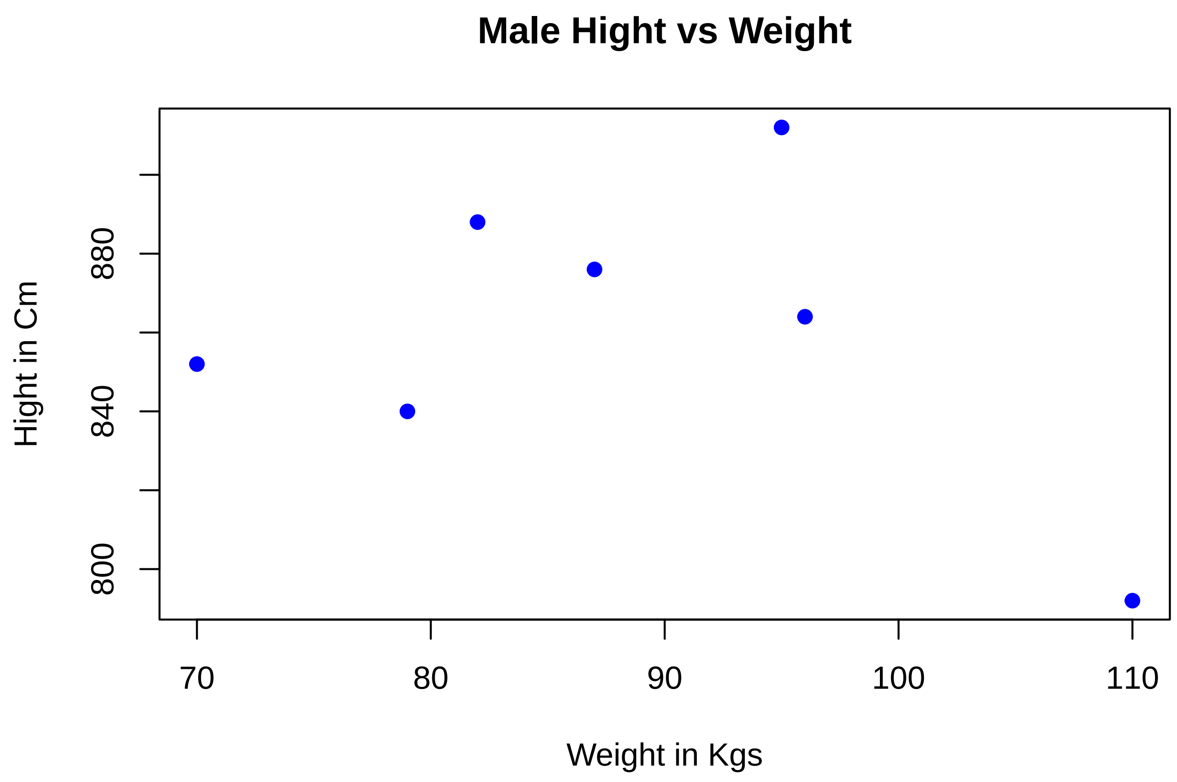
```
height_cm= height* 12
height_cm
```

```
## [1] 888 912 876 792 840 852 864
```

```r
weight_kg=round(weight*0.45359237)
weight_kg
```

```
## [1]  82  95  87 110  79  70  96
```

