talk08 练习与作业

目录

0.1 练习和作业说明

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

完成后，用工具栏里的”Knit” 按键生成 PDF 文档；

将 PDF 文档改为：姓名-学号-talk08 作业.pdf，并提交到老师指定的平台/钉群。

0.2 talk08 内容回顾

- for loop
- apply functions
- dplyr 的本质是遍历
- map functions in purrr package
- 遍历与并行计算

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：loop 初步

0.4.1 loop 练习（部分内容来自 r-exercises.com 网站）

1. 写一个循环，计算从 1 到 7 的平方并打印 `print`；
2. 取 `iris` 的列名，计算每个列名的长度，并打印为下面的格式：`Sepal.Length (12)`；
3. 写一个 `while` 循环，每次用 `rnorm` 取一个随机数字并打印，直到取到的数字大于 1；
4. 写一个循环，计算 Fibonacci 序列的值超过 1 百万所需的循环数；注：Fibonacci 序列的规则为：`0, 1, 1, 2, 3, 5, 8, 13, 21 ...`；

```
## 代码写这里，并运行；
library(stringr)

# 1.
for (it in 1:7)
  print(it ^ 2)
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
## [1] 36
## [1] 49
```

```r
# 2.
for (name in colnames(iris))
  print(paste(name, " (", str_length(name), ")", sep = ""))
```

```
## [1] "Sepal.Length (12)"
## [1] "Sepal.Width (11)"
## [1] "Petal.Length (12)"
## [1] "Petal.Width (11)"
## [1] "Species (7)"
```

```r
# 3.
while((rnd <- rnorm(1)) <= 1)
  print(rnd)
```

```
## [1] -0.5601494
```

```r
# 4.
prev <- 0
curr <- 1
cnt <- 2
while(curr < 1000000) {
  cnt <- cnt + 1
  nxt <- prev + curr
  prev <- curr
  curr <- nxt
```

```
}
print(cnt)
```

```
## [1] 32
```

0.5 练习与作业 2：loop 进阶，系统和其它函数

0.5.1 生成一个数字 matrix，并做练习

生成一个 100 x 100 的数字 matrix：

1. 行、列平均，用 rowMeans, colMeans 函数；
2. 行、列平均，用 apply 函数
3. 行、列总和，用 rowSums, colSums 函数；
4. 行、列总和，用 apply 函数
5. 使用自定义函数，同时计算：
    - 行平均、总和、sd
    - 列平均、总和、sd

```
## 代码写这里，并运行；
library(tidyverse)
```

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

## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.0      v forcats 0.5.2
## v readr   2.1.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
```

```
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# matrix 生成
matrix <-
  matrix(rnorm(10000), nrow = 100, ncol = 100)

# 1.
rowMeans(matrix)
```

```
##   [1]  0.0325434275  0.0428719117  0.0832948684  0.0624518705 -0.0943402013
##   [6]  0.1555209184 -0.0377115681  0.0209667192  0.1277860923  0.0766498861
##  [11] -0.0121086369  0.1752341163  0.0847784266  0.0269995246  0.0077971529
##  [16] -0.0191459430 -0.1193012165 -0.0256118327 -0.1802727143  0.0139524373
##  [21] -0.1715961938 -0.0720340086 -0.0511286530  0.1042072310  0.0709004382
##  [26]  0.0065468047 -0.0496526056  0.0763555058 -0.0144651977  0.0585829300
##  [31] -0.0177833724 -0.0076587767  0.0139664242 -0.0626221748 -0.0257659948
##  [36] -0.0782485876 -0.0588414485 -0.0050146625 -0.0280322215 -0.0965361292
##  [41] -0.1221816293 -0.0235298592  0.2558185642  0.0564374185 -0.0934948371
##  [46] -0.0730675198 -0.0193487313  0.1078279973 -0.0131279310 -0.0864731981
##  [51] -0.0413601064 -0.0969080797 -0.0030954254 -0.1802055993  0.0400503853
##  [56]  0.0155379932 -0.0296375310  0.0003442909 -0.1433630956  0.1976535437
##  [61]  0.0553886425 -0.0572461176  0.1689597195  0.3403169079  0.0462944037
##  [66] -0.0204549852 -0.0474767516 -0.1351051396  0.0985459052  0.0972276165
##  [71] -0.0296143187 -0.0407379218  0.2228229982 -0.1375699953 -0.0245748933
##  [76] -0.0024298219  0.0552778810 -0.1116797336  0.0823208051 -0.1057197993
##  [81]  0.0229877570  0.0416062460 -0.0930873055 -0.0501070075 -0.1552814219
##  [86] -0.1352056887  0.0084436550 -0.0092553315  0.1634489383 -0.0977971667
##  [91]  0.0255158245 -0.1642152600  0.0559347917  0.0024776116 -0.0321006658
##  [96]  0.0782274595  0.0573389532  0.1354943664 -0.0659274347 -0.0167658746
```

