talk03 练习与作业

目录

0.1 练习和作业说明

将相关代码填写入以 "'{r} "' 标志的代码框中，运行并看到正确的结果；

完成后，用工具栏里的"Knit" 按键生成 PDF 文档；

将生成的 PDF 改为：姓名`-`学号`-talk03` 作业`.pdf`，并提交到老师指定的平台/钉群。

0.2 talk03 内容回顾

- 二维表：`data.frame`, `tibble`
  - 声明
  - 操作

0.3 练习与作业：用户验证

请运行以下命令，验证你的用户名。

如你当前用户名不能体现你的真实姓名，请改为拼音后再运行本作业!

```
Sys.info()[["user"]]
```

```
## [1] "lucas"
```

```
Sys.getenv("HOME")
```

```
## [1] "/Users/lucas"
```

0.4 练习与作业 1，`data.frame`

注：以下内容来自 https://www.r-exercises.com/。

- 生成下面的 data.frame 的前三列，之后再增加 Sex 这列

	Age	Height	Weight	Sex
Alex	25	177	57	F
Lilly	31	163	69	F
Mark	23	190	83	M
Oliver	52	179	75	M
Martha	76	163	70	F
Lucas	49	183	83	M
Caroline	26	164	53	F

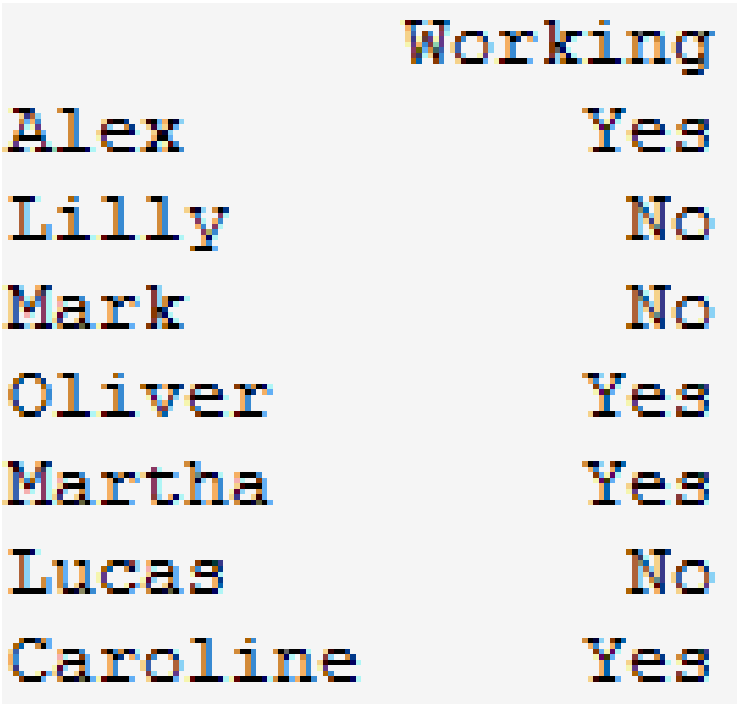
```r
# Generate the first 3 columns;
df01 = data.frame(
  Age = c(25, 31, 23, 52 ,76, 49, 26),
  Height = c(177, 163, 190, 179, 163, 183, 164),
  `Weight` = c(57, 69, 83, 75, 70, 83, 53)
)

# Rename
rownames(df01) [1:7] = c("Alex", "Lilly", "Mark", "Oliver", "Martha", "Lucas", "Carolin

# Insert the forth row
cbind(df01, "Sex" = c("F", "F", "M", "M", "F", "M", "F"))
```

```
##          Age Height Weight Sex
## Alex      25    177     57   F
## Lilly     31    163     69   F
## Mark      23    190     83   M
## Oliver    52    179     75   M
## Martha    76    163     70   F
## Lucas     49    183     83   M
## Caroline  26    164     53   F
```

```
# Print the final result
df01
```

```
##           Age Height Weight
## Alex       25    177     57
## Lilly      31    163     69
## Mark       23    190     83
## Oliver     52    179     75
## Martha     76    163     70
## Lucas      49    183     83
## Caroline   26    164     53
```

- 生成以下 data.frame，确保 Working 这列的类型是 character，而不是 factor



	Working
Alex	Yes
Lilly	No
Mark	No
Oliver	Yes
Martha	Yes
Lucas	No
Caroline	Yes

```r
# Generate data.frame
df02 = data.frame(
  "Working" = c("Yes", "No", "No", "Yes", "Yes", "No", "Yes")
)

# Rename
rownames(df02) [1:7] = c("Alex", "Lilly", "Mark", "Oliver", "Martha", "Lucas", "Carolin

# Print the final result
df02;
```

```
##          Working
## Alex         Yes
## Lilly         No
## Mark          No
## Oliver       Yes
## Martha       Yes
## Lucas         No
## Caroline     Yes
```

```r
# Print the statue of row "Working"
class(df02$Working);
```

```
## [1] "character"
```

```r
is.character(df02$Working)
```

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

- 检查系统自带变量 `state.center` 的内容，将其转化为 `data.frame`

```
## 代码写这里，并运行;

# Inspect the content
state.center
```

```
## $x
##  [1]  -86.7509 -127.2500 -111.6250  -92.2992 -119.7730 -105.5130  -72.3573
##  [8]  -74.9841  -81.6850  -83.3736 -126.2500 -113.9300  -89.3776  -86.0808
## [15]  -93.3714  -98.1156  -84.7674  -92.2724  -68.9801  -76.6459  -71.5800
## [22]  -84.6870  -94.6043  -89.8065  -92.5137 -109.3200  -99.5898 -116.8510
## [29]  -71.3924  -74.2336 -105.9420  -75.1449  -78.4686 -100.0990  -82.5963
## [36]  -97.1239 -120.0680  -77.4500  -71.1244  -80.5056  -99.7238  -86.4560
## [43]  -98.7857 -111.3300  -72.5450  -78.2005 -119.7460  -80.6665  -89.9941
## [50] -107.2560
##
## $y
##  [1] 32.5901 49.2500 34.2192 34.7336 36.5341 38.6777 41.5928 38.6777 27.8744
## [10] 32.3329 31.7500 43.5648 40.0495 40.0495 41.9358 38.4204 37.3915 30.6181
## [19] 45.6226 39.2778 42.3645 43.1361 46.3943 32.6758 38.3347 46.8230 41.3356
## [28] 39.1063 43.3934 39.9637 34.4764 43.1361 35.4195 47.2517 40.2210 35.5053
## [37] 43.9078 40.9069 41.5928 33.6190 44.3365 35.6767 31.3897 39.1063 44.2508
## [46] 37.5630 47.4231 38.4204 44.5937 43.0504
```

```
# Convert it into data.frame
df03 = data.frame(state.center)

# Print the data frame
df03
```

```
##            x       y
## 1   -86.7509 32.5901
## 2  -127.2500 49.2500
## 3  -111.6250 34.2192
```

```
## 4   -92.2992 34.7336
## 5  -119.7730 36.5341
## 6  -105.5130 38.6777
## 7   -72.3573 41.5928
## 8   -74.9841 38.6777
## 9   -81.6850 27.8744
## 10  -83.3736 32.3329
## 11 -126.2500 31.7500
## 12 -113.9300 43.5648
## 13  -89.3776 40.0495
## 14  -86.0808 40.0495
## 15  -93.3714 41.9358
## 16  -98.1156 38.4204
## 17  -84.7674 37.3915
## 18  -92.2724 30.6181
## 19  -68.9801 45.6226
## 20  -76.6459 39.2778
## 21  -71.5800 42.3645
## 22  -84.6870 43.1361
## 23  -94.6043 46.3943
## 24  -89.8065 32.6758
## 25  -92.5137 38.3347
## 26 -109.3200 46.8230
## 27  -99.5898 41.3356
## 28 -116.8510 39.1063
## 29  -71.3924 43.3934
## 30  -74.2336 39.9637
## 31 -105.9420 34.4764
## 32  -75.1449 43.1361
## 33  -78.4686 35.4195
## 34 -100.0990 47.2517
## 35  -82.5963 40.2210
## 36  -97.1239 35.5053
```

```
## 37 -120.0680 43.9078
## 38  -77.4500 40.9069
## 39  -71.1244 41.5928
## 40  -80.5056 33.6190
## 41  -99.7238 44.3365
## 42  -86.4560 35.6767
## 43  -98.7857 31.3897
## 44 -111.3300 39.1063
## 45  -72.5450 44.2508
## 46  -78.2005 37.5630
## 47 -119.7460 47.4231
## 48  -80.6665 38.4204
## 49  -89.9941 44.5937
## 50 -107.2560 43.0504
```

