Sta-518 Self Reflection

Sam_Inturi

11/16/2021

STA 518 Objectives:-

1.Import, manage, and clean data:-

I can import data from a variety of sources.

```
# Read TXT files with read.table()
children <- read.table("https://alexd106.github.io/intro2R/data/children.txt", header = TRUE)
children
```

```
##      names sex age weight height
## 1   ALFRED   M  14     69    112
## 2  BARBARA   F  13     62    102
## 3    JAMES   M  12     57     83
## 4     JANE   F  12     59     84
## 5     JOHN   M  12     59     99
## 6     JUDY   F  14     64     90
## 7   LOUISE   F  12     56     77
## 8     MARY   F  15     66    112
## 9   RONALD   M  15     67    133
## 10 WILLIAM   M  15     66    112
```

```
# Read in csv files with read_csv()
college<- read_csv("data/recent-grads.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   major = col_character(),
##   major_category = col_character()
## )
## i Use `spec()` for the full column specifications.
```

```
college
```

```
## # A tibble: 173 x 21
##     rank major_code major          major_category total sample_size   men women
##    <dbl>      <dbl> <chr>          <chr>          <dbl>       <dbl> <dbl> <dbl>
##  1     1       2419 Petroleum Engi~ Engineering     2339          36  2057   282
##  2     2       2416 Mining And Min~ Engineering      756           7   679    77
##  3     3       2415 Metallurgical ~ Engineering      856           3   725   131
##  4     4       2417 Naval Architec~ Engineering     1258          16  1123   135
##  5     5       2405 Chemical Engin~ Engineering    32260         289 21239 11021
##  6     6       2418 Nuclear Engine~ Engineering     2573          17  2200   373
```

```
##  7     7       6202 Actuarial Scie~ Business        3777          51  2110  1667
##  8     8       5001 Astronomy And ~ Physical Scie~  1792          10   832   960
##  9     9       2414 Mechanical Eng~ Engineering    91227        1029 80320 10907
## 10    10       2408 Electrical Eng~ Engineering    81527         631 65511 16016
## # ... with 163 more rows, and 13 more variables: sharewomen <dbl>,
## #   employed <dbl>, employed_fulltime <dbl>, employed_parttime <dbl>,
## #   employed_fulltime_yearround <dbl>, unemployed <dbl>,
## #   unemployment_rate <dbl>, p25th <dbl>, median <dbl>, p75th <dbl>,
## #   college_jobs <dbl>, non_college_jobs <dbl>, low_wage_jobs <dbl>
```

```
# Read in xlsx files with read_excel()
fxlsx <- read_excel("data/loyn.xlsx")
fxlsx
```

```
## # A tibble: 56 x 8
##     Site ABUND  AREA  DIST LDIST YR.ISOL GRAZE   ALT
##    <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl> <dbl>
##  1     1   5.3   0.1    39    39    1968     2   160
##  2     2   2     0.5   234   234    1920     5    60
##  3     3   1.5   0.5   104   311    1900     5   140
##  4     4  17.1   1      66    66    1966     3   160
##  5     5  13.8   1     246   246    1918     5   140
##  6     6  14.1   1     234   285    1965     3   130
##  7     7   3.8   1     467   467    1955     5    90
##  8     8   2.2   1     284  1829    1920     5    60
##  9     9   3.3   1     156   156    1965     4   130
## 10    10   3     1     311   571    1900     5   130
## # ... with 46 more rows
```

```
# Read in csv files with readr::read_csv()
banksal <- readr::read_csv("data/banksalary.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   bsal = col_double(),
##   ansal = col_double(),
##   sex = col_character(),
##   senior = col_double(),
##   age = col_double(),
##   educ = col_double(),
##   exper = col_double()
## )
```

```
banksal
```

```
## # A tibble: 93 x 7
##     bsal  ansal sex   senior   age  educ exper
##    <dbl>  <dbl> <chr>  <dbl> <dbl> <dbl> <dbl>
##  1  5040 12420  MALE      96   329    15  14
##  2  6300 12060  MALE      82   357    15  72
##  3  6000 15120  MALE      67   315    15  35.5
##  4  6000 16320. MALE      97   354    12  24
##  5  6000 12300  MALE      66   351    12  56
##  6  6840 10380  MALE      92   374    15  41.5
##  7  8100 13980. MALE      66   369    16  54.5
##  8  6000 10140  MALE      82   363    12  32
```

```
##  9  6000 12360  MALE      88   555    12 252
## 10  6900 10920  MALE      75   416    15 132
## # ... with 83 more rows
```

```
# reading a large data set
nls <- read_dta(file="https://github.com/ozanj/rclass/raw/master/data/nls72/nls72stu_percontor_vars.dta
nls
```

```
## # A tibble: 22,652 x 89
##       id schcode    bysex    srfq2a    srfq2d     fsex schlmrkr     ssex   crace
##    <dbl>   <dbl> <dbl+lb> <dbl+lbl> <dbl+lbl> <dbl+lb> <dbl+lb> <dbl+lb> <dbl+l>
##  1    18    3000 2 [2. f~ 98 [98. ~ 98 [98. ~ NA       1 [1. 1~  1 [1. ~ 4 [4. ~
##  2    67    3000 1 [1. m~ 98 [98. ~ 98 [98. ~ NA       0 [0. N~  1 [1. ~ 2 [2. ~
##  3    83    2518 2 [2. f~ 71 [71. ~ 98 [98. ~  2 [2. ~ 1 [1. 1~ 99 [99.~ 7 [7. ~
##  4   174    2911 1 [1. m~ 98 [98. ~ 71 [71. ~ NA       1 [1. 1~  1 [1. ~ 7 [7. ~
##  5   190     800 2 [2. f~ 98 [98. ~ 98 [98. ~  2 [2. ~ 1 [1. 1~  2 [2. ~ 7 [7. ~
##  6   232    7507 2 [2. f~ 98 [98. ~ 98 [98. ~  2 [2. ~ 1 [1. 1~  2 [2. ~ 7 [7. ~
##  7   315    3000 1 [1. m~ 72 [72. ~ 71 [71. ~ NA       0 [0. N~  1 [1. ~ 7 [7. ~
##  8   380    9516 1 [1. m~ 98 [98. ~ 98 [98. ~ NA       1 [1. 1~  1 [1. ~ 7 [7. ~
##  9   414    2518 1 [1. m~ 98 [98. ~ 98 [98. ~ NA       0 [0. N~ 98 [98.~ 7 [7. ~
## 10   430    2701 1 [1. m~ 98 [98. ~ 98 [98. ~ NA       1 [1. 1~  1 [1. ~ 4 [4. ~
## # ... with 22,642 more rows, and 80 more variables: csex <dbl+lbl>,
## #   tfusex <dbl+lbl>, ft67 <dbl+lbl>, ftfusex <dbl+lbl>, fi29 <dbl+lbl>,
## #   fi39 <dbl+lbl>, fi43 <dbl+lbl>, fi52a <dbl+lbl>, fi52b <dbl+lbl>,
## #   fi52c <dbl+lbl>, fi52d <dbl+lbl>, fi52e <dbl+lbl>, fi52f <dbl+lbl>,
## #   race86 <dbl+lbl>, birthmon <dbl>, birthday <dbl>, birthyr <dbl>,
## #   srfq0 <dbl>, srfq2b <dbl>, srfq2c <dbl>, srfq2e <dbl>, srfq2f <dbl>,
## #   srfq2g <dbl>, srfq2h <dbl>, srfq2i <dbl>, scvocsc <dbl>, scpic1 <dbl>,
## #   scpic2 <dbl>, scpict <dbl>, scrdsc <dbl>, sclgsc <dbl>, scmatsc <dbl>,
## #   scmscm1 <dbl>, scmscm2 <dbl>, scmscm3 <dbl>, scmscmt <dbl>, srifprk <dbl>,
## #   fbirthmo <dbl>, fbirthda <dbl>, fbirthyr <dbl>, cfaocp <dbl>,
## #   cbirthm <dbl>, cbirthd <dbl>, cbirthyr <dbl>, fi40cm <dbl>, fi40cy <dbl>,
## #   fi41cm <dbl>, fi41cy <dbl>, fi42cm <dbl>, fi42by <dbl>, fi111a84 <dbl>,
## #   fi111b84 <dbl>, fi111c84 <dbl>, fi111d84 <dbl>, fi111e84 <dbl>,
## #   fi111f84 <dbl>, fi111g84 <dbl>, fi111h84 <dbl>, fi111i84 <dbl>,
## #   fi111j84 <dbl>, fi111k84 <dbl>, fi111l84 <dbl>, fi111m84 <dbl>,
## #   fi111n84 <dbl>, fi111o84 <dbl>, fi111a85 <dbl>, fi111b85 <dbl>,
## #   fi111c85 <dbl>, fi111d85 <dbl>, fi111e85 <dbl>, fi111f85 <dbl>,
## #   fi111g85 <dbl>, fi111h85 <dbl>, fi111i85 <dbl>, fi111j85 <dbl>,
## #   fi111k85 <dbl>, fi111l85 <dbl>, fi111m85 <dbl>, fi111n85 <dbl>,
## #   fi111o85 <dbl>
```

```
nc <- read_csv("data/nc.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   fage = col_double(),
##   mage = col_double(),
##   mature = col_character(),
##   weeks = col_double(),
##   premie = col_character(),
##   visits = col_double(),
##   marital = col_character(),
##   gained = col_double(),
##   weight = col_double(),
```

```
##   lowbirthweight = col_character(),
##   gender = col_character(),
##   habit = col_character(),
##   whitemom = col_character()
## )
```

```
country <- read_excel("data/country.xlsx")
people <- read_excel("data/people.xlsx")
```