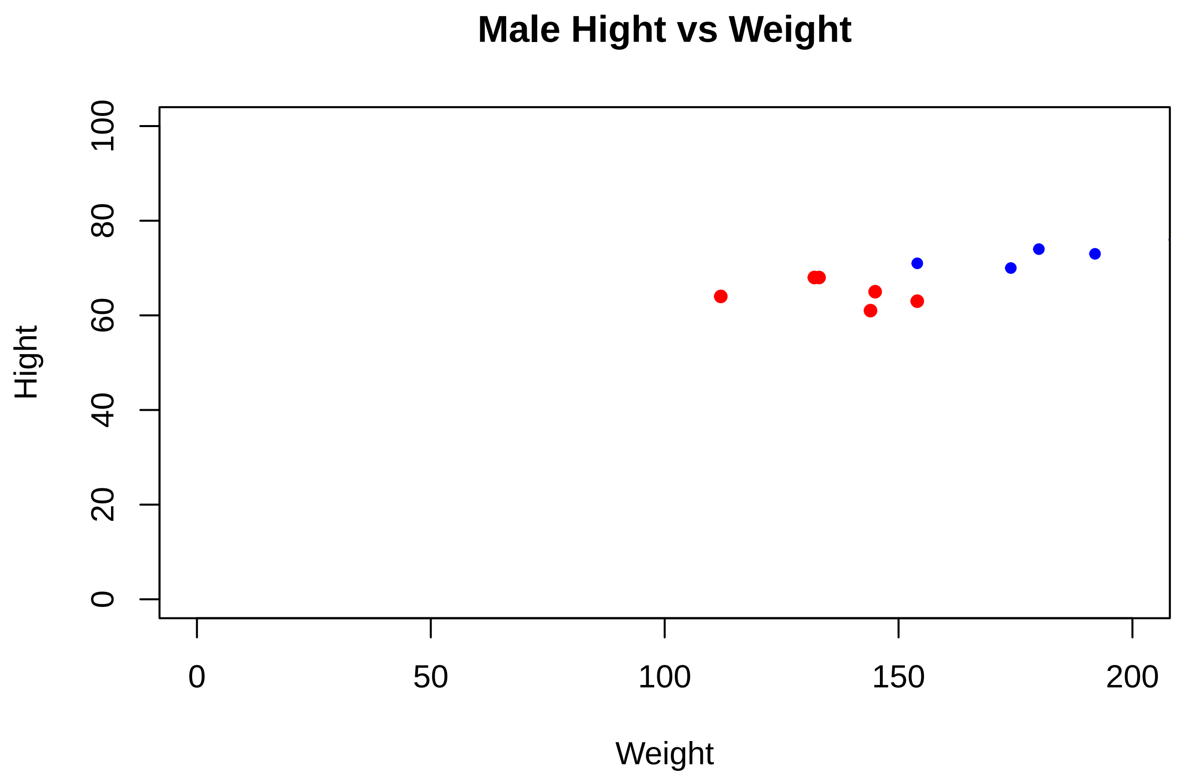
```r
plot(weight_kg, height_cm,
     main='Male Hight vs Weight',
     xlab='Weight in Kgs',
     ylab = 'Hight in Cm',
     pch=20,
     lty=2,
     lwd=3,
     col='blue')
```



42. Make another plot (using either unit system), but this time add the female students in addition to the male students. Use different coloured points for each group. Make sure the plot range shows all of the data points.

```r
height=c(74, 76, 73,66, 70, 71, 72)
weight=c(180, 209, 192, 242, 174, 154, 212)
height_F=c( 63,68,65, 68, 61,64)
weight_F=c(154, 132, 145,133, 144, 112)
plot(weight, height,
     main='Male Hight vs Weight',
     xlab='Weight',
     ylab = 'Hight',
     pch=20,
     lty=3,
     lwd=1,
     col='blue',
     xlim=c(0, 200),
     ylim=c(0, 100))

points(weight_F, height_F,
     main='Male Hight vs Weight',
     xlab='Weight',
     ylab = 'Hight',
     pch=20,
     lty=2,
     lwd=2,
     col='red')
```



More on Plotting

43. (Optional) While the most common application of R's line plot command is to present an ordered data series (such as a time series or a plot of a mathematical function), these commands just connect lines

in the order given to them, meaning that they can also be used to draw various shapes by doubling back on the X-axis values. Use this to draw the following:

a. A square, using vertices at +/-1 in each coordinate.
b. A Z-shape, using the same vertices.
c. A N-shape, using the same vertices.
d. A pentagon, inscribed in a circle with radius 1. (You don't need to plot the circle.)
e. A five-pointed star, using the same vertices. (Note - if the X and Y plotting scales are not the same the figures may be "squashed": the square may appear as a rectangle, etc. You don't need to worry about this.)

When you finish all questions, please submit your work on Canvas as a compiled PDF. If you are not able to finish within the allotted time, you may consult the solutions for help, but do not just upload the solutions as your own work. Submissions will not be marked but will be checked for completion.