- **生成一个 50 行 * 5 列的 matrix，将其行名改为：row_i 格式，其中 i 为当前的行号，比如 row_1, row_2 等**

```r
## 代码写这里，并运行；

# Generate a 50*5 matrix
m01 = matrix(sample(1:250), nrow = 50, ncol = 5);

#Rename
row_name_prefix = "row_"
col_name_prefix = "col_"
rownames(m01)[1:50] = paste0(row_name_prefix, 1:50)
colnames(m01)[1:5] = paste0(col_name_prefix, 1:5)
m01
```

```
##       col_1 col_2 col_3 col_4 col_5
## row_1   111    46     1     4   180
```

```
## row_2     15    19   229    57   236
## row_3    135   189    32   188   123
## row_4    247   208   202   214    51
## row_5    160   227   242    66   174
## row_6    113   220   222   118   149
## row_7    152   194   184   122   145
## row_8    112   172   163   212    44
## row_9    245    60   159   116   136
## row_10    85   226    81   185   134
## row_11   130    75   213    33   146
## row_12   198     2   165    83    41
## row_13   201   132   181    27    90
## row_14   187   124    79    42   197
## row_15   129   106   233    43   196
## row_16   186    26   200   175   217
## row_17    86   235    62   209   109
## row_18    58   221    64    63    34
## row_19    23   176   139   183   104
## row_20     5    53   151    87     6
## row_21   131    76    25   121   140
## row_22   238   228   115   230   148
## row_23    78    95    22    59    50
## row_24   241   157   225    31    35
## row_25    97   105   142   127   248
## row_26    54    99    45   168   147
## row_27   117   125   144    67   243
## row_28   179   120   102   164   206
## row_29   199   210   153    39   138
## row_30    21   173    11   205   101
## row_31    80    68   250    55    14
## row_32    92   246    17    28    82
## row_33   154    77   215   150    74
## row_34    47   126     8    52    71
```

```
## row_35     9    73   191   162    49
## row_36     7   100    72   137   141
## row_37   166   128    84   234   193
## row_38   178    98   249    65    89
## row_39    13   237    88    91   223
## row_40   119   204   177   231   143
## row_41   182   207   114   232    16
## row_42   211    70    40    30   103
## row_43    69     3    29    96   190
## row_44   218   239   171   107    56
## row_45   156   161   110   155   158
## row_46   240    48    93    20   195
## row_47   108   133   244    18    94
## row_48   219    61   224   170    36
## row_49    10   167    38   203   192
## row_50    12    37   216    24   169
```

- **使用系统自带变量 `VADeaths`，做如下练习：**
- 检查 `VADeaths` 的类型，如果不是 `data.frame`，则转换之；
- 添加新的一列，取名 `Total`，其值为每行的总合
- 调整列的顺序，将 `Total` 变为第一列。

```
## 代码写这里，并运行；

VADeaths
```

```
##       Rural Male Rural Female Urban Male Urban Female
## 50-54       11.7          8.7       15.4          8.4
## 55-59       18.1         11.7       24.3         13.6
## 60-64       26.9         20.3       37.0         19.3
```

```
## 65-69       41.0          30.9       54.6          35.1
## 70-74       66.0          54.3       71.1          50.0
```

```r
# Inspect the class of VADeaths
if (!is.data.frame(VADeaths)) {
  df04 = as.data.frame(VADeaths)
}

# Add the row "Total", which presents the sum of each column
df04$Total = rowSums(df04)

# Adjust the order to put "Total" to the first column
df04 = df04[, c("Total", names(df04)[1:(ncol(df04)-1)])]

# Print the result
df04
```

```
##       Total Rural Male Rural Female Urban Male Urban Female
## 50-54  44.2       11.7          8.7       15.4          8.4
## 55-59  67.7       18.1         11.7       24.3         13.6
## 60-64 103.5       26.9         20.3       37.0         19.3
## 65-69 161.6       41.0         30.9       54.6         35.1
## 70-74 241.4       66.0         54.3       71.1         50.0
```

- **用系统自带的 `swiss` 数据做练习：**
- 取子集，选取第 1, 2, 3, 10, 11, 12 and 13 行，第 `Examination`, `Education` 和 `Infant.Mortality` 列；
- 将 `Sarine` 行 `Infant.Mortality` 列的值改为 NA；
- 增加一列，命名为 `Mean`，其值为当前行的平均值；

```
## 代码写这里，并运行；

# Load the data
data(swiss)

# Choose the subset
df05_subset = swiss[c(1, 2, 3, 10, 11, 12, 13), c("Examination", "Education", "Infant.M

# Replace the data of "Sarine"'s "Infant.Mortality to NA
df05_subset[df05_subset$Infant.Mortality == "Sarine", "Infant.Mortality"] = NA

# Add a row called "Mean"
df05_subset = rowMeans(df05_subset, na.rm = TRUE)

# Print the data frame
df05_subset
```

```
##   Courtelary     Delemont Franches-Mnt       Sarine      Veveyse        Aigle
##     16.40000     12.40000     10.06667     17.80000     14.83333     16.50000
##      Aubonne
##     13.36667
```