I can isolate information from a larger data source.

```
# I am taking the above children table to perform data isolation operations
# filter operation

filter(children, sex == "M" & age == "15")
```

```
##     names sex age weight height
## 1  RONALD   M  15     67    133
## 2 WILLIAM   M  15     66    112
```

```
# I am taking the above banksal table to perform data isolation operations
fbanksal <- filter(banksal, bsal >= 6000 & senior >= 60)
fbanksal
```

```
## # A tibble: 26 x 7
##     bsal  ansal sex    senior   age  educ exper
##    <dbl>  <dbl> <chr>  <dbl> <dbl> <dbl> <dbl>
##  1  6300 12060  MALE      82   357    15  72
##  2  6000 15120  MALE      67   315    15  35.5
##  3  6000 16320. MALE      97   354    12  24
##  4  6000 12300  MALE      66   351    12  56
##  5  6840 10380  MALE      92   374    15  41.5
##  6  8100 13980. MALE      66   369    16  54.5
##  7  6000 10140  MALE      82   363    12  32
##  8  6000 12360  MALE      88   555    12 252
##  9  6900 10920  MALE      75   416    15 132
## 10  6900 10920  MALE      89   481    12 175
## # ... with 16 more rows
```

```
# Randomly select rows from banksal table
slice_sample(banksal, n = 5, replace = TRUE)
```

```
## # A tibble: 5 x 7
##    bsal  ansal sex    senior   age  educ exper
##   <dbl>  <dbl> <chr>   <dbl> <dbl> <dbl> <dbl>
## 1  5040 12420  MALE       96   329    15  14
## 2  6300  9780  FEMALE     66   394    12  86.5
## 3  8100 13980. MALE       66   369    16  54.5
## 4  6300 10860. FEMALE     84   662    15 231
## 5  5400 12600  MALE       78   305    12   7
```

```
# using select statement
select(iris, petal_length = Petal.Length)
```

```
##     petal_length
## 1            1.4
## 2            1.4
## 3            1.3
```

```
## 4           1.5
## 5           1.4
## 6           1.7
## 7           1.4
## 8           1.5
## 9           1.4
## 10          1.5
## 11          1.5
## 12          1.6
## 13          1.4
## 14          1.1
## 15          1.2
## 16          1.5
## 17          1.3
## 18          1.4
## 19          1.7
## 20          1.5
## 21          1.7
## 22          1.5
## 23          1.0
## 24          1.7
## 25          1.9
## 26          1.6
## 27          1.6
## 28          1.5
## 29          1.4
## 30          1.6
## 31          1.6
## 32          1.5
## 33          1.5
## 34          1.4
## 35          1.5
## 36          1.2
## 37          1.3
## 38          1.4
## 39          1.3
## 40          1.5
## 41          1.3
## 42          1.3
## 43          1.3
## 44          1.6
## 45          1.9
## 46          1.4
## 47          1.6
## 48          1.4
## 49          1.5
## 50          1.4
## 51          4.7
## 52          4.5
## 53          4.9
## 54          4.0
## 55          4.6
## 56          4.5
## 57          4.7
```

```
## 58          3.3
## 59          4.6
## 60          3.9
## 61          3.5
## 62          4.2
## 63          4.0
## 64          4.7
## 65          3.6
## 66          4.4
## 67          4.5
## 68          4.1
## 69          4.5
## 70          3.9
## 71          4.8
## 72          4.0
## 73          4.9
## 74          4.7
## 75          4.3
## 76          4.4
## 77          4.8
## 78          5.0
## 79          4.5
## 80          3.5
## 81          3.8
## 82          3.7
## 83          3.9
## 84          5.1
## 85          4.5
## 86          4.5
## 87          4.7
## 88          4.4
## 89          4.1
## 90          4.0
## 91          4.4
## 92          4.6
## 93          4.0
## 94          3.3
## 95          4.2
## 96          4.2
## 97          4.2
## 98          4.3
## 99          3.0
## 100         4.1
## 101         6.0
## 102         5.1
## 103         5.9
## 104         5.6
## 105         5.8
## 106         6.6
## 107         4.5
## 108         6.3
## 109         5.8
## 110         6.1
## 111         5.1
```

```
## 112          5.3
## 113          5.5
## 114          5.0
## 115          5.1
## 116          5.3
## 117          5.5
## 118          6.7
## 119          6.9
## 120          5.0
## 121          5.7
## 122          4.9
## 123          6.7
## 124          4.9
## 125          5.7
## 126          6.0
## 127          4.8
## 128          4.9
## 129          5.6
## 130          5.8
## 131          6.1
## 132          6.4
## 133          5.6
## 134          5.1
## 135          5.6
## 136          6.1
## 137          5.6
## 138          5.5
## 139          4.8
## 140          5.4
## 141          5.6
## 142          5.1
## 143          5.1
## 144          5.9
## 145          5.7
## 146          5.2
## 147          5.0
## 148          5.2
## 149          5.4
## 150          5.1
```

