talk03 练习与作业

目录

0.1 练习和作业说明 1

0.2 talk03 内容回顾 1

0.3 练习与作业：用户验证 2

0.4 练习与作业 1，`data.frame` 2

0.5 练习与作业 2，`tibble` 15

0.6 练习与作业 3：IO 23

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
    - 增减行、列

        - 合并
    - 常用相关函数
        - nrow, ncol, dim , str , head, tail
    - data.frame 和 tibble 的不同
    - 高级技巧：
        - with, within
- IO
    - 系统自带函数
    - readr 带的函数
    - 不同格式的读取
    - 从网络、压缩文件读取

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] "sicheng.wu"
```

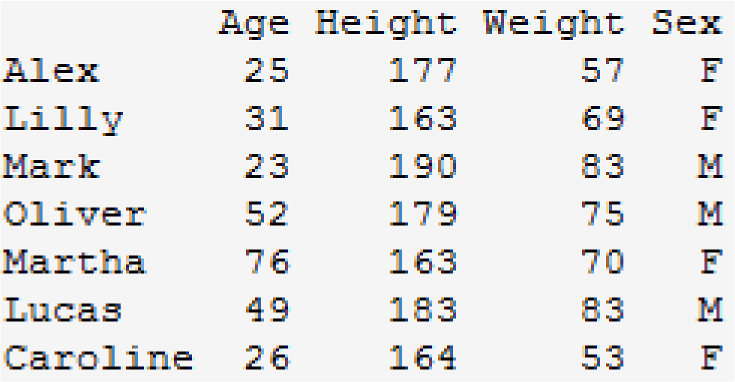
```
Sys.getenv("HOME")
```

```
## [1] "/home/vkorpela"
```

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
```

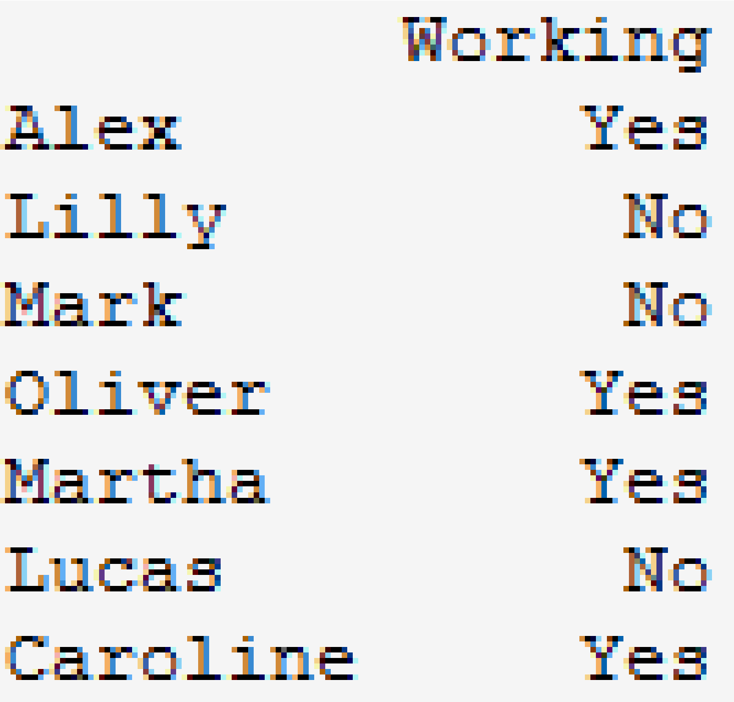
```
## 先生成前三列；
df1 <- data.frame(
  Age = c(25, 31, 23, 52, 76, 49, 26),
  Height = c(177, 163,  190, 179, 163, 183, 164),
  Weight = c(57, 69, 83, 75, 70, 83, 53),
  row.names = c("Alex", "Lilly", "Mark", "Oliver", "Martha", "Lucas", "Caroline")
)

## 再插入第四列
df1 <- cbind(df1, Sex = c("F", "F", "M", "M", "F", "M", "F"))

## 显示最终结果
df1
```

```
##          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
```