- **将下面三个变量合并生成一个 data.frame**

```
Id <- LETTERS

x <- seq(1,43,along.with=Id)

y <- seq(-20,0,along.with=Id)
```

```
## 代码写这里，并运行；

# Create the variables
```

```r
Id = LETTERS
x = seq(1, 43, along.with = Id)
y = seq(-20, 0, along.with = Id)

# Combine to data.frame
df06 <- data.frame(Id = Id, x = x, y = y)

# Show the result
df06
```

```
##    Id     x     y
## 1   A  1.00 -20.0
## 2   B  2.68 -19.2
## 3   C  4.36 -18.4
## 4   D  6.04 -17.6
## 5   E  7.72 -16.8
## 6   F  9.40 -16.0
## 7   G 11.08 -15.2
## 8   H 12.76 -14.4
## 9   I 14.44 -13.6
## 10  J 16.12 -12.8
## 11  K 17.80 -12.0
## 12  L 19.48 -11.2
## 13  M 21.16 -10.4
## 14  N 22.84  -9.6
## 15  O 24.52  -8.8
## 16  P 26.20  -8.0
## 17  Q 27.88  -7.2
## 18  R 29.56  -6.4
## 19  S 31.24  -5.6
## 20  T 32.92  -4.8
## 21  U 34.60  -4.0
## 22  V 36.28  -3.2
```

```
## 23  W 37.96  -2.4
## 24  X 39.64  -1.6
## 25  Y 41.32  -0.8
## 26  Z 43.00   0.0
```

问： seq 函数中的 along.with 参数的意义是什么？请举例说明。

答：

seq() 函数中的 **along.with** 参数是一个用于指定生成序列的长度和步长的参数。它是一个可选参数，通常与 **from** 和 **to** 参数一起使用，用于确保生成的序列具有与 **along.with** 参数相同的长度。

当提供 **along.with** 参数时，R 会根据 **along.with** 参数中的向量的长度来确定生成序列的长度，并根据需要对 **from** 和 **to** 参数进行适当的调整，以确保生成的序列具有与 **along.with** 相同的长度。

Partly based on wikipedia.org

```
## 代码写这里，并运行；

# Create an along.with vector
along_with_vector = c("A", "B", "C", "D", "E")

# Generate a sequence using the along.with parameter
sequence = seq(1, 10, along.with = along_with_vector)

# Display the generated sequence and the along.with vector
along_with_vector
```

```
## [1] "A" "B" "C" "D" "E"
```

```
sequence
```

```
## [1]  1.00  3.25  5.50  7.75 10.00
```

- **提供代码，合并以下两个 data.frame**

```
> df1 的内容
Id Age
1 14
2 12
3 15
4 10

>df2 的内容
Id Sex Code
1 F a
2 M b
3 M c
4 F d
```

合并之后的结果：

```
> M
Id Age Sex Code
1 14 F a
2 12 M b
3 15 M c
4 10 F d
```

```r
## 代码写这里，并运行；

# Create df1
df1 = data.frame(Id = 1:4, Age = c(14, 12, 15, 10))

# Create df2
df2 = data.frame(Id = 1:4, Sex = c("F", "M", "M", "F"), Code = c("a", "b", "c", "d"))
```

```r
# Merge the two data frames
result = merge(df1, df2, by = "Id")

# Rename the column names of the result
colnames(result) = c("Id", "Age", "Sex", "Code")

# Print the merged result
result
```

```
##   Id Age Sex Code
## 1  1  14   F    a
## 2  2  12   M    b
## 3  3  15   M    c
## 4  4  10   F    d
```

- **从上面的 `data.frame` 中删除 `code` 列**
- Method 1: Using the `subset()` function

```r
## 代码写这里，并运行；

# Use the subset() function to remove the Code column
result = subset(result, select = -Code)

# Print the data frame after removing the Code column
result
```

```
##   Id Age Sex
## 1  1  14   F
## 2  2  12   M
## 3  3  15   M
## 4  4  10   F
```

- Method 2: Directly specifying NULL

```
## 代码写这里，并运行;

# Set the Code column to NULL directly
result$Code = NULL

# Print the data frame after removing the Code column
result
```

```
##   Id Age Sex
## 1  1  14   F
## 2  2  12   M
## 3  3  15   M
## 4  4  10   F
```

- **练习，回答代码中的问题**

```
## 1. 生成一个10 行2 列的data.frame
df3 <- data.frame( data = 1:10, group = c("A","B") );
## 2. 增加一列，其长度是1，可以吗?
cbind(df3, newcol = 1);
## 3. 增加一列，其长度是10，可以吗?
cbind(df3, newcol = 1:10);
## 4. 增加一列，其长度是2，可以吗?
cbind(df3, newcol = 1:2);
## 5. 增加一列，其长度是3，可以吗?
cbind(df3, newcol = 1:3);
```

答:

All of the question is OK, except for the last one. Here is the reason:

When using the `cbind()` function to add a new column to a data frame, the length of the new column must match the number of rows in the data frame; otherwise, it will result in an error.

Partly based on the R documentation.

0.5 练习与作业 2，tibble

- **运行以下代码，生成一个新的 `tibble`:**

```
## 如果系统中没有 lubridate 包，则安装:
if (!require("lubridate")){
  chooseCRANmirror();
  install.packages("lubridate");
}
```

```
## Loading required package: lubridate
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(lubridate);

if (!require("tibble")){
  chooseCRANmirror();
  install.packages("tibble");
}
```

```
## Loading required package: tibble
```

```r
library(tibble);

tibble(
  a = lubridate::now() + runif(1e3) * 86400,
  b = lubridate::today() + runif(1e3) * 30,
  c = 1:1e3,
  d = runif(1e3),
  e = sample(letters, 1e3, replace = TRUE)
)
```

```
## # A tibble: 1,000 x 5
##    a                   b              c     d e
##    <dttm>              <date>     <int> <dbl> <chr>
##  1 2023-09-18 17:54:40 2023-10-02     1 0.383 y
##  2 2023-09-19 02:41:42 2023-09-28     2 0.926 y
##  3 2023-09-19 16:19:42 2023-10-13     3 0.154 o
##  4 2023-09-19 10:12:40 2023-10-14     4 0.988 b
##  5 2023-09-19 10:56:46 2023-10-06     5 0.805 n
##  6 2023-09-19 03:49:41 2023-09-25     6 0.619 v
##  7 2023-09-18 19:40:56 2023-09-22     7 0.528 x
##  8 2023-09-18 17:46:21 2023-09-30     8 0.221 j
##  9 2023-09-19 15:36:07 2023-09-18     9 0.313 z
## 10 2023-09-19 13:34:35 2023-10-03    10 0.814 f
## # i 990 more rows
```

从中可以看出，tibble 支持一些细分数据类型，包括：

- <dttm>
- <date>

等；

- **生成一个如下的 tibble，完成以下任务：**

```
df <- tibble(
  x = runif(5),
  y = rnorm(5)
)
```

任务：

- 取一列，比如 x 这一列，得到一个 tibble；
- 取一列，比如 y 这一列，得到一个 vector；

```
## 代码写这里，并运行；
# Generate a new tibble
df = tibble(
  x = runif(5),
  y = rnorm(5)
)
df
```

```
## # A tibble: 5 x 2
##       x      y
##   <dbl>  <dbl>
## 1 0.612  0.818
## 2 0.980 -0.776
## 3 0.875  0.866
## 4 0.585 -0.602
## 5 0.405 -0.204
```

```
# Extract "x" row
df_x = df[,1]
df_ext_x = tibble(df_x)
df_ext_x
```

```
## # A tibble: 5 x 1
##       x
##   <dbl>
## 1 0.612
## 2 0.980
## 3 0.875
## 4 0.585
## 5 0.405
```

```
# Extract "y" row
df_y = df[,2]
df_ext_y = tibble(df_y)
df_ext_y
```

```
## # A tibble: 5 x 1
##        y
##    <dbl>
## 1  0.818
## 2 -0.776
## 3  0.866
## 4 -0.602
## 5 -0.204
```