I can combine information from multiple data sources

Example data tables

```
orders <- read.csv("https://raw.githubusercontent.com/ds4stats/r-tutorials/master/merging/data/orders.cs
orders
```

```
##   order id   date
## 1     1  4 Jan-01
## 2     2  8 Feb-01
## 3     3 42 Apr-15
## 4     4 50 Apr-17
```

```
customers <- read.csv("https://raw.githubusercontent.com/ds4stats/r-tutorials/master/merging/data/custom
customers
```

```
##   id    name
```

```
## 1  4   Tukey
## 2  8 Wickham
## 3 15   Mason
## 4 16  Jordan
## 5 23   Patil
## 6 42     Cox
```

Joining data tables

```
# Inner_join creates a new table which is restricted to cases where the values of "by variable" exist i
inner_join(x = orders, y = customers, by = "id")
```

```
##   order id   date    name
## 1     1  4 Jan-01   Tukey
## 2     2  8 Feb-01 Wickham
## 3     3 42 Apr-15     Cox
```

```
# Left_join returns all cases from the x data table, regardless of whether there are matching values of
left_join(x = orders, y = customers, by = "id")
```

```
##   order id   date    name
## 1     1  4 Jan-01   Tukey
## 2     2  8 Feb-01 Wickham
## 3     3 42 Apr-15     Cox
## 4     4 50 Apr-17    <NA>
```

```
# Right_join returns all cases from the y data table, regardless of whether there are matching values o
right_join(x = orders, y = customers, by = "id")
```

```
##   order id   date    name
## 1     1  4 Jan-01   Tukey
## 2     2  8 Feb-01 Wickham
## 3     3 42 Apr-15     Cox
## 4    NA 15   <NA>   Mason
## 5    NA 16   <NA>  Jordan
## 6    NA 23   <NA>   Patil
```

```
# Full_join returns all rows and columns from both x and y.
full_join(x = orders, y = customers, by = "id")
```

```
##   order id   date    name
## 1     1  4 Jan-01   Tukey
## 2     2  8 Feb-01 Wickham
## 3     3 42 Apr-15     Cox
## 4     4 50 Apr-17    <NA>
## 5    NA 15   <NA>   Mason
## 6    NA 16   <NA>  Jordan
## 7    NA 23   <NA>   Patil
```

```
# Semi_join returns all rows from the x data table where there are matching values of the by variable(s
semi_join(x = orders, y = customers, by = "id")
```

```
##   order id   date
## 1     1  4 Jan-01
## 2     2  8 Feb-01
## 3     3 42 Apr-15
```

I can restructure information to be in a “tidy” format.

```r
# restructure a dataset to be in a more efficient format and add features to make a table more understa
wide_measures <- country %>%
  filter(year == 2002) %>%
  group_by(continent) %>%
  summarise(
    med_LE = median(pop),
    mean_LE = mean(pop)
  )

wide_measures %>%
  pivot_longer(
    cols = ends_with("_LE"),
    names_to = "measure",
    values_to = "values"
  )
```

```
## # A tibble: 10 x 3
##    continent measure     values
##    <chr>     <chr>        <dbl>
##  1 Africa    med_LE    8821778.
##  2 Africa    mean_LE  16033152.
##  3 Americas  med_LE    8650322
##  4 Americas  mean_LE  33990910.
##  5 Asia      med_LE   22662365
##  6 Asia      mean_LE 109145521.
##  7 Europe    med_LE    9518744
##  8 Europe    mean_LE  19274129.
##  9 Oceania   med_LE   11727414.
## 10 Oceania   mean_LE  11727414.
```

```r
long_measures <- wide_measures %>%
  pivot_longer(
    cols = ends_with("_LE"),
    names_to = "measure",
    values_to = "values"
  )

long_measures %>%
  pivot_wider(
    names_from = continent,
    values_from = values
  )
```

```
## # A tibble: 2 x 6
##   measure    Africa  Americas       Asia    Europe   Oceania
##   <chr>       <dbl>     <dbl>      <dbl>     <dbl>     <dbl>
## 1 med_LE   8821778.  8650322   22662365   9518744  11727414.
## 2 mean_LE 16033152. 33990910. 109145521. 19274129. 11727414.
```