```
colMeans(matrix)
```

```
##   [1] -2.037825e-01 -2.657688e-03 -5.802431e-02 -1.596598e-01 -9.690657e-02
```

```
##   [6]  1.608592e-01 -1.804986e-01  2.114791e-02  1.468540e-02 -2.715176e-02
##  [11]  3.467055e-02 -1.952165e-03  1.463964e-01 -9.876786e-02  4.957266e-03
##  [16] -1.347120e-02  5.636130e-02 -1.201412e-01  4.196866e-02 -1.206471e-01
##  [21]  1.672957e-01  8.257670e-02 -1.502577e-01 -1.109822e-01  1.571000e-01
##  [26] -2.436824e-01  1.536761e-02 -8.547891e-02  4.631827e-02 -1.859302e-01
##  [31]  1.758185e-02 -5.495701e-02 -1.593242e-06  8.270861e-02  1.254629e-01
##  [36]  1.350680e-01  1.303743e-01 -2.186567e-01 -1.480827e-02  1.233280e-01
##  [41]  6.329530e-03  5.548793e-02  2.607311e-02  9.381683e-02  6.295803e-02
##  [46] -1.550976e-02  1.473325e-02 -1.020233e-01 -3.830333e-02 -9.554149e-02
##  [51] -7.669489e-02  1.399763e-01 -3.801175e-03 -1.534343e-01  3.165933e-02
##  [56]  7.464777e-02  1.299174e-01 -5.289055e-02 -1.126933e-01  1.186857e-02
##  [61]  1.393459e-02  1.778689e-02 -1.470040e-01  1.237466e-01  6.290203e-02
##  [66] -2.843380e-02  1.228141e-01 -1.209915e-01  9.296355e-02  8.561570e-02
##  [71]  8.647106e-02  4.658601e-02 -1.125660e-01 -6.858778e-02  3.400140e-02
##  [76]  4.746721e-02 -1.952270e-02  7.223008e-02  1.402312e-01  1.307867e-03
##  [81] -4.980046e-02 -5.345778e-02 -8.439010e-02  7.183895e-03  4.271054e-02
##  [86]  5.627819e-02 -3.009682e-02 -6.346351e-02 -1.515026e-01 -7.417148e-02
##  [91]  5.679940e-02  1.024616e-01  2.170326e-01  2.736635e-01 -1.133030e-01
##  [96]  1.964512e-02 -6.910142e-02  6.120836e-02 -3.527931e-02  1.193073e-02
```

```
# 2.
matrix %>% apply(., 1, mean)
```

```
##   [1]  0.0325434275  0.0428719117  0.0832948684  0.0624518705 -0.0943402013
##   [6]  0.1555209184 -0.0377115681  0.0209667192  0.1277860923  0.0766498861
##  [11] -0.0121086369  0.1752341163  0.0847784266  0.0269995246  0.0077971529
##  [16] -0.0191459430 -0.1193012165 -0.0256118327 -0.1802727143  0.0139524373
##  [21] -0.1715961938 -0.0720340086 -0.0511286530  0.1042072310  0.0709004382
##  [26]  0.0065468047 -0.0496526056  0.0763555058 -0.0144651977  0.0585829300
##  [31] -0.0177833724 -0.0076587767  0.0139664242 -0.0626221748 -0.0257659948
##  [36] -0.0782485876 -0.0588414485 -0.0050146625 -0.0280322215 -0.0965361292
##  [41] -0.1221816293 -0.0235298592  0.2558185642  0.0564374185 -0.0934948371
##  [46] -0.0730675198 -0.0193487313  0.1078279973 -0.0131279310 -0.0864731981
##  [51] -0.0413601064 -0.0969080797 -0.0030954254 -0.1802055993  0.0400503853
```