- **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
```

```
## 生成 data.frame
df2 <- data.frame(
  Working = c("Yes", "No", "No", "Yes", "Yes", "No", "Yes"),
  row.names = c("Alex", "Lilly", "Mark", "Oliver", "Martha", "Lucas", "Caroline"),
  stringsAsFactors = FALSE
)

## 显示结果
df2
```

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

```
## 显示 Working 列的性质
class(df2[["Working"]])
```

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

- **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；
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
```

```
as.data.frame(state.center)
```

```
##             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**

```
## 代码写这里，并运行；
df3 <- as.data.frame(matrix(sample(1:1000, 250), nrow = 50))
row.names(df3) <- paste("row_", 1:50, sep = "")
df3
```

```
##          V1  V2  V3  V4  V5
## row_1   543 689 173 565 117
## row_2   557 924 792 604 860
## row_3  148 878 505 752 979
## row_4   67 956  29 578 186
## row_5   39 793 843  68 510
## row_6  675 866 977 218 767
## row_7  167 444 519 805 409
## row_8  448  63 183  44 536
## row_9  922 644 555 834 286
## row_10 635 554 103  48 363
## row_11 327 814 200 294 535
## row_12 730 232 424 972 341
## row_13 976 763 662 313 734
## row_14 141 126 296 548 274
## row_15   5 640 314 431 652
## row_16 940 697 854 667 964
## row_17 307  30 276 191 374
## row_18 449 490 213 196 538
## row_19 458 931 883 669 748
## row_20 471 992 269  42 967
## row_21 695 244 220 900 339
## row_22 455 451 710   9 806
## row_23 300  88 162 680 518
## row_24 589 418 284 758 707
## row_25  14 171 130  69   7
## row_26 177 396 122 551 641
## row_27 387 737 985 568 168
## row_28 395 248 506 798 714
## row_29 663 902 509  49 877
## row_30  25 440 277 782 865
## row_31 959 788 593  46 383
## row_32 534 957 833 497 414
## row_33 165 920  22 594 929
## row_34 620 174 187 637 596
## row_35 880 115 308 871 422
## row_36 225 616 717  65 960
## row_37 818 754 508 665 840
## row_38 687 152 154 827  74
## row_39 362 197 121  17 693
## row_40  37 786 226 569 634
## row_41 178 229 668 100 217
## row_42 164 394 467 359 545
## row_43 139 373 728 310 275
## row_44 691 655 756 140  15
## row_45 330 265 622 495 513
## row_46 511 795 825 723 618
## row_47 975 499  64 886 888
## row_48  31 812 361 842  76
## row_49 379 331 654 925 990
## row_50 781 911 958 790 982
```

- **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；
class(VADeaths)
```

```
## [1] "matrix" "array"
```

```
df4 <- as.data.frame(VADeaths)


df4 <- cbind(df4, Total = rowSums(df4))
df4 <- df4[, c(5, 1, 2, 3, 4)]
```

```
df4
```

```
##       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；
df5 <- as.data.frame(swiss)[c(1:3, 10:13), c("Examination", "Education", "Infant.Mortal
df5["Sarine", "Infant.Mortality"] <- NA
df5 <- cbind(df5, Mean = rowMeans(df5))

df5
```

```
##              Examination Education Infant.Mortality     Mean
## Courtelary            15        12             22.2 16.40000
## Delemont               6         9             22.2 12.40000
## Franches-Mnt           5         5             20.2 10.06667
## Sarine                16        13               NA       NA
## Veveyse               14         6             24.5 14.83333
## Aigle                 21        12             16.5 16.50000
## Aubonne               14         7             19.1 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；
Id <- LETTERS
x <- seq(1, 43, along.with = Id)
y <- seq(-20, 0, along.with = Id)
df6 <- data.frame(Id, x, y)
df6
```

```
##    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：along.with 参数的意义是使 seq 输出的向量长度与 along.with 给定的向量一致。例如说 seq(0, 10, along.with = 1:101) 就会自动计算步长，输出长度为 101 的向量。