I can transform information to be in a format better suited for specific tasks.

```r
# reducing the dataset for better fit
reducing_dataset <- people %>%
  select(skin_color) %>%
  separate(skin_color,
           into =  c("skin_first", "skin_second")) %>%
```

```
  pivot_longer(cols = everything(),
               names_to = "skin_order",
               values_to = "skin_color",
               values_drop_na = TRUE) %>%
  select(skin_color)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 4 rows [16, 47, 67,
## 76].
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 71 rows [1, 2, 4,
## 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 21, 22, 23, ...].
```

```
reducing_dataset
```

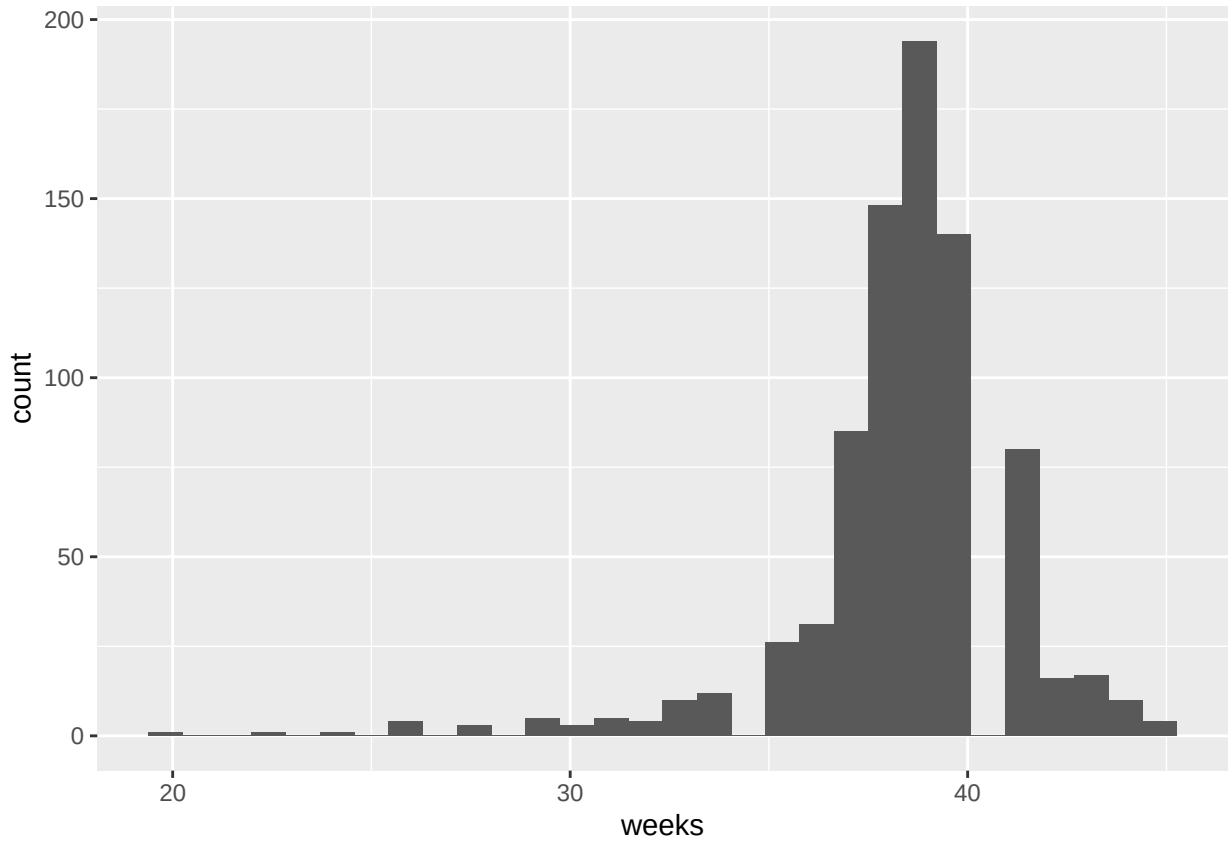
```
## # A tibble: 103 x 1
##    skin_color
##    <chr>
##  1 fair
##  2 gold
##  3 white
##  4 blue
##  5 white
##  6 light
##  7 light
##  8 light
##  9 white
## 10 red
## # ... with 93 more rows
```

Create graphical displays and numerical summaries of data for exploratory analysis and presentations.

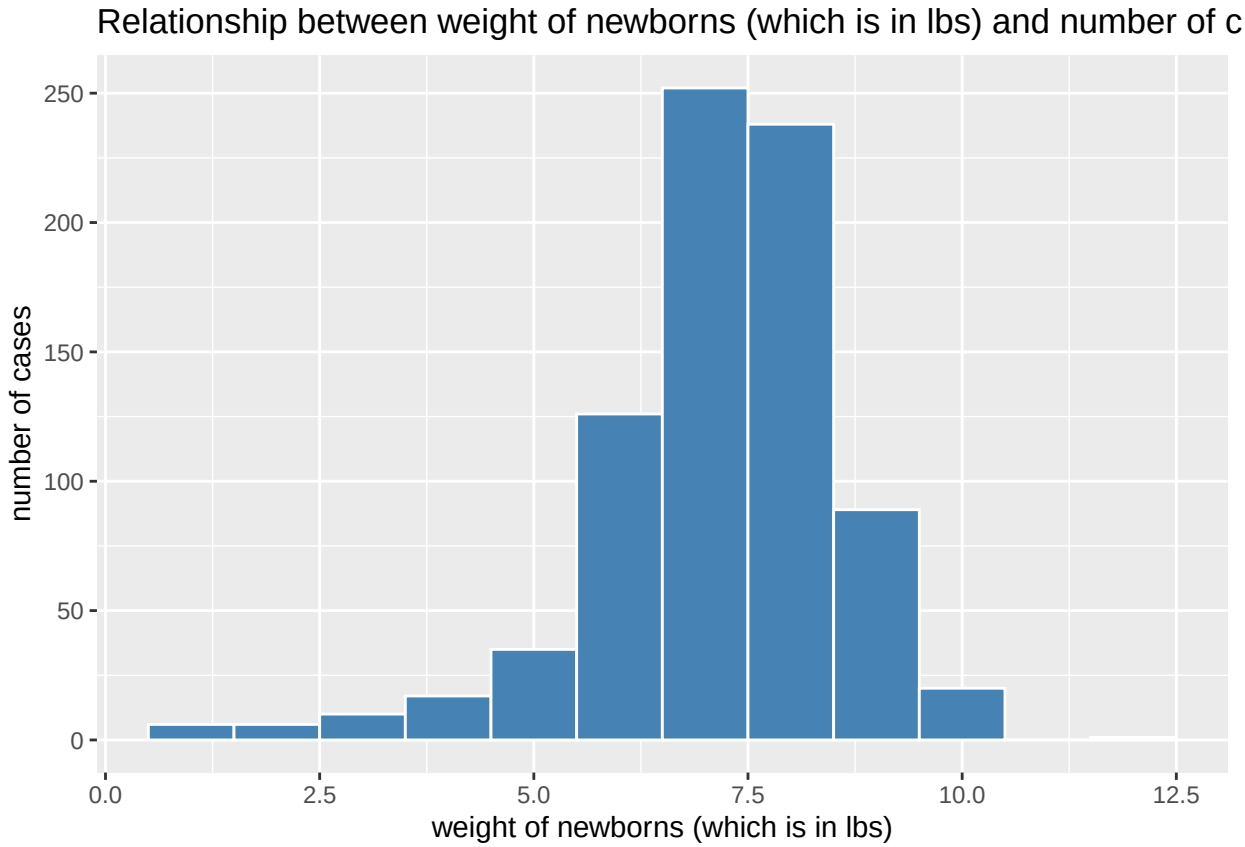
I can create graphical displays of data that highlight key features. I can combine multiple graphical displays or numerical summaries into an effective data product.

```
# histogram
ggplot(data = nc, aes(x = weeks))+
  geom_histogram()
```

