talk03 练习与作业

目录

练习和作业说明 1
talk03 内容回顾 1
练习与作业 1，`data.frame` 2
练习与作业 2，`tibble` 23
练习与作业 3：IO 34

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。

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

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；
row_names<-c('Alex','Lilly','Mark','Oliver','Martha','Lucas','Caroline')
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)
df_1<-data.frame(Age=Age,Height=Height,Weight=Weight)
rownames(df_1)<-row_names
df_1
```

```
##           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
```

```
## 再插入第四列
Sex<-c('F','F','M','M','F','M','F')
df_1<-cbind(df_1,Sex)
## 显示最终结果
df_1
```

```
##           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
row_names<-c('Alex','Lilly','Mark','Oliver','Martha','Lucas','Caroline')
Working<-c('Yes','No','No','Yes','Yes','No','Yes')
df_2<-data.frame(Working=Working)
row.names(df_2)=row_names
## 显示结果
df_2
```

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

```
## 显示 Working 列的性质
cat('Working 这一列的性质为: ',class(df_2$Working))
```

```
## Working这一列的性质为:  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
```

```
str(state.center)
```

```
## List of 2
##  $ x: num [1:50] -86.8 -127.2 -111.6 -92.3 -119.8 ...
##  $ y: num [1:50] 32.6 49.2 34.2 34.7 36.5 ...
```

```
(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；
matrix_1<-matrix(1:250, 50, 5)
row_names=NULL
for (i in 1:50){
  row_names<-c(row_names,paste('row_',as.character(i),sep = ''))
}
row.names(matrix_1)<-row_names
matrix_1
```

```
##        [,1] [,2] [,3] [,4] [,5]
## row_1     1   51  101  151  201
## row_2     2   52  102  152  202
## row_3     3   53  103  153  203
## row_4     4   54  104  154  204
## row_5     5   55  105  155  205
## row_6     6   56  106  156  206
## row_7     7   57  107  157  207
## row_8     8   58  108  158  208
## row_9     9   59  109  159  209
## row_10   10   60  110  160  210
## row_11   11   61  111  161  211
## row_12   12   62  112  162  212
## row_13   13   63  113  163  213
## row_14   14   64  114  164  214
## row_15   15   65  115  165  215
## row_16   16   66  116  166  216
## row_17   17   67  117  167  217
## row_18   18   68  118  168  218
## row_19   19   69  119  169  219
## row_20   20   70  120  170  220
## row_21   21   71  121  171  221
## row_22   22   72  122  172  222
## row_23   23   73  123  173  223
## row_24   24   74  124  174  224
## row_25   25   75  125  175  225
## row_26   26   76  126  176  226
## row_27   27   77  127  177  227
## row_28   28   78  128  178  228
## row_29   29   79  129  179  229
## row_30   30   80  130  180  230
## row_31   31   81  131  181  231
## row_32   32   82  132  182  232
```