```
##  [56]  0.0155379932 -0.0296375310  0.0003442909 -0.1433630956  0.1976535437
##  [61]  0.0553886425 -0.0572461176  0.1689597195  0.3403169079  0.0462944037
##  [66] -0.0204549852 -0.0474767516 -0.1351051396  0.0985459052  0.0972276165
##  [71] -0.0296143187 -0.0407379218  0.2228229982 -0.1375699953 -0.0245748933
##  [76] -0.0024298219  0.0552778810 -0.1116797336  0.0823208051 -0.1057197993
##  [81]  0.0229877570  0.0416062460 -0.0930873055 -0.0501070075 -0.1552814219
##  [86] -0.1352056887  0.0084436550 -0.0092553315  0.1634489383 -0.0977971667
##  [91]  0.0255158245 -0.1642152600  0.0559347917  0.0024776116 -0.0321006658
##  [96]  0.0782274595  0.0573389532  0.1354943664 -0.0659274347 -0.0167658746
```

```
matrix %>% apply(., 2, mean)
```

```
##   [1] -2.037825e-01 -2.657688e-03 -5.802431e-02 -1.596598e-01 -9.690657e-02
##   [6]  1.608592e-01 -1.804986e-01  2.114791e-02  1.468540e-02 -2.715176e-02
##  [11]  3.467055e-02 -1.952165e-03  1.463964e-01 -9.876786e-02  4.957266e-03
##  [16] -1.347120e-02  5.636130e-02 -1.201412e-01  4.196866e-02 -1.206471e-01
##  [21]  1.672957e-01  8.257670e-02 -1.502577e-01 -1.109822e-01  1.571000e-01
##  [26] -2.436824e-01  1.536761e-02 -8.547891e-02  4.631827e-02 -1.859302e-01
##  [31]  1.758185e-02 -5.495701e-02 -1.593242e-06  8.270861e-02  1.254629e-01
##  [36]  1.350680e-01  1.303743e-01 -2.186567e-01 -1.480827e-02  1.233280e-01
##  [41]  6.329530e-03  5.548793e-02  2.607311e-02  9.381683e-02  6.295803e-02
##  [46] -1.550976e-02  1.473325e-02 -1.020233e-01 -3.830333e-02 -9.554149e-02
##  [51] -7.669489e-02  1.399763e-01 -3.801175e-03 -1.534343e-01  3.165933e-02
##  [56]  7.464777e-02  1.299174e-01 -5.289055e-02 -1.126933e-01  1.186857e-02
##  [61]  1.393459e-02  1.778689e-02 -1.470040e-01  1.237466e-01  6.290203e-02
##  [66] -2.843380e-02  1.228141e-01 -1.209915e-01  9.296355e-02  8.561570e-02
##  [71]  8.647106e-02  4.658601e-02 -1.125660e-01 -6.858778e-02  3.400140e-02
##  [76]  4.746721e-02 -1.952270e-02  7.223008e-02  1.402312e-01  1.307867e-03
##  [81] -4.980046e-02 -5.345778e-02 -8.439010e-02  7.183895e-03  4.271054e-02
##  [86]  5.627819e-02 -3.009682e-02 -6.346351e-02 -1.515026e-01 -7.417148e-02
##  [91]  5.679940e-02  1.024616e-01  2.170326e-01  2.736635e-01 -1.133030e-01
##  [96]  1.964512e-02 -6.910142e-02  6.120836e-02 -3.527931e-02  1.193073e-02
```