- **用 tibble 函数创建一个新的空表，并逐行增加一些随机的数据，共增加三行：**

```
## 代码写这里，并运行；
## 新 tibble, with defined columns ... 创建表头
tb = tibble( name = character(), age = integer(), salary = double() );

## 增加三行随机数据；
# Generate the data and add the pre-generated data to the table
```

```
for(i in 1:3) {
  new_row = tibble(name = sample(c("Alice", "Bob", "Charlie"), 1),
                   age = sample(20:60, 1),
                   salary = runif(1, 30000, 80000))
  tb <- add_row(tb, new_row)
}

# Print the tibble
tb
```

```
## # A tibble: 3 x 3
##   name      age salary
##   <chr>   <int>  <dbl>
## 1 Bob        51 71898.
## 2 Alice      21 68873.
## 3 Charlie    43 46388.
```

> 下面的题目本来也想和前面那道题一样用英文回答的，但是感觉说的并不是很清楚，于是就用中文了

- ** 请解释为什么下面第一行代码能够运行成功，但第二个不行？ **

这个可以：

```
data.frame(a = 1:6, b = LETTERS[1:2]);
```

但下面这个不行：

```
tibble(a = 1:6, b = LETTERS[1:2]);
```

问：为什么？tibble 循环的规则是什么？

答：

这是因为 `data.frame` 和 `tibble` 对于列的处理方式不同。

1. **data.frame**: `data.frame` 允许列的长度不同，如果列的长度不同，它会自动进行适配。因此，可以在 `data.frame` 中创建不同长度的列，如下：

```
data.frame(a = 1:6, b = LETTERS[1:2])
```

2. **tibble**: 与 `data.frame` 不同，`tibble` 要求所有的列长度必须相同。因此，如果尝试在 `tibble` 中创建不同长度的列，就像第二个示例一样，引发错误。

```
Error: Tibble columns must have consistent lengths, only values of length one are r
* Length 6: Column `a`
* Length 2: Column `b`
```

- **attach 和 detach:**

问：这个两个函数的用途是什么？请用 `iris` 这个系统自带变量举例说明。

答：

`attach` 和 `detach` 是在 R 中用于将数据框或其他数据对象附加到搜索路径（search path）的函数。它们的主要目的是可以更方便地访问数据框中的变量。

`attach` 函数的作用是将一个数据框或数据对象添加到搜索路径中，这样就可以直接使用数据框中的变量名而不需要使用数据框的名称来访问这些变量。

`detach` 函数的作用是从搜索路径中移除附加的数据框或数据对象。这可以帮助避免变量名的歧义，并将搜索路径恢复到较原始的状态。

下面是示例：

```
# Using attach to add "iris" data frame into the search path
attach(iris)
```

```r
# I can now call on the data in data.frame "iris" directly, not need to add prefix "iri
head(Sepal.Length)  # Call on "Sepal.Length"
```

```
## [1] 5.1 4.9 4.7 4.6 5.0 5.4
```

```r
head(Species)       # Call on "Species"
```

```
## [1] setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica
```

```r
# Using detach to remove "iris" data frame from the cearch path
detach(iris)

# Re-accessing the variable in the "iris" data box now results in an error because it i
# head(Sepal.Length)  # This line of code will cause an error
# head(Species)       # This line of code will cause an error
```

需要注意的是，虽然 `attach` 和 `detach` 可以使代码看起来更简洁，但它们容易引发变量名的歧义，尤其当多个数据框中有相同名称的变量时。为了编写更清晰和健壮的代码，建议使用 `$` 运算符或 `with()` 函数来显式访问数据框中的变量，而不是使用 `attach` 和 `detach`。

Partly based on R Documentation and the "DigitalOcean" forum.

- **使用内置变量 `airquality`:**
- 检查它是否是 `tibble`;
- 如果不是，转化为 `tibble`;

```
## 代码写这里，并运行；

# Check if airquality is a tibble
if (!is_tibble(airquality)) {
  # If it's not a tibble, convert it to a tibble
  airquality = as_tibble(airquality)
}

# Now, airquality should be a tibble
airquality
```

```
## # A tibble: 153 x 6
##    Ozone Solar.R  Wind  Temp Month   Day
##    <int>   <int> <dbl> <int> <int> <int>
##  1    41     190   7.4    67     5     1
##  2    36     118   8      72     5     2
##  3    12     149  12.6    74     5     3
##  4    18     313  11.5    62     5     4
##  5    NA      NA  14.3    56     5     5
##  6    28      NA  14.9    66     5     6
##  7    23     299   8.6    65     5     7
##  8    19      99  13.8    59     5     8
##  9     8      19  20.1    61     5     9
## 10    NA     194   8.6    69     5    10
## # i 143 more rows
```

- **问：tibble::enframe 函数的用途是什么？请举例说明：**

答：

`tibble::enframe()` 函数用于将向量或列表转换为 `tibble` 格式的数据框，其中一列包含原始数据，另一列包含索引或名称。以下是一个示例：

```r
# Create a vector
test_vector = c("apple", "banana", "cherry")

# Converting vectors to tibble using enframe
test_tibble_converted = tibble::enframe(test_vector, name = "fruit")

# Print the result
print(test_tibble_converted)
```

```
## # A tibble: 3 x 2
##   fruit value
##   <int> <chr>
## 1     1 apple
## 2     2 banana
## 3     3 cherry
```

Partly based on R Documentation and the "DigitalOcean" forum.

- **简述 tibble 相比 data.frame 的优势？并用实例展示**

答：

1. **可读性：** tibble 对于数据的输出和显示更加友好，它会自动缩短过长的列名以提高可读性。这对于处理大型数据集时特别有用。
2. **数据结构一致性：** tibble 强制要求所有列的长度必须相同，防止了一些常见的数据不一致问题，有助于提高数据的质量。
3. **列的类型推断：** tibble 会自动推断列的数据类型，而 data.frame 通常将字符向量视为因子，这可能导致数据处理问题。
4. **更好的子集选择：** tibble 具有更好的列子集选择语法，可以使用 dplyr 包中的函数进行更灵活的数据操作。

下面是一个示例，展示了 tibble 相对于 data.frame 的优势：

```
## 代码写这里，并运行；

# Creating a data.frame
df = data.frame(
  name = c("Alice", "Bob", "Charlie"),
  age = c(25, 30, 35),
  stringsAsFactors = FALSE
)

# Creating a tibble
tb = tibble(
  name = c("Alice", "Bob", "Charlie"),
  age = c(25, 30, 35)
)

# Print data.frame
print(df)
```

```
##      name age
## 1   Alice  25
## 2     Bob  30
## 3 Charlie  35
```

```
# Print tibble
print(tb)
```

```
## # A tibble: 3 x 2
##   name      age
##   <chr>   <dbl>
## 1 Alice      25
## 2 Bob        30
## 3 Charlie    35
```

Partly based on R Documentation and the “DigitalOcean” & “CSDN” forum.