```
## row_33   33   83  133  183  233
## row_34   34   84  134  184  234
## row_35   35   85  135  185  235
## row_36   36   86  136  186  236
## row_37   37   87  137  187  237
## row_38   38   88  138  188  238
## row_39   39   89  139  189  239
## row_40   40   90  140  190  240
## row_41   41   91  141  191  241
## row_42   42   92  142  192  242
## row_43   43   93  143  193  243
## row_44   44   94  144  194  244
## row_45   45   95  145  195  245
## row_46   46   96  146  196  246
## row_47   47   97  147  197  247
## row_48   48   98  148  198  248
## row_49   49   99  149  199  249
## row_50   50  100  150  200  250
```

- **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"
```

```
if(class(VADeaths)[1]!='data.frame')
{
  cat('VADeaths 的类型不是 data.frame, 现进行转换')
  new_VADeaths<-as.data.frame(VADeaths)
}else
{
  new_VADeaths<-VADeaths
}
```

```
## VADeaths 的类型不是data.frame, 现进行转换
```

```
new_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
```

```
Total<-apply(new_VADeaths,1,sum)
cat('\n行和为: ',Total)
```

```
##
## 行和为:  44.2 67.7 103.5 161.6 241.4
```

```
new_VADeaths<-cbind(Total,new_VADeaths)
new_VADeaths
```

```
##       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；
cat('原始 swiss 如下\n')
```

```
## 原始swiss如下
```

```
swiss
```

```
##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
## Porrentruy        76.1        35.3           9         7    90.57
## Broye             83.8        70.2          16         7    92.85
## Glane             92.4        67.8          14         8    97.16
## Gruyere           82.4        53.3          12         7    97.67
## Sarine            82.9        45.2          16        13    91.38
## Veveyse           87.1        64.5          14         6    98.61
## Aigle             64.1        62.0          21        12     8.52
## Aubonne           66.9        67.5          14         7     2.27
```

```
## Avenches          68.9        60.7          19          12      4.43
## Cossonay          61.7        69.3          22           5      2.82
## Echallens         68.3        72.6          18           2     24.20
## Grandson          71.7        34.0          17           8      3.30
## Lausanne          55.7        19.4          26          28     12.11
## La Vallee         54.3        15.2          31          20      2.15
## Lavaux            65.1        73.0          19           9      2.84
## Morges            65.5        59.8          22          10      5.23
## Moudon            65.0        55.1          14           3      4.52
## Nyone             56.6        50.9          22          12     15.14
## Orbe              57.4        54.1          20           6      4.20
## Oron              72.5        71.2          12           1      2.40
## Payerne           74.2        58.1          14           8      5.23
## Paysd'enhaut      72.0        63.5           6           3      2.56
## Rolle             60.5        60.8          16          10      7.72
## Vevey             58.3        26.8          25          19     18.46
## Yverdon           65.4        49.5          15           8      6.10
## Conthey           75.5        85.9           3           2     99.71
## Entremont         69.3        84.9           7           6     99.68
## Herens            77.3        89.7           5           2    100.00
## Martigwy          70.5        78.2          12           6     98.96
## Monthey           79.4        64.9           7           3     98.22
## St Maurice        65.0        75.9           9           9     99.06
## Sierre            92.2        84.6           3           3     99.46
## Sion              79.3        63.1          13          13     96.83
## Boudry            70.4        38.4          26          12      5.62
## La Chauxdfnd      65.7         7.7          29          11     13.79
## Le Locle          72.7        16.7          22          13     11.22
## Neuchatel         64.4        17.6          35          32     16.92
## Val de Ruz        77.6        37.6          15           7      4.97
## ValdeTravers      67.6        18.7          25           7      8.65
## V. De Geneve      35.0         1.2          37          53     42.34
## Rive Droite       44.7        46.6          16          29     50.43
```

```
## Rive Gauche       42.8        27.7          22        29    58.33
##              Infant.Mortality
## Courtelary               22.2
## Delemont                 22.2
## Franches-Mnt             20.2
## Moutier                  20.3
## Neuveville               20.6
## Porrentruy               26.6
## Broye                    23.6
## Glane                    24.9
## Gruyere                  21.0
## Sarine                   24.4
## Veveyse                  24.5
## Aigle                    16.5
## Aubonne                  19.1
## Avenches                 22.7
## Cossonay                 18.7
## Echallens                21.2
## Grandson                 20.0
## Lausanne                 20.2
## La Vallee                10.8
## Lavaux                   20.0
## Morges                   18.0
## Moudon                   22.4
## Nyone                    16.7
## Orbe                     15.3
## Oron                     21.0
## Payerne                  23.8
## Paysd'enhaut             18.0
## Rolle                    16.3
## Vevey                    20.9
## Yverdon                  22.5
## Conthey                  15.1
```

```
## Entremont                19.8
## Herens                   18.3
## Martigwy                 19.4
## Monthey                  20.2
## St Maurice               17.8
## Sierre                   16.3
## Sion                     18.1
## Boudry                   20.3
## La Chauxdfnd             20.5
## Le Locle                 18.9
## Neuchatel                23.0
## Val de Ruz               20.0
## ValdeTravers             19.5
## V. De Geneve             18.0
## Rive Droite              18.2
## Rive Gauche              19.3
```

```
tar_row<-c(1,2,3,10,11,12,13)
tar_col<-c('Examination','Education','Infant.Mortality')
cat('取完子集后\n')
```

```
## 取完子集后
```

```
new_swiss<-swiss[tar_row,tar_col]
new_swiss
```

```
##              Examination Education Infant.Mortality
## Courtelary            15        12             22.2
## Delemont               6         9             22.2
## Franches-Mnt           5         5             20.2
## Sarine                16        13             24.4
## Veveyse               14         6             24.5
## Aigle                 21        12             16.5
## Aubonne               14         7             19.1
```

```
cat('按题意修改后: \n')
```

```
## 按题意修改后:
```

```
new_swiss['Sarine','Infant.Mortality']<-NA
new_swiss
```

```
##              Examination Education Infant.Mortality
## Courtelary            15        12             22.2
## Delemont               6         9             22.2
## Franches-Mnt           5         5             20.2
## Sarine                16        13               NA
## Veveyse               14         6             24.5
## Aigle                 21        12             16.5
## Aubonne               14         7             19.1
```

```
cat('增加一列 Mean 后')
```

```
## 增加一列Mean后
```

```
Mean<-apply(new_swiss, 1, mean)
new_swiss<-cbind(new_swiss,Mean)
new_swiss
```

```
##              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)
df_3<-data.frame(Id=Id,x=x,y=y)
df_3
```

```
##    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 函数的目的时返回一个 from to 的向量，along.with 是对于这个向量返回值的一个参数

官方文档提到的是 along.with take the length from the length of this argument. 也就是说使生成的向量的长度与 along.with 这个参数的长度一致，并且没有其他特别要求的情况下，以等步长的形式生成该向量

简单地说，along.with 表示产生的等间隔数列与向量具有相同的长度

举例如下

a 为 1，2，3，...，10，长度为 10，在指定 along.with=a 的时候，创建出来的 b c 的长度均为 10

而在不指定 along.with=a 的时候，b_ 的值为从 2 到 20 以 1 为步长的所有值，长度为 19，可见 along.with 的作用就是产生等间隔数列，且该向量与 along.with 的参数向量长度一致