```
## 代码写这里，并运行；
seq(0, 10, along.with = 1:101)
```

```
##   [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
```

- **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
```

```
## 代码写这里，并运行；
df1 <- data.frame(
  Id = 1:4,
  Age = c(14, 12, 15, 10)
)

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

M <- merge(df1, df2, by.df1 = "Id", by.df2 = "Id")
M
```

```
##   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**

```
## 代码写这里，并运行；
M["Code"] <- NULL
M
```

```
##   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：前三个可以，第四个不可以

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
```

```
## Warning in system("timedatectl", intern = TRUE): running command 'timedatectl'
## had status 1
```

```
##
## 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
```

```
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 2022-09-16 10:39:23 2022-10-14     1 0.252 h
##  2 2022-09-16 12:26:27 2022-09-18     2 0.681 q
##  3 2022-09-16 12:14:59 2022-09-18     3 0.648 j
##  4 2022-09-16 23:14:04 2022-09-16     4 0.196 e
##  5 2022-09-16 12:21:42 2022-09-20     5 0.926 s
##  6 2022-09-16 13:57:23 2022-10-09     6 0.630 l
##  7 2022-09-17 07:09:20 2022-10-04     7 0.893 g
##  8 2022-09-17 06:17:31 2022-09-28     8 0.729 a
##  9 2022-09-16 10:10:00 2022-10-15     9 0.370 v
## 10 2022-09-17 04:00:57 2022-10-11    10 0.556 g
## # ... with 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;
df <- tibble(
  x = runif(5),
  y = rnorm(5)
)

df["x"]
```

```
## # A tibble: 5 x 1
##          x
##      <dbl>
## 1 0.928
## 2 0.330
## 3 0.461
## 4 0.530
## 5 0.0198
```

```
df[["y"]]
```

```
## [1] -1.57044870 -0.63765885  0.08137459 -0.21515831 -1.99011782
```

- **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；
tb <- add_row(tb, name = sample(LETTERS, 3), age = sample(18:60, 3), salary = sample(10

tb
```

```
## # A tibble: 3 x 3
##   name    age salary
##   <chr> <int>  <dbl>
## 1 A        33  21900
## 2 Z        31  31000
## 3 V        30  39200
```

- ** 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：因为 `data.frame` 可以循环使用长度能够整除行数的向量，而 `tibble` 只能重复使用长度为 1 的向量。

- **`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 可以将数据按列加载到环境变量中，便于直接访问。detach 则可以卸载 attach 加载的数据。

```
# 加载以前 Sepal.Length 无法直接访问，需要指明数据
iris[["Sepal.Length"]][1:5]
```

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

```
# 使用 attach 加载 iris 后可以直接通过环境变量访问 iris 中的列
attach(iris)
Sepal.Length[1:5]
```

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

```
# 使用 detach 卸载后重新回到开始时无法访问的状态
detach(iris)
```

- **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；
class(airquality)
```

```
## [1] "data.frame"
```

```
as_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
## # ... with 143 more rows
```

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

答：`tibble::enframe` 可以将向量转变为带有 `name` 列的 `tibble` 类型数据。其中 `name` 列内容会根据向量中元素的命名情况自动生成。

```r
# 若无命名，name 列为向量中的下标
unnamed <- 10:1
enframe(unnamed)
```

```
## # A tibble: 10 x 2
##     name value
##    <int> <int>
##  1     1    10
##  2     2     9
##  3     3     8
##  4     4     7
##  5     5     6
##   6      6      5
##   7      7      4
##   8      8      3
##   9      9      2
##  10     10      1
```