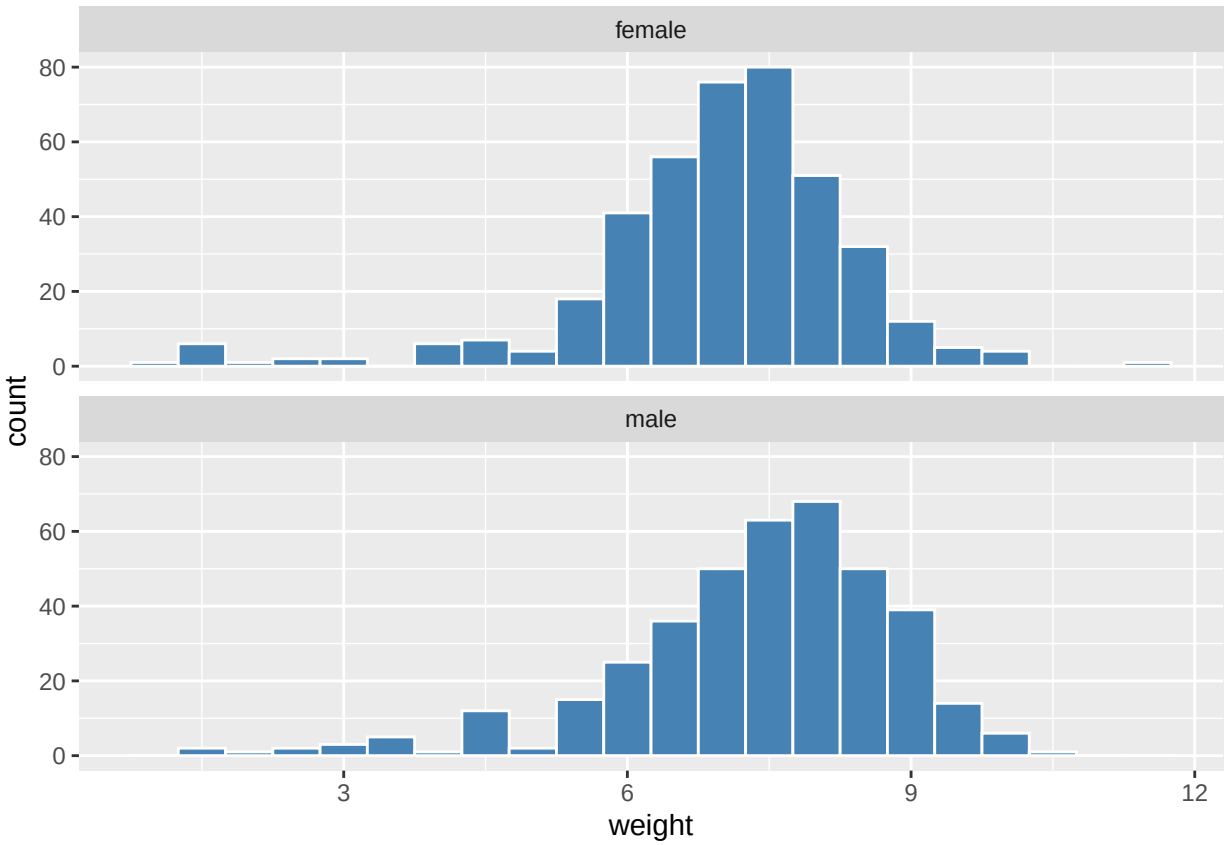
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