0.6 练习与作业 3：IO

- **提供代码，正确读取以下文件：**

注：数据在当前目录下的 `data/` 子目录里

- Table0.txt
- Table1.txt
- Table2.txt
- Table3.txt
- Table4.txt
- Table5.txt
- Table6.txt
- states1.csv
- states2.csv

注 2：每个文件读取需要提供两种方法，一种是利用系统自带函数，另一种是 `readr` 包的函数；

注 3：请注意观察每列的数据特点，并将之读取为合理的数据类型；比如体重 `1,77` 可理解为 `1.77` 米，并将之读取为 `col_double()` 类型；

```
## 用系统自带函数，并显示读取的内容；

# Table0
read.table(file = "data/Table0.txt",
           col.names = c("Name", "Age", "Height", "Weight", "Gender"))
```

```
##       Name Age Height Weight Gender
## 1     Alex  25    177     57      F
## 2    Lilly  31    163     69      F
```

```
## 3     Mark  23    190     83      M
## 4   Oliver  52    179     75      M
## 5   Martha  76    163     70      F
## 6    Lucas  49    183     83      M
## 7 Caroline  26    164     53      F
```

```
# Table1
read.table(file = "data/Table1.txt",
           header = 1)
```

```
##       Name Age Height Weight Sex
## 1     Alex  25    177     57   F
## 2    Lilly  31    163     69   F
## 3     Mark  23    190     83   M
## 4   Oliver  52    179     75   M
## 5   Martha  76    163     70   F
## 6    Lucas  49    183     83   M
## 7 Caroline  26    164     53   F
```

```
# Table2
read.table(file = "data/Table2.txt",
           skip = 2,
           header = 3,
           quote = "/")
```

```
##       Name Age Height Weight Sex
## 1     Alex  25    177     57   F
## 2    Lilly  31    163     69   F
## 3     Mark  23    190     83   M
## 4   Oliver  52    179     75   M
## 5   Martha  76    163     70   F
## 6    Lucas  49    183     83   M
## 7 Caroline  26    164     53   F
```

```r
# Table3
table3 = read.table(
  file = "data/Table3.txt",
  skip = 2,
  header = 3,
  stringsAsFactors = FALSE
)

# Replace the odd character with NA
table3[table3 == "--" | table3 == "*"] = NA

table3$Age = as.numeric(table3$Age)
table3$Height = as.numeric(table3$Height)
table3$Weight = as.numeric(table3$Weight)
```

```
## Warning: NAs introduced by coercion
```

```r
print(table3)
```

```
##       Name Age Height Weight Sex
## 1     Alex  25    177     57   F
## 2    Lilly  31     NA     69   F
## 3     Mark  NA    190     83   M
## 4   Oliver  52    179     75   M
## 5   Martha  76     NA     70   F
## 6    Lucas  49    183     NA   M
## 7 Caroline  26    164     53   F
```

```r
# Table4
table4 = read.table(
  file = "data/Table4.txt",
  header = 1,
  stringsAsFactors = FALSE
```

```
)

table4[table4 == "--" | table4 == "*" | table4 == "**"] <- NA

table4$Age <- as.numeric(gsub(",", ".", table4$Age))
table4$Height <- as.numeric(gsub(",", ".", table4$Height))
table4$Weight <- as.numeric(table4$Weight)

print(table4)
```

```
##       Name Age Height Weight Sex
## 1     Alex  25   1.77     57   F
## 2    Lilly  31     NA     69   F
## 3     Mark  NA   1.90     83   M
## 4   Oliver  52   1.79     75   M
## 5   Martha  76     NA     70   F
## 6    Lucas  49   1.83     NA   M
## 7 Caroline  26   1.64     53   F
```

```
# Table5
read.csv2(file = "data/Table5.txt",
          header = 1)
```

```
##       Name Age Height Weight Sex
## 1     Alex  25   1.77     57   F
## 2    Lilly  31     NA     69   F
## 3     Mark  --   1.90     83   M
## 4   Oliver  52   1.79     75   M
## 5   Martha  76     NA     70   F
## 6    Lucas  49   1.83     **   M
## 7 Caroline  26   1.64     53   F
```

```r
# Table6
# To avoid error, I should delete the " " after "@"
lines_01 = readLines("data/Table6.txt")
cleaned_lines_01 = character(0)
for (line_01 in lines_01) {
  cleaned_line_01 = sub("@ ", "@", line_01)
  cleaned_lines_01 = c(cleaned_lines_01, cleaned_line_01)
}
cleaned_text_01 = paste(cleaned_lines_01, collapse = "\n")
read.table(text = cleaned_text_01,
           skip = 2,
           header = 1)
```

```
##          Name Age Height Weight         Sex
## 1        Alex  25    177     57      F@Boss
## 2       Lilly  31    163     69 F@Secretary
## 3        Mark  23    190     83           M
## 4      Oliver  52    179     75           M
## 5     Martha  76    163     70           F
## 6       Lucas  49    183     83           M
## 7   Caroline  26    164     53           F
## 8        Alex  25    177     57           F
## 9       Lilly  31    163     69           F
## 10       Mark  23    190     83           M
## 11     Oliver  52    179     75           M
## 12    Martha  76    163     70           F
## 13     Lucas  49    183     83           M
## 14  Caroline  26    164     53           F
## 15       Alex  25    177     57           F
## 16      Lilly  31    163     69           F
## 17       Mark  23    190     83           M
## 18     Oliver  52    179     75           M
## 19    Martha  76    163     70           F
```

```
## 20     Lucas  49    183     83           M
## 21  Caroline  26    164     53           F
## 22      Alex  25    177     57           F
## 23     Lilly  31    163     69           F
## 24      Mark  23    190     83           M
## 25    Oliver  52    179     75           M
## 26    Martha  76    163     70           F
## 27     Lucas  49    183     83           M
## 28  Caroline  26    164     53           F
## 29      Alex  25    177     57           F
## 30     Lilly  31    163     69           F
## 31      Mark  23    190     83           M
## 32    Oliver  52    179     75           M
## 33    Martha  76    163     70           F
## 34     Lucas  49    183     83           M
## 35  Caroline  26    164     53           F
## 36      Alex  25    177     57           F
## 37     Lilly  31    163     69           F
## 38      Mark  23    190     83           M
## 39    Oliver  52    179     75           M
## 40    Martha  76    163     70           F
## 41     Lucas  49    183     83           M
## 42  Caroline  26    164     53           F
## 43      Alex  25    177     57           F
## 44     Lilly  31    163     69           F
## 45      Mark  23    190     83           M
## 46    Oliver  52    179     75           M
## 47    Martha  76    163     70           F
## 48     Lucas  49    183     83           M
## 49  Caroline  26    164     53           F
## 50      Alex  25    177     57           F
## 51     Lilly  31    163     69           F
## 52      Mark  23    190     83           M
```

```