```r
## 代码写这里，并运行；
a<-1:10
b<-seq(2,20,along.with=a)
b_<-seq(2,20)
c<-seq(0,100,along.with=a)
a
```

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

```
b
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

```
b_
```

```
##  [1]  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```
c
```

```
##  [1]   0.00000  11.11111  22.22222  33.33333  44.44444  55.55556  66.66667
##  [8]  77.77778  88.88889 100.00000
```

```
df_temp<-data.frame(a,b,c)
df_temp
```

```
##     a  b         c
## 1   1  2   0.00000
## 2   2  4  11.11111
## 3   3  6  22.22222
## 4   4  8  33.33333
## 5   5 10  44.44444
## 6   6 12  55.55556
## 7   7 14  66.66667
## 8   8 16  77.77778
## 9   9 18  88.88889
## 10 10 20 100.00000
```

- 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；
Id<-1:4
Age<-c(14,12,15,10)
Sex<-c('F','M','M','F')
Code<-c('a','b','c','d')
df1<-data.frame(Id=Id,Age=Age)
df1
```

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

```
df2<-data.frame(Id,Sex,Code)
df2
```

```
##   Id Sex Code
## 1  1   F    a
## 2  2   M    b
## 3  3   M    c
## 4  4   F    d
```

```
M<-merge(df1,df2)
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<-subset(M,select=-Code)
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);
```

```
## 代码写这里，并运行；
df3 <- data.frame( data = 1:10, group = c("A","B") )
df3
```

```
##    data group
## 1     1     A
## 2     2     B
## 3     3     A
## 4     4     B
## 5     5     A
## 6     6     B
## 7     7     A
## 8     8     B
## 9     9     A
## 10   10     B
```

```
cbind(df3, newcol = 1);
```

```
##    data group newcol
## 1     1     A      1
## 2     2     B      1
```

```
## 3     3     A      1
## 4     4     B      1
## 5     5     A      1
## 6     6     B      1
## 7     7     A      1
## 8     8     B      1
## 9     9     A      1
## 10   10     B      1
```

```
cbind(df3, newcol = 1:10);
```

```
##    data group newcol
## 1     1     A      1
## 2     2     B      2
## 3     3     A      3
## 4     4     B      4
## 5     5     A      5
## 6     6     B      6
## 7     7     A      7
## 8     8     B      8
## 9     9     A      9
## 10   10     B     10
```

```
cbind(df3, newcol = 1:2);
```

```
##    data group newcol
## 1     1     A      1
## 2     2     B      2
## 3     3     A      1
## 4     4     B      2
## 5     5     A      1
## 6     6     B      2
## 7     7     A      1
## 8     8     B      2
```

```
## 9     9     A       1
## 10    10    B       2
```

```
# cbind(df3, newcol = 1:3);
```

答：通过运行代码，可以发现

2. 增加一列，其长度是 1；3. 增加一列，其长度是 10；4. 增加一列，其长度是 2 这三个是可以的，因为其长度正好是行数可以整除的数：10/1=10；10/10=1；10/2=5，可以完成并行计算或者说循环补齐

但 5. 增加一列，其长度是 3 这一条不行，出现报错

Error in data.frame(…, check.names = FALSE) : 参数值意味着不同的行数: 10, 3

因为其长度不是行数可以整除的数：10/3=3……1，无法正常完成循环计算、并行计算

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");
}
```

```
## 载入需要的程辑包：lubridate
```

```
##
## 载入程辑包：'lubridate'
```

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

```
library(lubridate);

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

```
## 载入需要的程辑包: 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 2021-09-23 00:17:14 2021-10-10     1 0.261  p
##  2 2021-09-22 13:06:18 2021-10-03     2 0.751  i
##  3 2021-09-23 02:37:58 2021-10-17     3 0.0272 d
##  4 2021-09-23 09:29:19 2021-10-11     4 0.0170 m
##  5 2021-09-23 03:43:28 2021-09-29     5 0.989  i
##  6 2021-09-22 23:07:31 2021-09-27     6 0.528  u
##  7 2021-09-22 12:44:52 2021-09-23     7 0.0592 x
##  8 2021-09-22 13:11:51 2021-10-19     8 0.905  u
```

```
##  9 2021-09-22 17:35:51 2021-10-12     9 0.151  q
## 10 2021-09-23 02:25:22 2021-10-16    10 0.355  o
## # ... 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>
- 还有 int dbl chr lgl fctr
- 以上分别表示：日期 + 时间；日期；integer；double；character；逻辑数据类型；代表因子类型