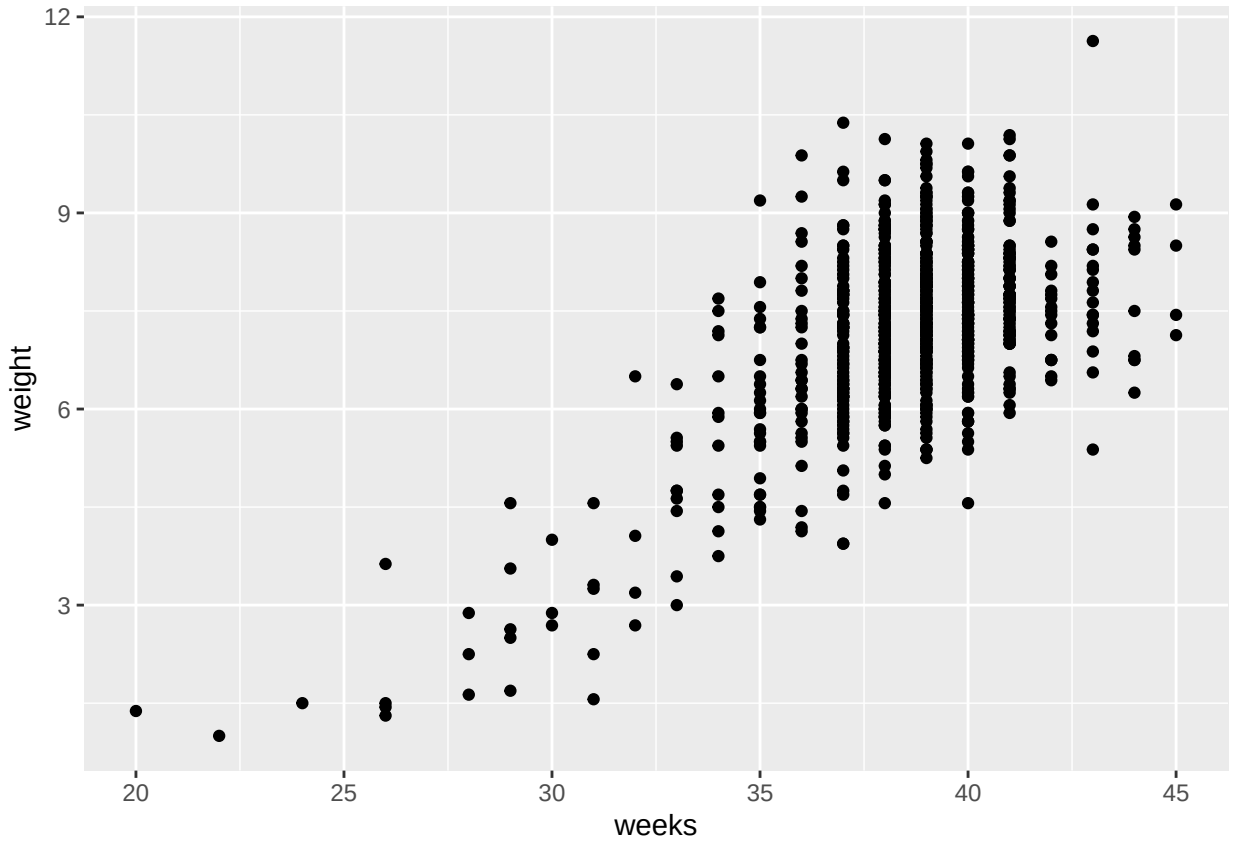

```
# collered and labeled histogram
ggplot(data = nc, aes(x = weight))+
  geom_histogram(binwidth = 1, color = "white", fill = "steelblue")+
  labs(x = "weight of newborns (which is in lbs)", y = "number of cases",
       title = "Relationship between weight of newborns (which is in lbs) and number of cases")
```



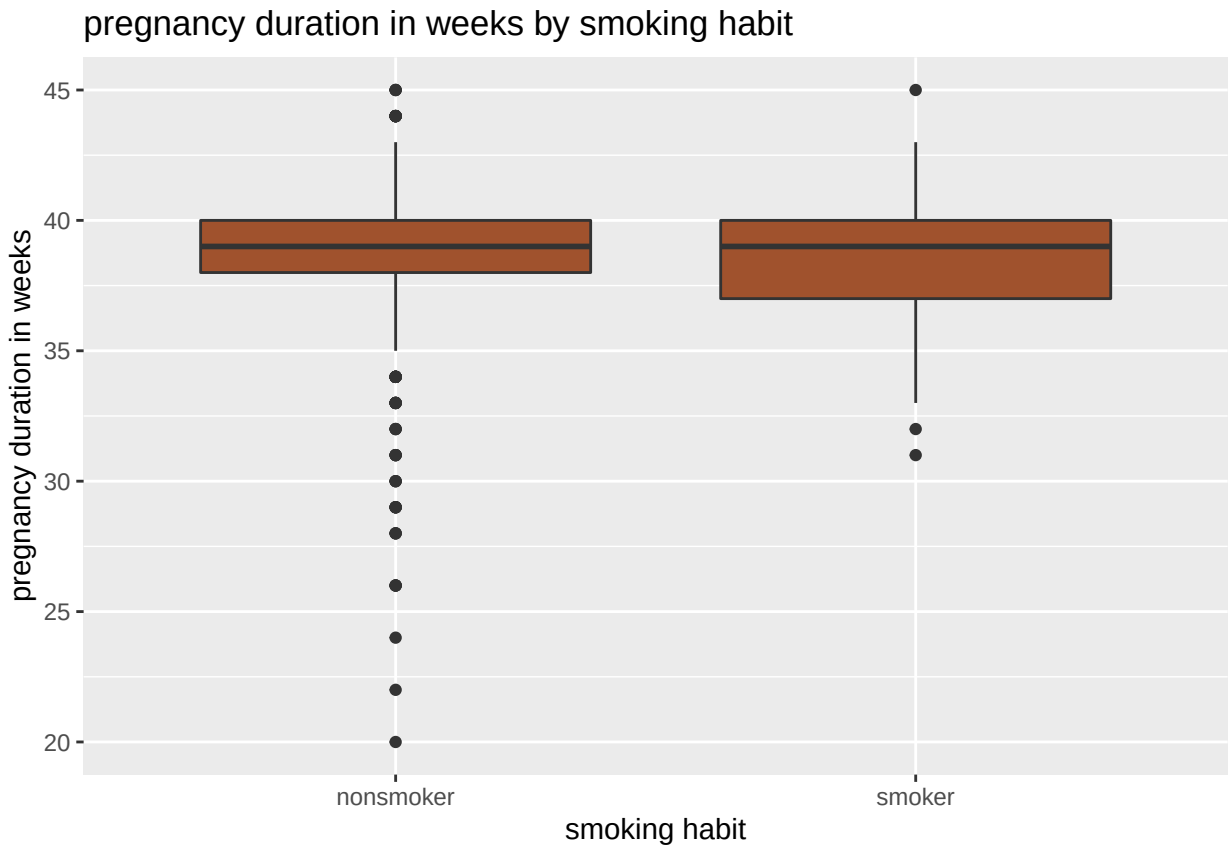

```
# two histo grams in one plane
ggplot(data = nc, aes(x = weight)) +
  geom_histogram(binwidth = 0.5, color = "white", fill = "steelblue") +
  facet_wrap(~ gender, ncol = 1)
```