## 53    Oliver  52    179     75           M
## 54    Martha  76    163     70           F
## 55     Lucas  49    183     83           M
## 56  Caroline  26    164     53           F
## 57      Alex  25    177     57           F
## 58     Lilly  31    163     69           F
## 59      Mark  23    190     83           M
## 60    Oliver  52    179     75           M
## 61    Martha  76    163     70           F
## 62     Lucas  49    183     83           M
## 63  Caroline  26    164     53           F
## 64      Alex  25    177     57           F
## 65     Lilly  31    163     69           F
## 66      Mark  23    190     83           M
## 67    Oliver  52    179     75           M
## 68    Martha  76    163     70           F
## 69     Lucas  49    183     83           M
## 70  Caroline  26    164     53           F
## 71      Alex  25    177     57           F
## 72     Lilly  31    163     69           F
## 73      Mark  23    190     83           M
## 74    Oliver  52    179     75           M
## 75    Martha  76    163     70           F
## 76     Lucas  49    183     83           M
## 77  Caroline  26    164     53           F
## 78      Alex  25    177     57           F
## 79     Lilly  31    163     69           F
## 80      Mark  23    190     83           M
## 81    Oliver  52    179     75           M
## 82    Martha  76    163     70           F
## 83     Lucas  49    183     83           M
## 84  Caroline  26    164     53           F
## 85      Alex  25    177     57           F
```

```
## 86     Lilly  31    163     69           F
## 87      Mark  23    190     83           M
## 88    Oliver  52    179     75           M
## 89    Martha  76    163     70           F
## 90     Lucas  49    183     83           M
## 91  Caroline  26    164     53           F
## 92      Alex  25    177     57           F
## 93     Lilly  31    163     69           F
## 94      Mark  23    190     83           M
## 95    Oliver  52    179     75           M
## 96    Martha  76    163     70           F
## 97     Lucas  49    183     83           M
## 98  Caroline  26    164     53           F
## 99      Alex  25    177     57           F
## 100    Lilly  31    163     69           F
## 101     Mark  23    190     83           M
## 102   Oliver  52    179     75           M
## 103   Martha  76    163     70           F
## 104    Lucas  49    183     83           M
## 105 Caroline  26    164     53           F
```

```
read.csv(file = "data/states1.csv")
```

```
##                X Population Income Illiteracy Life.Exp Murder HS.Grad Frost
## 1        Alabama       3615   3624        2.1    69.05   15.1    41.3    20
## 2         Alaska        365   6315        1.5    69.31   11.3    66.7   152
## 3        Arizona       2212   4530        1.8    70.55    7.8    58.1    15
## 4       Arkansas       2110   3378        1.9    70.66   10.1    39.9    65
## 5     California      21198   5114        1.1    71.71   10.3    62.6    20
## 6       Colorado       2541   4884        0.7    72.06    6.8    63.9   166
## 7    Connecticut       3100   5348        1.1    72.48    3.1    56.0   139
## 8       Delaware        579   4809        0.9    70.06    6.2    54.6   103
## 9        Florida       8277   4815        1.3    70.66   10.7    52.6    11
## 10       Georgia       4931   4091        2.0    68.54   13.9    40.6    60
```

```
## 11          Hawaii   868  4963        1.9    73.60   6.2    61.9    0
## 12           Idaho   813  4119        0.6    71.87   5.3    59.5  126
## 13        Illinois 11197  5107        0.9    70.14  10.3    52.6  127
## 14         Indiana  5313  4458        0.7    70.88   7.1    52.9  122
## 15            Iowa  2861  4628        0.5    72.56   2.3    59.0  140
## 16          Kansas  2280  4669        0.6    72.58   4.5    59.9  114
## 17        Kentucky  3387  3712        1.6    70.10  10.6    38.5   95
## 18       Louisiana  3806  3545        2.8    68.76  13.2    42.2   12
## 19           Maine  1058  3694        0.7    70.39   2.7    54.7  161
## 20        Maryland  4122  5299        0.9    70.22   8.5    52.3  101
## 21   Massachusetts  5814  4755        1.1    71.83   3.3    58.5  103
## 22        Michigan  9111  4751        0.9    70.63  11.1    52.8  125
## 23       Minnesota  3921  4675        0.6    72.96   2.3    57.6  160
## 24     Mississippi  2341  3098        2.4    68.09  12.5    41.0   50
## 25        Missouri  4767  4254        0.8    70.69   9.3    48.8  108
## 26         Montana   746  4347        0.6    70.56   5.0    59.2  155
## 27        Nebraska  1544  4508        0.6    72.60   2.9    59.3  139
## 28          Nevada   590  5149        0.5    69.03  11.5    65.2  188
## 29   New Hampshire   812  4281        0.7    71.23   3.3    57.6  174
## 30      New Jersey  7333  5237        1.1    70.93   5.2    52.5  115
## 31      New Mexico  1144  3601        2.2    70.32   9.7    55.2  120
## 32        New York 18076  4903        1.4    70.55  10.9    52.7   82
## 33  North Carolina  5441  3875        1.8    69.21  11.1    38.5   80
## 34    North Dakota   637  5087        0.8    72.78   1.4    50.3  186
## 35            Ohio 10735  4561        0.8    70.82   7.4    53.2  124
## 36        Oklahoma  2715  3983        1.1    71.42   6.4    51.6   82
## 37          Oregon  2284  4660        0.6    72.13   4.2    60.0   44
## 38    Pennsylvania 11860  4449        1.0    70.43   6.1    50.2  126
## 39    Rhode Island   931  4558        1.3    71.90   2.4    46.4  127
## 40  South Carolina  2816  3635        2.3    67.96  11.6    37.8   65
## 41    South Dakota   681  4167        0.5    72.08   1.7    53.3  172
## 42       Tennessee  4173  3821        1.7    70.11  11.0    41.8   70
## 43           Texas 12237  4188        2.2    70.90  12.2    47.4   35
```

```
## 44           Utah       1203   4022        0.6    72.90    4.5    67.3   137
## 45        Vermont        472   3907        0.6    71.64    5.5    57.1   168
## 46       Virginia       4981   4701        1.4    70.08    9.5    47.8    85
## 47     Washington       3559   4864        0.6    71.72    4.3    63.5    32
## 48  West Virginia       1799   3617        1.4    69.48    6.7    41.6   100
## 49      Wisconsin       4589   4468        0.7    72.48    3.0    54.5   149
## 50        Wyoming        376   4566        0.6    70.29    6.9    62.9   173
##       Area
## 1   50708
## 2  566432
## 3  113417
## 4   51945
## 5  156361
## 6  103766
## 7    4862
## 8    1982
## 9   54090
## 10  58073
## 11   6425
## 12  82677
## 13  55748
## 14  36097
## 15  55941
## 16  81787
## 17  39650
## 18  44930
## 19  30920
## 20   9891
## 21   7826
## 22  56817
## 23  79289
## 24  47296
## 25  68995
```

```
## 26 145587
## 27  76483
## 28 109889
## 29   9027
## 30   7521
## 31 121412
## 32  47831
## 33  48798
## 34  69273
## 35  40975
## 36  68782
## 37  96184
## 38  44966
## 39   1049
## 40  30225
## 41  75955
## 42  41328
## 43 262134
## 44  82096
## 45   9267
## 46  39780
## 47  66570
## 48  24070
## 49  54464
## 50  97203
```

```
read.csv("data/states2.csv",
           sep = ";")
```

```
##                 X Population Income Illiteracy Life.Exp Murder HS.Grad Frost
## 1         Alabama       3615   3624        2,1    69,05   15,1    41,3    20
## 2          Alaska        365   6315        1,5    69,31   11,3    66,7   152
## 3         Arizona       2212   4530        1,8    70,55    7,8    58,1    15
## 4        Arkansas       2110   3378        1,9    70,66   10,1    39,9    65
```