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)
)
str(df)
```

```
## tibble [5 x 2] (S3: tbl_df/tbl/data.frame)
##  $ x: num [1:5] 0.434 0.684 0.196 0.168 0.308
##  $ y: num [1:5] 0.176 -0.598 0.712 0.486 -0.653
```

```
df
```

```
## # A tibble: 5 x 2
##       x      y
##   <dbl>  <dbl>
## 1 0.434  0.176
## 2 0.684 -0.598
## 3 0.196  0.712
## 4 0.168  0.486
## 5 0.308 -0.653
```

```
tb1<-as_tibble(df['x'])
cat('tb1 的数据类型为 tibble T/F',is_tibble(tb1),'\n')
```

```
## tb1的数据类型为tibble T/F TRUE
```

```
tb1
```

```
## # A tibble: 5 x 1
##       x
##   <dbl>
## 1 0.434
## 2 0.684
## 3 0.196
## 4 0.168
## 5 0.308
```

```
tb2<-df[['y']]
# 通过查找，发现也可以用先转换为 matrix 在转换为 vector 的方法
# 代码如下
```

```
# tb2<-as.matrix(tb2[,'y'])
# tb2<-as.vector(tb2)
cat('tb2 的数据类型为 vector T/F',is.vector(tb2),'\n')
```

```
## tb2的数据类型为vector T/F TRUE
```

```
tb2
```

```
## [1]  0.1762601 -0.5983850  0.7115153  0.4858815 -0.6532078
```

- 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(30:50,3),
            salary=sample(3000:5000,3))
tb
```

```
## # A tibble: 3 x 3
##   name    age salary
##   <chr> <int>  <dbl>
## 1 C        39   3013
## 2 E        44   4593
## 3 K        45   4945
```

- ** 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 tibble 的 recycling 仅限于长度为 1 或等长；而 data.frame 则为整除即可，

所以对于 `data.frame(a = 1:6, b = LETTERS[1:2])` 是可以运行的，因为 6/2=3，可以整除，然后循环补齐即可

而对于 `tibble(a = 1:6, b = LETTERS[1:2]);` 是不行的，因为 b 的长度为 2，相当于要要循环的长度不为 1，而是 2，同时 a 的长度是 6，二者也不等长，所以是不行的

- **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 函数的用法如下：

attach(what, pos = 2L, name = deparse(substitute(what), backtick=FALSE),

warn.conflicts = TRUE)

CSDN 上找到的说法如下：

1.attach() 是对 what 添加路径索引，避免重复输入 what 名称，参数解释如下：

what：数据框或列表；

pos=2L：添加的路径存储的位置，一般默认即可。在对多个数据添加索引时，此位置会变成 3L,4L,5L…detach() 撤销索引路径时，会撤销对应位置的索引储存，具体例子见后；

name：不懂，遇见需要的情况再补充；

backtick=FALSE：反引号，经过测试，该参数固定为 FALSE 不可调，再调用索引时会用到；

warn.conflicts：是否打印警告。

2.**detach()** 是撤销 attach() 建立的路径索引，往往二者配套使用。

简单地说，按个人理解，attach() 就是使括号内的参数添加到环境变量，使其内的内容可以直接被调用，而不用通过 $ 来引用；detach 就可以认为是 attach() 的逆操作，也就是说将该参数从环境变量中删除，使其内的参数不能直接被调用，需要用 $ 才能调用

举例如下，可以发现，如果不用 attach，就必须用 iris$Sepal.Width，而如果用 attach，将会弹出 iris 的列名，并且直接用列名操作即可，不需要用 iris$

```
head(iris,n=5);
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
```

```
head(iris$Sepal.Width,n=5)
```

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

```
attach(iris)
  head(Sepal.Width,n=5)
```

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

```
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;
if (class(airquality)[1]=='tibble'){
  cat('airquality 的类型是 tibble，不用转换')
}else
{
  cat('airquality 的类型不是 tibble，需要转换\n')
  tibble_airquality=as_tibble(airquality)
}
```

```
## airquality 的类型不是tibble，需要转换
```

```
class(airquality)
```

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

```
str(tibble_airquality)
```

```
## tibble [153 x 6] (S3: tbl_df/tbl/data.frame)
##  $ Ozone  : int [1:153] 41 36 12 18 NA 28 23 19 8 NA ...
##  $ Solar.R: int [1:153] 190 118 149 313 NA NA 299 99 19 194 ...
##  $ Wind   : num [1:153] 7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
##  $ Temp   : int [1:153] 67 72 74 62 56 66 65 59 61 69 ...
##  $ Month  : int [1:153] 5 5 5 5 5 5 5 5 5 5 ...
##  $ Day    : int [1:153] 1 2 3 4 5 6 7 8 9 10 ...
```

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

答：官方文档 Help 给出来的说法是：Converting vectors to data frames, and vice versa. 更详细的说法是：enframe() converts named atomic vectors or lists to one- or two-column data frames. For a list, the result will be a nested tibble with a column of type list. For unnamed vectors, the natural sequence is used as name column. 简单地说，就是将一个 vector 或者 list 转变成一列或两列的 data frames