```
# 3.
rowSums(matrix)
```

```
##   [1]   3.25434275   4.28719117   8.32948684   6.24518705  -9.43402013
##   [6]  15.55209184  -3.77115681   2.09667192  12.77860923   7.66498861
##  [11]  -1.21086369  17.52341163   8.47784266   2.69995246   0.77971529
##  [16]  -1.91459430 -11.93012165  -2.56118327 -18.02727143   1.39524373
##  [21] -17.15961938  -7.20340086  -5.11286530  10.42072310   7.09004382
##  [26]   0.65468047  -4.96526056   7.63555058  -1.44651977   5.85829300
##  [31]  -1.77833724  -0.76587767   1.39664242  -6.26221748  -2.57659948
##  [36]  -7.82485876  -5.88414485  -0.50146625  -2.80322215  -9.65361292
##  [41] -12.21816293  -2.35298592  25.58185642   5.64374185  -9.34948371
##  [46]  -7.30675198  -1.93487313  10.78279973  -1.31279310  -8.64731981
##  [51]  -4.13601064  -9.69080797  -0.30954254 -18.02055993   4.00503853
##  [56]   1.55379932  -2.96375310   0.03442909 -14.33630956  19.76535437
##  [61]   5.53886425  -5.72461176  16.89597195  34.03169079   4.62944037
##  [66]  -2.04549852  -4.74767516 -13.51051396   9.85459052   9.72276165
##  [71]  -2.96143187  -4.07379218  22.28229982 -13.75699953  -2.45748933
##  [76]  -0.24298219   5.52778810 -11.16797336   8.23208051 -10.57197993
##  [81]   2.29877570   4.16062460  -9.30873055  -5.01070075 -15.52814219
##  [86] -13.52056887   0.84436550  -0.92553315  16.34489383  -9.77971667
##  [91]   2.55158245 -16.42152600   5.59347917   0.24776116  -3.21006658
##  [96]   7.82274595   5.73389532  13.54943664  -6.59274347  -1.67658746
```

```
colSums(matrix)
```

```
##   [1] -2.037825e+01 -2.657688e-01 -5.802431e+00 -1.596598e+01 -9.690657e+00
##   [6]  1.608592e+01 -1.804986e+01  2.114791e+00  1.468540e+00 -2.715176e+00
##  [11]  3.467055e+00 -1.952165e-01  1.463964e+01 -9.876786e+00  4.957266e-01
##  [16] -1.347120e+00  5.636130e+00 -1.201412e+01  4.196866e+00 -1.206471e+01
##  [21]  1.672957e+01  8.257670e+00 -1.502577e+01 -1.109822e+01  1.571000e+01
##  [26] -2.436824e+01  1.536761e+00 -8.547891e+00  4.631827e+00 -1.859302e+01
##  [31]  1.758185e+00 -5.495701e+00 -1.593242e-04  8.270861e+00  1.254629e+01
```

```
##  [36]  1.350680e+01  1.303743e+01 -2.186567e+01 -1.480827e+00  1.233280e+01
##  [41]  6.329530e-01  5.548793e+00  2.607311e+00  9.381683e+00  6.295803e+00
##  [46] -1.550976e+00  1.473325e+00 -1.020233e+01 -3.830333e+00 -9.554149e+00
##  [51] -7.669489e+00  1.399763e+01 -3.801175e-01 -1.534343e+01  3.165933e+00
##  [56]  7.464777e+00  1.299174e+01 -5.289055e+00 -1.126933e+01  1.186857e+00
##  [61]  1.393459e+00  1.778689e+00 -1.470040e+01  1.237466e+01  6.290203e+00
##  [66] -2.843380e+00  1.228141e+01 -1.209915e+01  9.296355e+00  8.561570e+00
##  [71]  8.647106e+00  4.658601e+00 -1.125660e+01 -6.858778e+00  3.400140e+00
##  [76]  4.746721e+00 -1.952270e+00  7.223008e+00  1.402312e+01  1.307867e-01
##  [81] -4.980046e+00 -5.345778e+00 -8.439010e+00  7.183895e-01  4.271054e+00
##  [86]  5.627819e+00 -3.009682e+00 -6.346351e+00 -1.515026e+01 -7.417148e+00
##  [91]  5.679940e+00  1.024616e+01  2.170326e+01  2.736635e+01 -1.133030e+01
##  [96]  1.964512e+00 -6.910142e+00  6.120836e+00 -3.527931e+00  1.193073e+00
```