## 5	California	21198	5114	1,1	71,71	10,3	62,6	20
## 6	Colorado	2541	4884	0,7	72,06	6,8	63,9	166
## 7	Connecticut	3100	5348	1,1	72,48	3,1	56	139
## 8	Delaware	579	4809	0,9	70,06	6,2	54,6	103
## 9	Florida	8277	4815	1,3	70,66	10,7	52,6	11
## 10	Georgia	4931	4091	2	68,54	13,9	40,6	60
## 11	Hawaii	868	4963	1,9	73,6	6,2	61,9	0
## 12	Idaho	813	4119	0,6	71,87	5,3	59,5	126
## 13	Illinois	11197	5107	0,9	70,14	10,3	52,6	127
## 14	Indiana	5313	4458	0,7	70,88	7,1	52,9	122
## 15	Iowa	2861	4628	0,5	72,56	2,3	59	140
## 16	Kansas	2280	4669	0,6	72,58	4,5	59,9	114
## 17	Kentucky	3387	3712	1,6	70,1	10,6	38,5	95
## 18	Louisiana	3806	3545	2,8	68,76	13,2	42,2	12
## 19	Maine	1058	3694	0,7	70,39	2,7	54,7	161
## 20	Maryland	4122	5299	0,9	70,22	8,5	52,3	101
## 21	Massachusetts	5814	4755	1,1	71,83	3,3	58,5	103
## 22	Michigan	9111	4751	0,9	70,63	11,1	52,8	125
## 23	Minnesota	3921	4675	0,6	72,96	2,3	57,6	160
## 24	Mississippi	2341	3098	2,4	68,09	12,5	41	50
## 25	Missouri	4767	4254	0,8	70,69	9,3	48,8	108
## 26	Montana	746	4347	0,6	70,56	5	59,2	155
## 27	Nebraska	1544	4508	0,6	72,6	2,9	59,3	139
## 28	Nevada	590	5149	0,5	69,03	11,5	65,2	188
## 29	New Hampshire	812	4281	0,7	71,23	3,3	57,6	174
## 30	New Jersey	7333	5237	1,1	70,93	5,2	52,5	115
## 31	New Mexico	1144	3601	2,2	70,32	9,7	55,2	120
## 32	New York	18076	4903	1,4	70,55	10,9	52,7	82
## 33	North Carolina	5441	3875	1,8	69,21	11,1	38,5	80
## 34	North Dakota	637	5087	0,8	72,78	1,4	50,3	186
## 35	Ohio	10735	4561	0,8	70,82	7,4	53,2	124
## 36	Oklahoma	2715	3983	1,1	71,42	6,4	51,6	82
## 37	Oregon	2284	4660	0,6	72,13	4,2	60	44

```
## 38   Pennsylvania      11860   4449          1    70,43    6,1    50,2   126
## 39   Rhode Island        931   4558        1,3     71,9    2,4    46,4   127
## 40 South Carolina       2816   3635        2,3    67,96   11,6    37,8    65
## 41   South Dakota        681   4167        0,5    72,08    1,7    53,3   172
## 42      Tennessee       4173   3821        1,7    70,11     11    41,8    70
## 43          Texas      12237   4188        2,2     70,9   12,2    47,4    35
## 44           Utah       1203   4022        0,6     72,9    4,5    67,3   137
## 45        Vermont        472   3907        0,6    71,64    5,5    57,1   168
## 46       Virginia       4981   4701        1,4    70,08    9,5    47,8    85
## 47     Washington       3559   4864        0,6    71,72    4,3    63,5    32
## 48  West Virginia       1799   3617        1,4    69,48    6,7    41,6   100
## 49      Wisconsin       4589   4468        0,7    72,48      3    54,5   149
## 50        Wyoming        376   4566        0,6    70,29    6,9    62,9   173
##      Area
## 1   50708
## 2  566432
## 3  113417
## 4   51945
## 5  156361
## 6  103766
## 7    4862
## 8    1982
## 9   54090
## 10  58073
## 11   6425
## 12  82677
## 13  55748
## 14  36097
## 15  55941
## 16  81787
## 17  39650
## 18  44930
## 19  30920
```

```
## 20   9891
## 21   7826
## 22  56817
## 23  79289
## 24  47296
## 25  68995
## 26 145587
## 27  76483
## 28 109889
## 29   9027
## 30   7521
## 31 121412
## 32  47831
## 33  48798
## 34  69273
## 35  40975
## 36  68782
## 37  96184
## 38  44966
## 39   1049
## 40  30225
## 41  75955
## 42  41328
## 43 262134
## 44  82096
## 45   9267
## 46  39780
## 47  66570
## 48  24070
## 49  54464
## 50  97203
```

```
## 用 readr 包的函数读取，并显示读取的内容;
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
readr::read_table(file = "data/Table0.txt",
                  col_names = c("Name", "Age", "Height", "Weight", "Gender"),
                  show_col_types = FALSE)
```

```
## # A tibble: 7 x 5
##   Name       Age Height Weight Gender
##   <chr>    <dbl>  <dbl>  <dbl> <chr>
## 1 Alex        25    177     57 F
## 2 Lilly       31    163     69 F
## 3 Mark        23    190     83 M
## 4 Oliver      52    179     75 M
## 5 Martha      76    163     70 F
## 6 Lucas       49    183     83 M
## 7 Caroline    26    164     53 F
```

```
# Table1
readr::read_table(file = "data/Table1.txt",
                  show_col_types = FALSE)
```

```
## # A tibble: 7 x 5
##   Name       Age Height Weight Sex
##   <chr>    <dbl>  <dbl>  <dbl> <chr>
## 1 Alex        25    177     57 F
## 2 Lilly       31    163     69 F
## 3 Mark        23    190     83 M
## 4 Oliver      52    179     75 M
## 5 Martha      76    163     70 F
## 6 Lucas       49    183     83 M
## 7 Caroline    26    164     53 F
```

```
# Table2

table02 = readr::read_table(
  file = "data/Table2.txt",
  col_names = c("Name", "Age", "Height", "Weight", "Sex"),
  col_types = cols(
    Name = col_character(),
    Age = col_double(),
    Height = col_double(),
    Weight = col_double(),
    Sex = col_character()
  ),
  skip = 1
)
```

```
## Warning: 3 parsing failures.
## row    col expected actual              file
##   1 Age    a double Age    'data/Table2.txt'
##   1 Height a double Height 'data/Table2.txt'
##   1 Weight a double Weight 'data/Table2.txt'
```

```r
table02$Name = gsub("/", "", table02$Name)
table02$Sex = gsub("/", "", table02$Sex)

print(table02)
```

```
## # A tibble: 8 x 5
##   Name       Age Height Weight Sex
##   <chr>    <dbl>  <dbl>  <dbl> <chr>
## 1 Name        NA     NA     NA Sex
## 2 Alex        25    177     57 F
## 3 Lilly       31    163     69 F
## 4 Mark        23    190     83 M
## 5 Oliver      52    179     75 M
## 6 Martha      76    163     70 F
## 7 Lucas       49    183     83 M
## 8 Caroline    26    164     53 F
```

```r
# Table3
table03 = readr::read_table(file = "data/Table3.txt",
           skip = 2)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Name = col_character(),
##   Age = col_character(),
##   Height = col_character(),
##   Weight = col_character(),
##   Sex = col_character()
## )
```

```r
# Replace the odd character with NA
table03[table03 == "--" | table03 == "*"] = NA

table03$Age = as.numeric(table03$Age)
table03$Height = as.numeric(table3$Height)
table03$Weight = as.numeric(table3$Weight)

print(table03)
```

```
## # A tibble: 7 x 5
##   Name       Age Height Weight Sex
##   <chr>    <dbl>  <dbl>  <dbl> <chr>
## 1 Alex        25    177     57 F
## 2 Lilly       31     NA     69 F
## 3 Mark        NA    190     83 M
## 4 Oliver      52    179     75 M
## 5 Martha      76     NA     70 F
## 6 Lucas       49    183     NA M
## 7 Caroline    26    164     53 F
```

```r
# Table4
table04 = readr::read_table(
  file = "data/Table4.txt"
)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Name = col_character(),
##   Age = col_character(),
##   Height = col_character(),
##   Weight = col_character(),
##   Sex = col_character()
## )
```

```r
table04[table04 == "--" | table04 == "*" | table04 == "**"] <- NA

table04$Age = as.numeric(gsub(",", ".", table04$Age))
table04$Height = as.numeric(gsub(",", ".", table04$Height))
table04$Weight = as.numeric(table4$Weight)

print(table04)
```

```
## # A tibble: 7 x 5
##   Name       Age Height Weight Sex
##   <chr>    <dbl>  <dbl>  <dbl> <chr>
## 1 Alex        25   1.77     57 F
## 2 Lilly       31  NA        69 F
## 3 Mark        NA   1.9      83 M
## 4 Oliver      52   1.79     75 M
## 5 Martha      76  NA        70 F
## 6 Lucas       49   1.83     NA M
## 7 Caroline    26   1.64     53 F
```

```r
# Table5
readr::read_csv2(file = "data/Table5.txt",
                 show_col_types = FALSE)
```