举例如下

```
l<-list(a=sample(1:10,3),b=sample(1:10,3))
l
```

```
## $a
## [1] 8 7 2
##
## $b
## [1] 2 4 7
```

```
l_tb<-tibble::enframe(l)
l_tb
```

```
## # A tibble: 2 x 2
##   name  value
##   <chr> <list>
## 1 a     <int [3]>
## 2 b     <int [3]>
```

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

答：tibble 对 data.frame 做了重新的设定：

1. 不关心输入类型，可存储任意类型，包括 list
2. 支持任意的列名，会自动添加列名
3. 会懒加载参数，并且按顺序运行，或者说 tibble evaluates columns sequentially
4. 对于 subset 操作，tibble 不会造成困扰
5. tibble 可以进行可控的数据类型转换

   个人认为也有缺点，比如 tibble 的 recycling 仅限于长度为 1 或登场，而 data.frame 则为整除即可

   另外，data.frame 可以实现部分匹配，而 tibble 不可以

   劣势就不做展示了

```
## 代码写这里，并运行；
cat('对于前两条优势，可以用之前题目中出现过的代码展示')
```

```
## 对于前两条优势，可以用之前题目中出现过的代码展示
```

```
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 2021-09-22 21:43:27 2021-10-20     1 0.285 p
##  2 2021-09-22 10:54:21 2021-10-17     2 0.404 v
```

```
##  3 2021-09-22 14:50:16 2021-10-09     3 0.860 t
##  4 2021-09-22 15:36:54 2021-10-13     4 0.611 y
##  5 2021-09-22 10:55:47 2021-10-10     5 0.694 q
##  6 2021-09-23 00:09:24 2021-10-05     6 0.877 p
##  7 2021-09-22 13:09:14 2021-10-07     7 0.404 o
##  8 2021-09-22 12:12:13 2021-10-09     8 0.709 u
##  9 2021-09-22 17:15:27 2021-10-16     9 0.875 q
## 10 2021-09-22 22:58:14 2021-10-01    10 0.803 r
## # ... with 990 more rows
```

```
cat('对于第三条，可以用 PDF 的代码展示')
```

```
## 对于第三条，可以用PDF的代码展示
```

```
tibble(x=1:5,y=x^2)
```

```
## # A tibble: 5 x 2
##       x     y
##   <int> <dbl>
## 1     1     1
## 2     2     4
## 3     3     9
## 4     4    16
## 5     5    25
```

```
cat('对于第四条，可以用 PDF 的代码展示')
```

```
## 对于第四条，可以用PDF的代码展示
```

```
df1 <- data.frame(x = 1:3, y = 3:1);
class(df1[, 1:2]);
```

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

```
class(df1[, 1]);  # 会得到一个 vector...
```

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

```
df2 <- tibble(x = 1:3, y = 3:1);
class(df2[, 1]);  # 永远得到的都是 tibble
```

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

```
cat('对于第五条，可以用 PDF 的代码展示')
```

```
## 对于第五条，可以用PDF的代码展示
```

```
class(df2[[1]]); ## 取一列，转换为 vector
```

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

```
class(df2$x); ## 用 [[]] 或 $ 都可以哦
```

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

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.csv('data/Table0.txt')
```

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

```
read.csv('data/Table1.txt')
```

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

```
read.csv('data/Table2.txt')
```

```
##                            Table.2..Name Age Height
## 1   Name      Age   Height  Weight   Sex  NA     NA
## 2  /Alex/\t   25      177      57    /F/  NA     NA
```

```
## 3  /Lilly/    31      163      69\t  /F/  NA     NA
## 4  /Mark/\t   23      190      83    /M/  NA     NA
## 5  /Oliver/   52      179      75\t  /M/  NA     NA
## 6  /Martha/   76      163      70    /F/  NA     NA
## 7 /Lucas/\t   49      183      83    /M/  NA     NA
## 8  /Caroline/ 26      164      53    /F/  NA     NA
##   Weight.and.Sex.from.7.people
## 1                           NA
## 2                           NA
## 3                           NA
## 4                           NA
## 5                           NA
## 6                           NA
## 7                           NA
## 8                           NA
```

```
read.csv('data/Table3.txt')
```

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

```
read.csv('data/Table4.txt')
```

```
##                                   Name....Age...Height..Weight...Sex
## Alex\t 25      1                                    77     57     F
## Lilly \t 31        NA     69\t F
## Mark\t --\t 1                                       90     83     M
```

```
## Oliver\t 52      1                                    79     75\t M
## Martha   76         *     70     F
## Lucas\t 49      1                                  83     **     M
## Caroline 26      1                                  64     53     F
```

```
read.csv('data/Table5.txt')
```

```
##                   Name.Age.Height.Weight.Sex
## Alex;25;1                             77;57;F
## Lilly;31;NA;69;F
## Mark;--;1                             90;83;M
## Oliver;52;1                           79;75;M
## Martha;76;;70;F
## Lucas;49;1                            83;**;M
## Caroline;26;1                         64;53;F
```