```
# 4.
matrix %>% apply(., 1, sum)
```

```
##   [1]   3.25434275   4.28719117   8.32948684   6.24518705  -9.43402013
##   [6]  15.55209184  -3.77115681   2.09667192  12.77860923   7.66498861
##  [11]  -1.21086369  17.52341163   8.47784266   2.69995246   0.77971529
##  [16]  -1.91459430 -11.93012165  -2.56118327 -18.02727143   1.39524373
##  [21] -17.15961938  -7.20340086  -5.11286530  10.42072310   7.09004382
##  [26]   0.65468047  -4.96526056   7.63555058  -1.44651977   5.85829300
##  [31]  -1.77833724  -0.76587767   1.39664242  -6.26221748  -2.57659948
##  [36]  -7.82485876  -5.88414485  -0.50146625  -2.80322215  -9.65361292
##  [41] -12.21816293  -2.35298592  25.58185642   5.64374185  -9.34948371
##  [46]  -7.30675198  -1.93487313  10.78279973  -1.31279310  -8.64731981
##  [51]  -4.13601064  -9.69080797  -0.30954254 -18.02055993   4.00503853
##  [56]   1.55379932  -2.96375310   0.03442909 -14.33630956  19.76535437
##  [61]   5.53886425  -5.72461176  16.89597195  34.03169079   4.62944037
##  [66]  -2.04549852  -4.74767516 -13.51051396   9.85459052   9.72276165
##  [71]  -2.96143187  -4.07379218  22.28229982 -13.75699953  -2.45748933
##  [76]  -0.24298219   5.52778810 -11.16797336   8.23208051 -10.57197993
##  [81]   2.29877570   4.16062460  -9.30873055  -5.01070075 -15.52814219
```

```
##  [86] -13.52056887   0.84436550  -0.92553315  16.34489383  -9.77971667
##  [91]   2.55158245 -16.42152600   5.59347917   0.24776116  -3.21006658
##  [96]   7.82274595   5.73389532  13.54943664  -6.59274347  -1.67658746
```

```
matrix %>% apply(., 2, sum)
```

```
##   [1] -2.037825e+01 -2.657688e-01 -5.802431e+00 -1.596598e+01 -9.690657e+00
##   [6]  1.608592e+01 -1.804986e+01  2.114791e+00  1.468540e+00 -2.715176e+00
##  [11]  3.467055e+00 -1.952165e-01  1.463964e+01 -9.876786e+00  4.957266e-01
##  [16] -1.347120e+00  5.636130e+00 -1.201412e+01  4.196866e+00 -1.206471e+01
##  [21]  1.672957e+01  8.257670e+00 -1.502577e+01 -1.109822e+01  1.571000e+01
##  [26] -2.436824e+01  1.536761e+00 -8.547891e+00  4.631827e+00 -1.859302e+01
##  [31]  1.758185e+00 -5.495701e+00 -1.593242e-04  8.270861e+00  1.254629e+01
##  [36]  1.350680e+01  1.303743e+01 -2.186567e+01 -1.480827e+00  1.233280e+01
##  [41]  6.329530e-01  5.548793e+00  2.607311e+00  9.381683e+00  6.295803e+00
##  [46] -1.550976e+00  1.473325e+00 -1.020233e+01 -3.830333e+00 -9.554149e+00
##  [51] -7.669489e+00  1.399763e+01 -3.801175e-01 -1.534343e+01  3.165933e+00
##  [56]  7.464777e+00  1.299174e+01 -5.289055e+00 -1.126933e+01  1.186857e+00
##  [61]  1.393459e+00  1.778689e+00 -1.470040e+01  1.237466e+01  6.290203e+00
##  [66] -2.843380e+00  1.228141e+01 -1.209915e+01  9.296355e+00  8.561570e+00
##  [71]  8.647106e+00  4.658601e+00 -1.125660e+01 -6.858778e+00  3.400140e+00
##  [76]  4.746721e+00 -1.952270e+00  7.223008e+00  1.402312e+01  1.307867e-01
##  [81] -4.980046e+00 -5.345778e+00 -8.439010e+00  7.183895e-01  4.271054e+00
##  [86]  5.627819e+00 -3.009682e+00 -6.346351e+00 -1.515026e+01 -7.417148e+00
##  [91]  5.679940e+00  1.024616e+01  2.170326e+01  2.736635e+01 -1.133030e+01
##  [96]  1.964512e+00 -6.910142e+00  6.120836e+00 -3.527931e+00  1.193073e+00
```