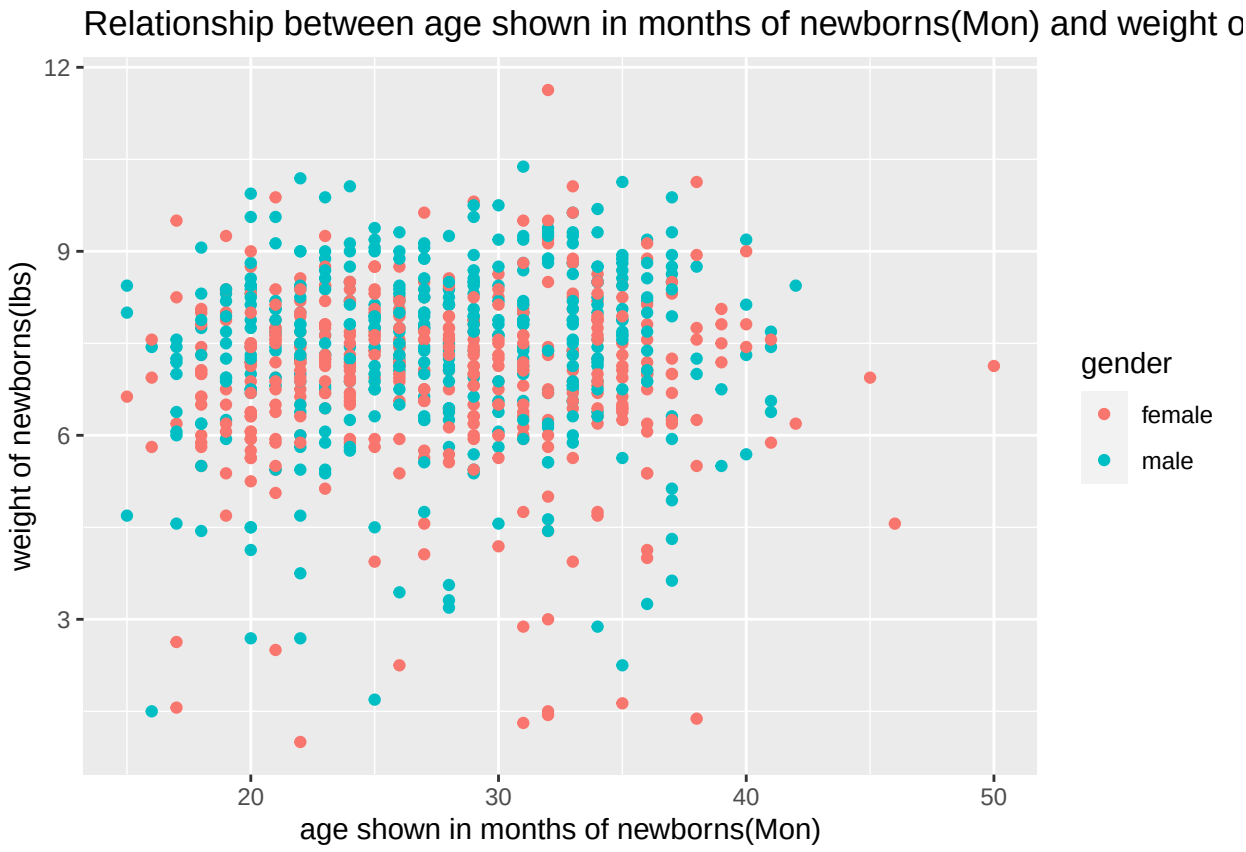

```
# scatter plot
ggplot(data = nc, aes(x = weeks, y = weight)) +
  geom_point()
```



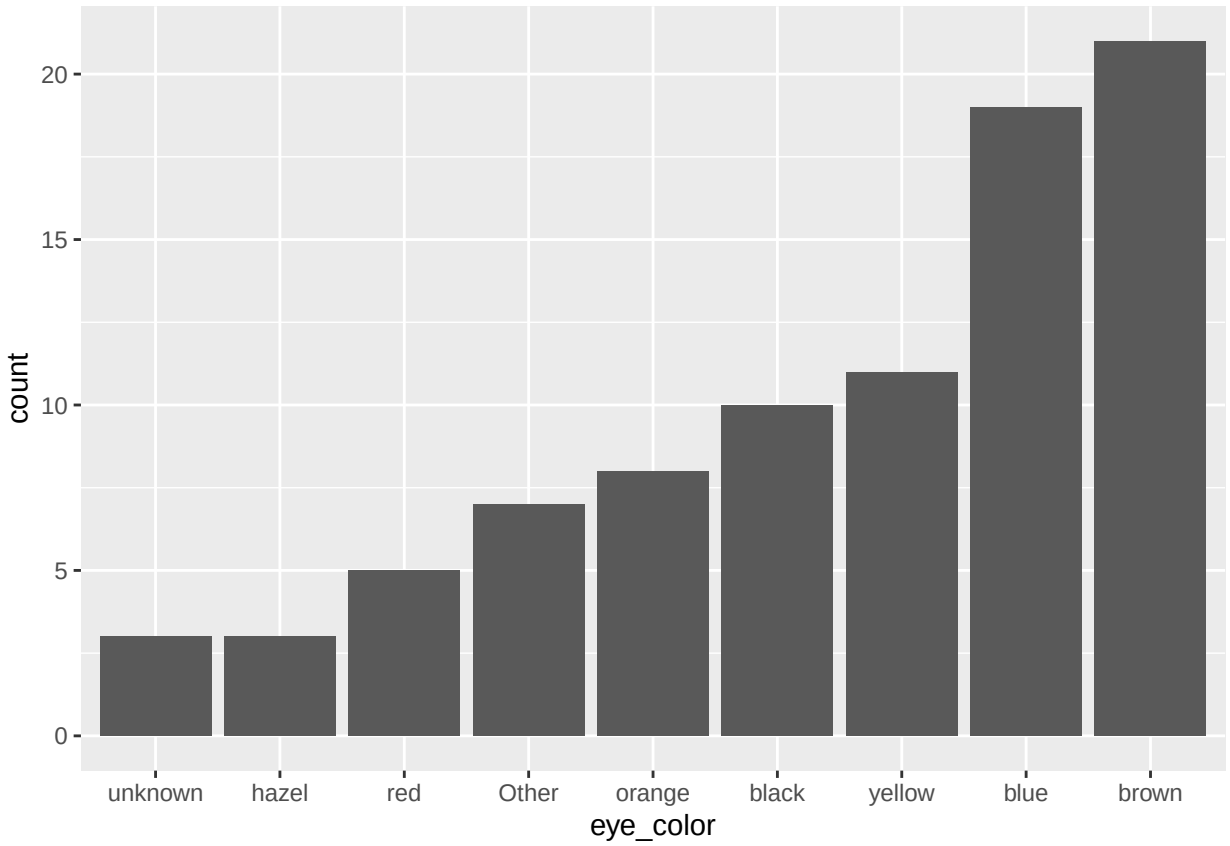

```
# box plot
ggplot(data = nc, aes(x = habit, y = weeks)) +
  geom_boxplot(fill = "sienna")+
  labs(x = "smoking habit", y = "pregnancy duration in weeks",
       title = "pregnancy duration in weeks by smoking habit")
```




```
# adding color and labels to the plot
ggplot(data = nc, aes(x = mage, y = weight, color = gender))+
  geom_point() +
  labs(x = "age shown in months of newborns(Mon)", y = "weight of newborns(lbs)",
       title = "Relationship between age shown in months of newborns(Mon) and weight of newborns(lbs)")
```




```
# bar plot of the variable eye_color
people %>%
  mutate(eye_color = fct_lump(eye_color, prop = 0.03),
         eye_color = fct_infreq(eye_color),
         eye_color = fct_rev(eye_color)) %>%
  ggplot(mapping = aes(x = eye_color)) +
  geom_bar()
```