```
read.csv('data/Table6.txt')
```

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

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

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

```
## 82             Oliver\t 52      179      75\t M  NA     NA
## 83           Martha   76      163      70     F  NA     NA
## 84            Lucas\t 49      183      83     M  NA     NA
## 85           Caroline 26      164      53     F  NA     NA
## 86             Alex\t 25      177      57     F  NA     NA
## 87             Lilly \t 31      163      69\t F  NA     NA
## 88             Mark\t 23      190      83     M  NA     NA
## 89             Oliver\t 52      179      75\t M  NA     NA
## 90           Martha   76      163      70     F  NA     NA
## 91            Lucas\t 49      183      83     M  NA     NA
## 92           Caroline 26      164      53     F  NA     NA
## 93             Alex\t 25      177      57     F  NA     NA
## 94             Lilly \t 31      163      69\t F  NA     NA
## 95             Mark\t 23      190      83     M  NA     NA
## 96             Oliver\t 52      179      75\t M  NA     NA
## 97           Martha   76      163      70     F  NA     NA
## 98            Lucas\t 49      183      83     M  NA     NA
## 99           Caroline 26      164      53     F  NA     NA
## 100            Alex\t 25      177      57     F  NA     NA
## 101            Lilly \t 31      163      69\t F  NA     NA
## 102            Mark\t 23      190      83     M  NA     NA
## 103            Oliver\t 52      179      75\t M  NA     NA
## 104          Martha   76      163      70     F  NA     NA
## 105           Lucas\t 49      183      83     M  NA     NA
## 106          Caroline 26      164      53     F  NA     NA
##     Weight.and.Sex.from.7.people
## 1                              NA
## 2                              NA
## 3                              NA
## 4                              NA
## 5                              NA
## 6                              NA
## 7                              NA
```

```
## 8                    NA
## 9                    NA
## 10                   NA
## 11                   NA
## 12                   NA
## 13                   NA
## 14                   NA
## 15                   NA
## 16                   NA
## 17                   NA
## 18                   NA
## 19                   NA
## 20                   NA
## 21                   NA
## 22                   NA
## 23                   NA
## 24                   NA
## 25                   NA
## 26                   NA
## 27                   NA
## 28                   NA
## 29                   NA
## 30                   NA
## 31                   NA
## 32                   NA
## 33                   NA
## 34                   NA
## 35                   NA
## 36                   NA
## 37                   NA
## 38                   NA
## 39                   NA
## 40                   NA
```

```
## 41                    NA
## 42                    NA
## 43                    NA
## 44                    NA
## 45                    NA
## 46                    NA
## 47                    NA
## 48                    NA
## 49                    NA
## 50                    NA
## 51                    NA
## 52                    NA
## 53                    NA
## 54                    NA
## 55                    NA
## 56                    NA
## 57                    NA
## 58                    NA
## 59                    NA
## 60                    NA
## 61                    NA
## 62                    NA
## 63                    NA
## 64                    NA
## 65                    NA
## 66                    NA
## 67                    NA
## 68                    NA
## 69                    NA
## 70                    NA
## 71                    NA
## 72                    NA
## 73                    NA
```

```
## 74                    NA
## 75                    NA
## 76                    NA
## 77                    NA
## 78                    NA
## 79                    NA
## 80                    NA
## 81                    NA
## 82                    NA
## 83                    NA
## 84                    NA
## 85                    NA
## 86                    NA
## 87                    NA
## 88                    NA
## 89                    NA
## 90                    NA
## 91                    NA
## 92                    NA
## 93                    NA
## 94                    NA
## 95                    NA
## 96                    NA
## 97                    NA
## 98                    NA
## 99                    NA
## 100                   NA
## 101                   NA
## 102                   NA
## 103                   NA
## 104                   NA
## 105                   NA
## 106                   NA
```

```
read.csv('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',header = F)
```

```
##                                                                        V1    V2
## 1  ;Population;Income;Illiteracy;Life Exp;Murder;HS Grad;Frost;Area
## 2                                                    Alabama;3615;3624;2  1;69
## 3                                                      Alaska;365;6315;1  5;69
## 4                                                    Arizona;2212;4530;1  8;70
## 5                                                   Arkansas;2110;3378;1  9;70
## 6                                                 California;21198;5114;1  1;71
## 7                                                    Colorado;2541;4884;0  7;72
## 8                                                 Connecticut;3100;5348;1  1;72
## 9                                                     Delaware;579;4809;0  9;70
## 10                                                    Florida;8277;4815;1  3;70
## 11                                                  Georgia;4931;4091;2;68 54;13
## 12                                                     Hawaii;868;4963;1  9;73
## 13                                                      Idaho;813;4119;0  6;71
## 14                                                  Illinois;11197;5107;0  9;70
## 15                                                    Indiana;5313;4458;0  7;70
## 16                                                       Iowa;2861;4628;0  5;72
## 17                                                    Kansas;2280;4669;0  6;72
## 18                                                  Kentucky;3387;3712;1  6;70
## 19                                                 Louisiana;3806;3545;2  8;68
## 20                                                      Maine;1058;3694;0  7;70
## 21                                                   Maryland;4122;5299;0  9;70
## 22                                              Massachusetts;5814;4755;1  1;71
## 23                                                   Michigan;9111;4751;0  9;70
## 24                                                  Minnesota;3921;4675;0  6;72
## 25                                                Mississippi;2341;3098;2  4;68
```