```
# 若有命名，name 列为命名内容
named <- 10:1
names(named) <- LETTERS[1:10]
enframe(named)
```

```
## # A tibble: 10 x 2
##     name  value
##     <chr> <int>
##   1 A        10
##   2 B         9
##   3 C         8
##   4 D         7
##   5 E         6
##   6 F         5
##   7 G         4
##   8 H         3
##   9 I         2
##  10 J         1
```

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

答：1. tibble 支持在创建的时候引用其中某一列的数据，data.frame 不可以；2. 取数据的子集时数据类型稳定为 tibble，而 data.frame 数据类型将随取的数据而定；

```
## 代码写这里，并运行；
## 1. 下面的代码可以正常执行 (而在 data.frame 中不可)
tb1 <- tibble(x = 1:10, y = 2 * x)
tb1
```

```
## # A tibble: 10 x 2
##        x     y
##    <int> <dbl>
##  1     1     2
##  2     2     4
##  3     3     6
##  4     4     8
##  5     5    10
##  6     6    12
##  7     7    14
##  8     8    16
##  9     9    18
## 10    10    20
```

```
## 2. data.frame 可能得到不同数据类型的 subset, 而 tibble 的 subset 一定为 tibble
df2 <- data.frame(v1 = 1:10, v2 = 10:1)
class(df2[1, 1])
```

```
## [1] "integer"
```

```
class(df2[1:2])
```

```
## [1] "data.frame"
```

```
tb2 <- tibble(v1 = 1:10, v2 = 10:1)
class(tb2[1, 1])
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
class(tb2[1:2])
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