```
# 5.
statCalc <- function(X) {
  return(c("mean" = mean(X), "sum" = sum(X), "sd" = sd(X)))
}

matrix %>% apply(., 1, statCalc)
```

```
##            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
## mean 0.03254343 0.04287191 0.08329487 0.06245187 -0.0943402  0.1555209
## sum  3.25434275 4.28719117 8.32948684 6.24518705 -9.4340201 15.5520918
## sd   0.95032579 0.91091724 0.89784673 0.91486606  0.9441052  0.9357754
##             [,7]       [,8]       [,9]      [,10]       [,11]      [,12]
## mean -0.03771157 0.02096672  0.1277861 0.07664989 -0.01210864  0.1752341
## sum  -3.77115681 2.09667192 12.7786092 7.66498861 -1.21086369 17.5234116
## sd    0.91281657 0.97036212  1.0323834 0.97079148  1.03635952  0.9617239
##            [,13]      [,14]       [,15]       [,16]       [,17]       [,18]
## mean 0.08477843 0.02699952 0.007797153 -0.01914594  -0.1193012 -0.02561183
## sum  8.47784266 2.69995246 0.779715286 -1.91459430 -11.9301217 -2.56118327
## sd   1.05497189 0.97919709 1.058503760  0.99833189   1.0089540  0.88924209
##            [,19]      [,20]       [,21]       [,22]       [,23]      [,24]
## mean  -0.1802727 0.01395244  -0.1715962 -0.07203401 -0.05112865  0.1042072
## sum  -18.0272714 1.39524373 -17.1596194 -7.20340086 -5.11286530 10.4207231
## sd     0.9655436 0.85433585   1.0496496  0.91805501  0.92295456  0.9148036
##            [,25]       [,26]       [,27]      [,28]       [,29]      [,30]
## mean 0.07090044 0.006546805 -0.04965261 0.07635551 -0.0144652 0.05858293
## sum  7.09004382 0.654680467 -4.96526056 7.63555058 -1.4465198 5.85829300
## sd   1.04825939 1.058683652  0.96848993 1.02415410  0.8764458 1.04434521
##            [,31]        [,32]      [,33]       [,34]       [,35]       [,36]
## mean -0.01778337 -0.007658777 0.01396642 -0.06262217 -0.02576599 -0.07824859
## sum  -1.77833724 -0.765877667 1.39664242 -6.26221748 -2.57659948 -7.82485876
## sd    1.05642228  0.961992912 1.02205877  1.02206121  1.04617647  1.01587072
##            [,37]        [,38]       [,39]       [,40]       [,41]       [,42]
## mean -0.05884145 -0.005014662 -0.02803222 -0.09653613  -0.1221816 -0.02352986
## sum  -5.88414485 -0.501466246 -2.80322215 -9.65361292 -12.2181629 -2.35298592
## sd    1.08739038  0.955967620  0.87782000  0.95330809   0.9723326  1.00421607
##            [,43]      [,44]       [,45]       [,46]       [,47]      [,48]
## mean  0.2558186 0.05643742 -0.09349484 -0.07306752 -0.01934873  0.1078280
## sum  25.5818564 5.64374185 -9.34948371 -7.30675198 -1.93487313 10.7827997
## sd    0.9969324 1.03818591  0.87406379  0.98959160  0.98925076  0.9624982
##            [,49]      [,50]       [,51]       [,52]        [,53]       [,54]
```