```
## 26                                                Missouri;4767;4254;0  8;70
## 27                                                  Montana;746;4347;0  6;70
## 28                                                Nebraska;1544;4508;0  6;72
## 29                                                   Nevada;590;5149;0  5;69
## 30                                           New Hampshire;812;4281;0  7;71
## 31                                             New Jersey;7333;5237;1  1;70
## 32                                             New Mexico;1144;3601;2  2;70
## 33                                              New York;18076;4903;1  4;70
## 34                                         North Carolina;5441;3875;1  8;69
## 35                                            North Dakota;637;5087;0  8;72
## 36                                                 Ohio;10735;4561;0  8;70
## 37                                               Oklahoma;2715;3983;1  1;71
## 38                                                  Oregon;2284;4660;0  6;72
## 39                                        Pennsylvania;11860;4449;1;70  43;6
## 40                                             Rhode Island;931;4558;1  3;71
## 41                                           South Carolina;2816;3635;2  3;67
## 42                                            South Dakota;681;4167;0  5;72
## 43                                               Tennessee;4173;3821;1  7;70
## 44                                                  Texas;12237;4188;2  2;70
## 45                                                    Utah;1203;4022;0  6;72
## 46                                                 Vermont;472;3907;0  6;71
## 47                                                Virginia;4981;4701;1  4;70
## 48                                              Washington;3559;4864;0  6;71
## 49                                           West Virginia;1799;3617;1  4;69
## 50                                               Wisconsin;4589;4468;0  7;72
## 51                                                  Wyoming;376;4566;0  6;70
##          V3             V4           V5
## 1
## 2     05;15           1;41   3;20;50708
## 3     31;11           3;66 7;152;566432
## 4      55;7           8;58  1;15;113417
## 5     66;10           1;39   9;65;51945
## 6     71;10           3;62  6;20;156361
```

```
## 7      06;6           8;63 9;166;103766
## 8      48;3  1;56;139;4862
## 9      06;6           2;54   6;103;1982
## 10    66;10           7;52   6;11;54090
## 11     9;40     6;60;58073
## 12      6;6           2;61     9;0;6425
## 13     87;5           3;59  5;126;82677
## 14    14;10           3;52  6;127;55748
## 15     88;7           1;52  9;122;36097
## 16     56;2 3;59;140;55941
## 17     58;4           5;59  9;114;81787
## 18     1;10           6;38   5;95;39650
## 19    76;13           2;42   2;12;44930
## 20     39;2           7;54  7;161;30920
## 21     22;8           5;52   3;101;9891
## 22     83;3           3;58   5;103;7826
## 23    63;11           1;52  8;125;56817
## 24     96;2           3;57  6;160;79289
## 25    09;12  5;41;50;47296
## 26     69;9           3;48  8;108;68995
## 27  56;5;59   2;155;145587
## 28      6;2           9;59  3;139;76483
## 29    03;11           5;65 2;188;109889
## 30     23;3           3;57   6;174;9027
## 31     93;5           2;52   5;115;7521
## 32     32;9           7;55 2;120;121412
## 33    55;10           9;52   7;82;47831
## 34    21;11           1;38   5;80;48798
## 35     78;1           4;50  3;186;69273
## 36     82;7           4;53  2;124;40975
## 37     42;6           4;51   6;82;68782
## 38     13;4  2;60;44;96184
## 39     1;50    2;126;44966
```

```
## 40      9;2           4;46   4;127;1049
## 41    96;11           6;37   8;65;30225
## 42     08;1           7;53  3;172;75955
## 43 11;11;41     8;70;41328
## 44     9;12           2;47  4;35;262134
## 45      9;4           5;67  3;137;82096
## 46     64;5           5;57   1;168;9267
## 47     08;9           5;47   8;85;39780
## 48     72;4           3;63   5;32;66570
## 49     48;6           7;41  6;100;24070
## 50  48;3;54    5;149;54464
## 51     29;6           9;62  9;173;97203
```

```
## 用 readr 包的函数读取，并显示读取的内容;
library(readr)
read_csv('data/Table0.txt')
```

```
## Rows: 6 Columns: 1

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Alex     25      177      57     F

##
## 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: 6 x 1
##   `Alex\t 25      177      57     F`
##   <chr>
## 1 "Lilly \t 31      163      69\t F"
## 2 "Mark\t 23      190      83     M"
## 3 "Oliver\t 52      179      75\t M"
## 4 "Martha   76      163      70     F"
```