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；

```
## 用系统自带函数，并显示读取的内容；
read.table("./data/Table0.txt", header = FALSE)
```

```
##         V1 V2  V3 V4 V5
## 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
read.table("./data/Table1.txt", header = TRUE)
```

```
##       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
read.table("./data/Table2.txt", header = TRUE, skip = 2, 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
read.table("./data/Table3.txt", header = TRUE, skip = 2, na.strings = c("--", "*", "**"
```

```
##       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
```

```
read.table("./data/Table4.txt", header = TRUE, dec = ",", na.strings = c("--", "*", "**
```

```
##       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
```

```
read.table("./data/Table5.txt", header = TRUE, sep = ";", dec = ",", na.strings = c("--
```

```
##       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
```

```
read.table("./data/Table6.txt", header = TRUE, skip = 2, comment.char = "@", nrows = 7)
```

```
##       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
read.csv("./data/states1.csv", row.names = 1)
```

```
##                Population Income Illiteracy Life.Exp Murder HS.Grad Frost
## Alabama              3615   3624        2.1    69.05   15.1    41.3    20
## Alaska                365   6315        1.5    69.31   11.3    66.7   152
## Arizona              2212   4530        1.8    70.55    7.8    58.1    15
## Arkansas             2110   3378        1.9    70.66   10.1    39.9    65
## California          21198   5114        1.1    71.71   10.3    62.6    20
## Colorado             2541   4884        0.7    72.06    6.8    63.9   166
## Connecticut          3100   5348        1.1    72.48    3.1    56.0   139
## Delaware              579   4809        0.9    70.06    6.2    54.6   103
## Florida              8277   4815        1.3    70.66   10.7    52.6    11
## Georgia              4931   4091        2.0    68.54   13.9    40.6    60
## Hawaii                868   4963        1.9    73.60    6.2    61.9     0
## Idaho                 813   4119        0.6    71.87    5.3    59.5   126
## Illinois            11197   5107        0.9    70.14   10.3    52.6   127
## Indiana              5313   4458        0.7    70.88    7.1    52.9   122
## Iowa                 2861   4628        0.5    72.56    2.3    59.0   140
## Kansas               2280   4669        0.6    72.58    4.5    59.9   114
## Kentucky             3387   3712        1.6    70.10   10.6    38.5    95
## Louisiana            3806   3545        2.8    68.76   13.2    42.2    12
## Maine                1058   3694        0.7    70.39    2.7    54.7   161
## Maryland             4122   5299        0.9    70.22    8.5    52.3   101
## Massachusetts        5814   4755        1.1    71.83    3.3    58.5   103
## Michigan             9111   4751        0.9    70.63   11.1    52.8   125
## Minnesota            3921   4675        0.6    72.96    2.3    57.6   160
## Mississippi          2341   3098        2.4    68.09   12.5    41.0    50
## Missouri             4767   4254        0.8    70.69    9.3    48.8   108
## Montana               746   4347        0.6    70.56    5.0    59.2   155
## Nebraska             1544   4508        0.6    72.60    2.9    59.3   139
## Nevada                590   5149        0.5    69.03   11.5    65.2   188
## New Hampshire         812   4281        0.7    71.23    3.3    57.6   174
## New Jersey           7333   5237        1.1    70.93    5.2    52.5   115
## New Mexico          1144   3601         2.2    70.32   9.7    55.2  120
## New York           18076   4903         1.4    70.55  10.9    52.7   82
## North Carolina      5441   3875         1.8    69.21  11.1    38.5   80
## North Dakota         637   5087         0.8    72.78   1.4    50.3  186
## Ohio               10735   4561         0.8    70.82   7.4    53.2  124
## Oklahoma            2715   3983         1.1    71.42   6.4    51.6   82
## Oregon              2284   4660         0.6    72.13   4.2    60.0   44
## Pennsylvania       11860   4449         1.0    70.43   6.1    50.2  126
## Rhode Island         931   4558         1.3    71.90   2.4    46.4  127
## South Carolina      2816   3635         2.3    67.96  11.6    37.8   65
## South Dakota         681   4167         0.5    72.08   1.7    53.3  172
## Tennessee           4173   3821         1.7    70.11  11.0    41.8   70
## Texas              12237   4188         2.2    70.90  12.2    47.4   35
## Utah                1203   4022         0.6    72.90   4.5    67.3  137
## Vermont              472   3907         0.6    71.64   5.5    57.1  168
## Virginia            4981   4701         1.4    70.08   9.5    47.8   85
## Washington          3559   4864         0.6    71.72   4.3    63.5   32
## West Virginia       1799   3617         1.4    69.48   6.7    41.6  100
## Wisconsin           4589   4468         0.7    72.48   3.0    54.5  149
## Wyoming              376   4566         0.6    70.29   6.9    62.9  173
##                  Area
## Alabama         50708
## Alaska         566432
## Arizona        113417
## Arkansas        51945
## California     156361
## Colorado       103766
## Connecticut      4862
## Delaware         1982
## Florida         54090
## Georgia         58073
## Hawaii           6425
## Idaho           82677
## Illinois        55748
## Indiana         36097
## Iowa            55941
## Kansas          81787
## Kentucky        39650
## Louisiana       44930
## Maine           30920
## Maryland         9891
## Massachusetts    7826
## Michigan        56817
## Minnesota       79289
## Mississippi     47296
## Missouri        68995
## Montana        145587
## Nebraska        76483
## Nevada         109889
## New Hampshire    9027
## New Jersey       7521
## New Mexico     121412
## New York        47831
## North Carolina  48798
## North Dakota    69273
## Ohio            40975
## Oklahoma        68782
## Oregon          96184
## Pennsylvania    44966
## Rhode Island     1049
## South Carolina  30225
## South Dakota    75955
## Tennessee       41328
## Texas          262134
## Utah            82096
## Vermont          9267
## Virginia        39780
## Washington      66570
## West Virginia   24070
## Wisconsin       54464
## Wyoming         97203
```