Creating my own functions

```
generate_n_samples <- function(n){
  tibble(
    normal_distribution = rnorm(n = n, mean = 50, sd = 10),
    exponential_distribution = rexp(n = n, rate = 0.2),
    binomial_distribution = rbinom(n = n, size = 100, prob = 0.25),
    uniform_distribution = runif(n = n, min = 10, max = 20)
  )
}

generate_n_samples(50)
```

```
## # A tibble: 50 x 4
##    normal_distributi~ exponential_distrib~ binomial_distribu~ uniform_distribut~
##                 <dbl>                <dbl>              <int>              <dbl>
##  1               31.1                9.25                 22               12.3
##  2               40.4                0.296                33               16.1
##  3               30.9                5.86                 23               13.0
##  4               45.9                0.157                21               16.9
##  5               48.3                7.76                 25               10.9
##  6               48.1                3.13                 26               11.2
##  7               51.2                0.615                26               17.0
##  8               34.6                6.37                 30               16.9
##  9               47.8               19.6                  24               10.9
## 10               25.4                6.78                 21               13.9
## # ... with 40 more rows
```

```r
pow <- function(x, y) {
# function to print x raised to the power y
result <- x^y
print(paste(x,"raised to the power", y, "is", result))
}

pow(2,10)
```

```
## [1] "2 raised to the power 10 is 1024"
```

```r
pow(5,2)
```

```
## [1] "5 raised to the power 2 is 25"
```

```r
# this function will return whether a given number is positive, negative or zero
check <- function(x) {
if (x > 0) {
result <- "Positive"
}
else if (x < 0) {
result <- "Negative"
}
else {
result <- "Zero"
}
return(result)
}

check(1)
```

```
## [1] "Positive"
```

```r
check(-10)
```

```
## [1] "Negative"
```

```r
check(0)
```

```
## [1] "Zero"
```

```r
g <- factor(c("a","b","a","b","a","b","a","b","a","b","a","b"))
v <- c(1,4,1,4,1,4,2,8,2,8,2,8)
d <- data.frame(g,v)
d$cs <- ave(v, g, FUN=cumsum)
d
```

```
##    g v cs
## 1  a 1  1
## 2  b 4  4
## 3  a 1  2
## 4  b 4  8
## 5  a 1  3
## 6  b 4 12
## 7  a 2  5
## 8  b 8 20
## 9  a 2  7
## 10 b 8 28
## 11 a 2  9
```

```
## 12 b 8 36
```

Iterations

```
# using The repeat() Statement
Newton <- function(n, j=2, x=1) {
   # Use Newton's method to find the positive, real jth root of n,
   # where the default is to find the square root of n or j = 2.
   # x is a seed value to start the search.

   old.x <- x
   repeat {
      # Update x
      new.x <- old.x - ((old.x^j - n)) / (j * old.x^(j - 1))
      # Compute relative error as a 2-norm.
      conv <- sum((new.x - old.x)^2 / old.x^2)
      # Exit test with return() statement
      if(conv < 1e-10) return(old.x)
      # Save interation result
      old.x <- new.x
   }
}
Newton(500, 2, 4)
```

```
## [1] 22.36068
```

```
# using The while() Statement
bit.string <- function(n) {
tmp.string <- numeric(50)
i <- 0
while(n > 0) {
   tmp.string[50 - i] <- n %% 2    # modula
   n <- n %/% 2                    # integer divide
   i <- i + 1
}

first.one <- match(1, tmp.string)
return(tmp.string[first.one:50])
}

# Test the function
bit.string(1)
```

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

```
bit.string(2)
```

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

```
bit.string(3)
```

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

```
bit.string(4)
```

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

```
bit.string(5)
```

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

```
# using The for() Statement
#creating a matrix using for()
x <- matrix(0, nrow = 3, ncol = 4)
for(i in 1:3)  {
   for(j in 1:4)  {
      x[i,j] <- i+j
   }
}
x
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    3    4    5
## [2,]    3    4    5    6
## [3,]    4    5    6    7
```

```
# Use for() to create vector of cumulative sums
silly.csum <- function(N) {
   s <- vector("numeric", N)
   for (i in 1:N){
      if(i == 1) s[i] <- 1
   else s[i] <- s[i-1] + i;
   }
return(s)
}

# Test the function
silly.csum(10)
```

```
##  [1]  1  3  6 10 15 21 28 36 45 55
```

```
# Speed test1
system.time(silly.csum(1e7))
```

```
##    user  system elapsed
##   0.886   0.052   0.941
```

```
# speed test2
system.time(cumsum(as.numeric(1:1e7)))
```

```
##    user  system elapsed
##   0.064   0.085   0.149
```