```
## 5 "Lucas\t 49      183      83     M"
## 6 "Caroline 26      164      53     F"
```

```
read_csv('data/Table1.txt')
```

```
## Rows: 7 Columns: 1
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Name    Age   Height  Weight   Sex
```

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

```
read_csv('data/Table2.txt')
```

```
## Rows: 0 Columns: 0
##
## 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: 0 x 0
```

```
read_csv('data/Table3.txt')
```

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

```
## Rows: 8 Columns: 4
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Table 2: Name
## lgl (3): Age, Height, Weight and Sex from 7 people
```

```
##
## 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: 8 x 4
##   `Table 2: Name`                     Age   Height `Weight and Sex from 7 peop~
##   <chr>                               <lgl> <lgl>  <lgl>
## 1 "Name    Age   Height  Weight   Sex" NA    NA     NA
## 2 "Alex\t 25      177      57     F"   NA    NA     NA
## 3 "Lilly \t 31       NA      69\t F"   NA    NA     NA
## 4 "Mark\t --\t 190      83     M"      NA    NA     NA
## 5 "Oliver\t 52      179      75\t M"   NA    NA     NA
## 6 "Martha   76        *      70     F" NA    NA     NA
## 7 "Lucas\t 49      183      **     M"  NA    NA     NA
## 8 "Caroline 26      164      53     F" NA    NA     NA
```

```
read_csv('data/Table4.txt')
```

```
## Rows: 7 Columns: 1
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Name    Age   Height  Weight   Sex
```

```
##
## 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.

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

## # A tibble: 7 x 1
##   `Name    Age   Height  Weight   Sex`
##   <chr>
## 1 "Alex\t 25      1,77     57     F"
## 2 "Lilly \t 31        NA     69\t F"
## 3 "Mark\t --\t 1,90     83     M"
## 4 "Oliver\t 52      1,79     75\t M"
## 5 "Martha   76         *     70     F"
## 6 "Lucas\t 49      1,83     **     M"
## 7 "Caroline 26      1,64     53     F"
```

```
read_csv('data/Table5.txt')
```

```
## Rows: 7 Columns: 1

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Name;Age;Height;Weight;Sex

##
## 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.

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

## # A tibble: 7 x 1
##   `Name;Age;Height;Weight;Sex`
##   <chr>
```

```
## 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;;70;F
## 6 Lucas;49;1,83;**;M
## 7 Caroline;26;1,64;53;F
```

```
read_csv('data/Table6.txt')
```

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

## Rows: 106 Columns: 4

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Table 2: Name
## lgl (3): Age, Height, Weight and Sex from 7 people

##
## 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: 106 x 4
##    `Table 2: Name`                                Age   Height `Weight and Sex f~
##    <chr>                                          <lgl> <lgl>  <lgl>
##  1 "Name    Age   Height  Weight   Sex"          NA    NA     NA
##  2 "Alex\t 25      177      57     F @Boss"      NA    NA     NA
##  3 "Lilly \t 31      163      69\t F @Secretary" NA    NA     NA
##  4 "Mark\t 23      190      83     M"            NA    NA     NA
##  5 "Oliver\t 52      179      75\t M"            NA    NA     NA
##  6 "Martha   76      163      70     F"          NA    NA     NA
##  7 "Lucas\t 49      183      83     M"          NA    NA     NA
##  8 "Caroline 26      164      53     F"          NA    NA     NA
```

```
##  9 "Alex\t 25      177      57     F"            NA    NA     NA
## 10 "Lilly \t 31      163      69\t F"            NA    NA     NA
## # ... with 96 more rows
```

```
read_csv('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
## # ... with 40 more rows
```

```
read_csv('data/states2.csv')
```

```
## Rows: 50 Columns: 1

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): ;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.

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

## # A tibble: 50 x 1
##    `;Population;Income;Illiteracy;Life Exp;Murder;HS Grad;Frost;Area`
##    <chr>
##  1 Alabama;3615;3624;2,1;69,05;15,1;41,3;20;50708
##  2 Alaska;365;6315;1,5;69,31;11,3;66,7;152;566432
##  3 Arizona;2212;4530;1,8;70,55;7,8;58,1;15;113417
##  4 Arkansas;2110;3378;1,9;70,66;10,1;39,9;65;51945
##  5 California;21198;5114;1,1;71,71;10,3;62,6;20;156361
##  6 Colorado;2541;4884;0,7;72,06;6,8;63,9;166;103766
##  7 Connecticut;3100;5348;1,1;72,48;3,1;56;139;4862
##  8 Delaware;579;4809;0,9;70,06;6,2;54,6;103;1982
##  9 Florida;8277;4815;1,3;70,66;10,7;52,6;11;54090
## 10 Georgia;4931;4091;2;68,54;13,9;40,6;60;58073
## # ... with 40 more rows
```