```r
read.csv2("./data/states2.csv", row.names = 1)
```

```
##                Population Income Illiteracy Life.Exp Murder HS.Grad Frost
## Alabama              3615   3624        2.1    69.05   15.1    41.3    20
## Alaska                365   6315        1.5    69.31   11.3    66.7   152
## Arizona              2212   4530        1.8    70.55    7.8    58.1    15
## Arkansas             2110   3378        1.9    70.66   10.1    39.9    65
## California          21198   5114        1.1    71.71   10.3    62.6    20
## Colorado             2541   4884        0.7    72.06    6.8    63.9   166
## Connecticut          3100   5348        1.1    72.48    3.1    56.0   139
## Delaware              579   4809        0.9    70.06    6.2    54.6   103
## Florida              8277   4815        1.3    70.66   10.7    52.6    11
## Georgia              4931   4091        2.0    68.54   13.9    40.6    60
## Hawaii                868   4963        1.9    73.60    6.2    61.9     0
## Idaho                 813   4119        0.6    71.87    5.3    59.5   126
## Illinois            11197   5107        0.9    70.14   10.3    52.6   127
## Indiana              5313   4458        0.7    70.88    7.1    52.9   122
## Iowa                 2861   4628        0.5    72.56    2.3    59.0   140
## Kansas               2280   4669        0.6    72.58    4.5    59.9   114
## Kentucky             3387   3712        1.6    70.10   10.6    38.5    95
## Louisiana            3806   3545        2.8    68.76   13.2    42.2    12
## Maine                1058   3694        0.7    70.39    2.7    54.7   161
## Maryland             4122   5299        0.9    70.22    8.5    52.3   101
## Massachusetts        5814   4755        1.1    71.83    3.3    58.5   103
## Michigan             9111   4751        0.9    70.63   11.1    52.8   125
## Minnesota            3921   4675        0.6    72.96    2.3    57.6   160
## Mississippi          2341   3098        2.4    68.09   12.5    41.0    50
## Missouri             4767   4254        0.8    70.69    9.3    48.8   108
## Montana               746   4347        0.6    70.56    5.0    59.2   155
## Nebraska             1544   4508        0.6    72.60    2.9    59.3   139
## Nevada                590   5149        0.5    69.03   11.5    65.2   188
## New Hampshire         812   4281        0.7    71.23    3.3    57.6   174
## New Jersey           7333   5237        1.1    70.93    5.2    52.5   115
## New Mexico           1144   3601        2.2    70.32    9.7    55.2   120
## New York            18076   4903        1.4    70.55   10.9    52.7    82
## North Carolina       5441   3875        1.8    69.21   11.1    38.5    80
## North Dakota          637   5087        0.8    72.78    1.4    50.3   186
## Ohio                10735   4561        0.8    70.82    7.4    53.2   124
## Oklahoma             2715   3983        1.1    71.42    6.4    51.6    82
## Oregon               2284   4660        0.6    72.13    4.2    60.0    44
## Pennsylvania        11860   4449        1.0    70.43    6.1    50.2   126
## Rhode Island          931   4558        1.3    71.90    2.4    46.4   127
## South Carolina       2816   3635        2.3    67.96   11.6    37.8    65
## South Dakota          681   4167        0.5    72.08    1.7    53.3   172
## Tennessee            4173   3821        1.7    70.11   11.0    41.8    70
## Texas               12237   4188        2.2    70.90   12.2    47.4    35
## Utah                 1203   4022        0.6    72.90    4.5    67.3   137
## Vermont               472   3907        0.6    71.64    5.5    57.1   168
## Virginia             4981   4701        1.4    70.08    9.5    47.8    85
## Washington           3559   4864        0.6    71.72    4.3    63.5    32
## West Virginia        1799   3617        1.4    69.48    6.7    41.6   100
## Wisconsin            4589   4468        0.7    72.48    3.0    54.5   149
## Wyoming               376   4566        0.6    70.29    6.9    62.9   173
##                   Area
## Alabama          50708
## Alaska          566432
## Arizona         113417
## Arkansas         51945
## California      156361
## Colorado        103766
## Connecticut       4862
## Delaware         1982
## Florida         54090
## Georgia         58073
## Hawaii           6425
## Idaho           82677
## Illinois        55748
## Indiana         36097
## Iowa            55941
## Kansas          81787
## Kentucky        39650
## Louisiana       44930
## Maine           30920
## Maryland         9891
## Massachusetts    7826
## Michigan        56817
## Minnesota       79289
## Mississippi     47296
## Missouri        68995
## Montana        145587
## Nebraska        76483
## Nevada         109889
## New Hampshire    9027
## New Jersey       7521
## New Mexico     121412
## New York        47831
## North Carolina  48798
## North Dakota    69273
## Ohio            40975
## Oklahoma        68782
## Oregon          96184
## Pennsylvania    44966
## Rhode Island     1049
## South Carolina  30225
## South Dakota    75955
## Tennessee       41328
## Texas          262134
## Utah            82096
## Vermont          9267
## Virginia        39780
## Washington      66570
## West Virginia   24070
## Wisconsin       54464
## Wyoming         97203
```

```
## 用 readr 包的函数读取，并显示读取的内容;
library(readr)
read_table("./data/Table0.txt", col_names = FALSE)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   X1 = col_character(),
##   X2 = col_double(),
##   X3 = col_double(),
##   X4 = col_double(),
##   X5 = col_character()
## )
```

```
## # A tibble: 7 x 5
##   X1          X2    X3    X4 X5
##   <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
```

```
read_table("./data/Table1.txt", col_names = TRUE)
```