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

```
## # A tibble: 7 x 5
##   Name     Age   Height Weight Sex
##   <chr>    <chr>  <dbl> <chr>  <chr>
## 1 Alex     25      1.77 57     F
## 2 Lilly    31     NA    69     F
## 3 Mark     --      1.9  83     M
## 4 Oliver   52      1.79 75     M
## 5 Martha   76     NA    70     F
```

```
## 6 Lucas    49      1.83 **     M
## 7 Caroline 26      1.64 53     F
```

```
# Table6
# To avoid error, I should delete the " " after "@"
lines_01 = readLines("data/Table6.txt")
cleaned_lines_01 = character(0)
for (line_01 in lines_01) {
  cleaned_line_01 = sub("@ ", "@", line_01)
  cleaned_lines_01 = c(cleaned_lines_01, cleaned_line_01)
}
cleaned_text_01 = paste(cleaned_lines_01, collapse = "\n")
readr::read_table(cleaned_text_01,
           skip = 2)
```

```
## # A tibble: 105 x 5
##    Name       Age Height Weight Sex
##    <chr>    <dbl>  <dbl>  <dbl> <chr>
##  1 Alex        25    177     57 F@Boss
##  2 Lilly       31    163     69 F@Secretary
##  3 Mark        23    190     83 M
##  4 Oliver      52    179     75 M
##  5 Martha      76    163     70 F
##  6 Lucas       49    183     83 M
##  7 Caroline    26    164     53 F
##  8 Alex        25    177     57 F
##  9 Lilly       31    163     69 F
## 10 Mark        23    190     83 M
## # i 95 more rows
```

```
readr::read_csv(file = "data/states1.csv")
```

```
## New names:
## * `` -> `...1`
```

```
## Rows: 50 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ...1
## dbl (8): Population, Income, Illiteracy, Life Exp, Murder, HS Grad, Frost, Area
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

## # A tibble: 50 x 9
##    ...1    Population Income Illiteracy `Life Exp` Murder `HS Grad` Frost   Area
##    <chr>        <dbl>  <dbl>      <dbl>      <dbl>  <dbl>     <dbl> <dbl>  <dbl>
##  1 Alabama       3615   3624        2.1       69.0   15.1      41.3    20  50708
##  2 Alaska         365   6315        1.5       69.3   11.3      66.7   152 566432
##  3 Arizona       2212   4530        1.8       70.6    7.8      58.1    15 113417
##  4 Arkans~       2110   3378        1.9       70.7   10.1      39.9    65  51945
##  5 Califo~      21198   5114        1.1       71.7   10.3      62.6    20 156361
##  6 Colora~       2541   4884        0.7       72.1    6.8      63.9   166 103766
##  7 Connec~       3100   5348        1.1       72.5    3.1      56     139   4862
##  8 Delawa~        579   4809        0.9       70.1    6.2      54.6   103   1982
##  9 Florida       8277   4815        1.3       70.7   10.7      52.6    11  54090
## 10 Georgia       4931   4091        2         68.5   13.9      40.6    60  58073
## # i 40 more rows
```

```
readr::read_delim("data/states2.csv",
                  delim = ";",
                  escape_double = FALSE,
                  trim_ws = TRUE)
```

```
## New names:
## Rows: 50 Columns: 9
## -- Column specification
## -------------------------------------------------------- Delimiter: ";" chr
## (2): ...1, Illiteracy dbl (4): Population, Income, Frost, Area num (3): Life
```

```
## Exp, Murder, HS Grad
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
## # A tibble: 50 x 9
##    ...1    Population Income Illiteracy `Life Exp` Murder `HS Grad` Frost   Area
##    <chr>        <dbl>  <dbl> <chr>           <dbl>  <dbl>     <dbl> <dbl>  <dbl>
##  1 Alabama       3615   3624 2,1              6905    151       413    20  50708
##  2 Alaska         365   6315 1,5              6931    113       667   152 566432
##  3 Arizona       2212   4530 1,8              7055     78       581    15 113417
##  4 Arkans~       2110   3378 1,9              7066    101       399    65  51945
##  5 Califo~      21198   5114 1,1              7171    103       626    20 156361
##  6 Colora~       2541   4884 0,7              7206     68       639   166 103766
##  7 Connec~       3100   5348 1,1              7248     31        56   139   4862
##  8 Delawa~        579   4809 0,9              7006     62       546   103   1982
##  9 Florida       8277   4815 1,3              7066    107       526    11  54090
## 10 Georgia       4931   4091 2                6854    139       406    60  58073
## # i 40 more rows
```

0.7 Some bugs haven't been fixed:

- The white space before "@" in "Table 06.txt". I can only delete it to avoid errors;
- The "/" character of "Table2.txt" in "readr" section;
- Transform the data type such as "1,77"(character) into "1.77"(numeric).

All the bugs are fixed in this version of homework.