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

```
## # 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
```

```
read_table("./data/Table2.txt", col_names = TRUE, skip = 2)
```

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

```
## # 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/
```

```
read_table("./data/Table3.txt", col_names = TRUE, skip = 2, na = c("--", "*", "**", "NA
```

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

```
## # 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
```

```
read_table("./data/Table4.txt", col_names = TRUE, na = c("--", "*", "**", "NA"))
```

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

```
## # 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
```

```
read_delim("./data/Table5.txt", delim = ";",col_names = TRUE, na = c("--", "*", "**", "
```

```
## Warning: One or more parsing issues, see `problems()` for details
```

```
## Rows: 7 Columns: 5
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (2): Name, Sex
## dbl (2): Age, Weight
##
## 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: 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
```

```
read_table("./data/Table6.txt", col_names = TRUE, skip = 2, n_max = 7, comment = "@")
```

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

```
## # 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
```

```
read_csv("./data/states1.csv")
```

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

```
## # A tibble: 50 x 9
##    ...1        Population Income Illiteracy Life E~1 Murder HS Gr~2 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 Arkansas         2110   3378        1.9     70.7   10.1    39.9    65  51945
##   5 California      21198   5114        1.1     71.7   10.3    62.6    20 156361
##   6 Colorado         2541   4884        0.7     72.1    6.8    63.9   166 103766
##   7 Connecticut      3100   5348        1.1     72.5    3.1    56       139   4862
##   8 Delaware          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
## # ... with 40 more rows, and abbreviated variable names 1: `Life Exp`,
## #   2: `HS Grad`
```

```
read_csv2("./data/states2.csv")
```

```
## i Using "',"' as decimal and "'.'" as grouping mark. Use `read_delim()` for more con
## New names: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 E~1 Murder HS Gr~2 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 Arkansas          2110   3378        1.9     70.7   10.1    39.9    65  51945
##  5 California       21198   5114        1.1     71.7   10.3    62.6    20 156361
##  6 Colorado          2541   4884        0.7     72.1    6.8    63.9   166 103766
##  7 Connecticut       3100   5348        1.1     72.5    3.1    56     139   4862
##  8 Delaware           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
## # ... with 40 more rows, and abbreviated variable names 1: `Life Exp`,
## #   2: `HS Grad`
```