```
## mean -0.01312793 -0.0864732 -0.04136011 -0.09690808 -0.003095425  -0.1802056
## sum  -1.31279310 -8.6473198 -4.13601064 -9.69080797 -0.309542542 -18.0205599
## sd    0.94776808  1.0368165  1.00873209  1.08817627  0.943576751   1.0974959
##           [,55]      [,56]       [,57]        [,58]       [,59]      [,60]
## mean 0.04005039 0.01553799 -0.02963753 0.0003442909  -0.1433631  0.1976535
## sum  4.00503853 1.55379932 -2.96375310 0.0344290888 -14.3363096 19.7653544
## sd   1.04975878 1.01605226  1.01976799 0.9307216555   1.0279106  1.1077629
##           [,61]       [,62]      [,63]      [,64]     [,65]       [,66]
## mean 0.05538864 -0.05724612  0.1689597  0.3403169 0.0462944 -0.02045499
## sum  5.53886425 -5.72461176 16.8959720 34.0316908 4.6294404 -2.04549852
## sd   0.92325304  0.93667151  1.0734733  0.9602994 1.1082983  1.09705737
##            [,67]       [,68]      [,69]      [,70]       [,71]       [,72]
## mean -0.04747675  -0.1351051 0.09854591 0.09722762 -0.02961432 -0.04073792
## sum  -4.74767516 -13.5105140 9.85459052 9.72276165 -2.96143187 -4.07379218
## sd    1.05487546   0.9863787 1.00137512 0.95475486  1.09215379  0.98163050
##          [,73]      [,74]       [,75]        [,76]      [,77]       [,78]
## mean  0.222823  -0.137570 -0.02457489 -0.002429822 0.05527788  -0.1116797
## sum  22.282300 -13.757000 -2.45748933 -0.242982186 5.52778810 -11.1679734
## sd    1.054666   1.021476  0.94089822  0.878382796 0.95709089   1.0647143
##           [,79]       [,80]      [,81]      [,82]       [,83]       [,84]
## mean 0.08232081  -0.1057198 0.02298776 0.04160625 -0.09308731 -0.05010701
## sum  8.23208051 -10.5719799 2.29877570 4.16062460 -9.30873055 -5.01070075
## sd   0.93691235   0.9496025 0.97935807 0.87810032  0.98921462  0.96072508
##            [,85]       [,86]       [,87]        [,88]      [,89]       [,90]
## mean  -0.1552814  -0.1352057 0.008443655 -0.009255332  0.1634489 -0.09779717
## sum  -15.5281422 -13.5205689 0.844365503 -0.925533155 16.3448938 -9.77971667
## sd     1.0353484   0.9151718 0.953799578  0.936860974  0.9590071  1.01875529
##           [,91]       [,92]      [,93]       [,94]       [,95]      [,96]
## mean 0.02551582  -0.1642153 0.05593479 0.002477612 -0.03210067 0.07822746
## sum  2.55158245 -16.4215260 5.59347917 0.247761159 -3.21006658 7.82274595
## sd   0.94048616   1.0398835 0.99601972 1.007858468  0.89511203 0.97612879
##           [,97]      [,98]       [,99]      [,100]
## mean 0.05733895  0.1354944 -0.06592743 -0.01676587
```

```
## sum  5.73389532 13.5494366 -6.59274347 -1.67658746
## sd   1.01260707  0.9854024  1.02602999  0.90175548
```

```
matrix %>% apply(., 2, statCalc)
```

```
##             [,1]         [,2]        [,3]        [,4]        [,5]       [,6]
## mean  -0.2037825 -0.002657688 -0.05802431  -0.1596598 -0.09690657  0.1608592
## sum  -20.3782535 -0.265768758 -5.80243107 -15.9659839 -9.69065673 16.0859186
## sd     0.8922498  0.975381429  1.06178429   0.9649244  0.89619057  0.8825617
##             [,7]       [,8]      [,9]       [,10]      [,11]        [,12]
## mean  -0.1804986 0.02114791 0.0146854 -0.02715176 0.03467055 -0.001952165
## sum  -18.0498579 2.11479124 1.4685400 -2.71517554 3.46705464 -0.195216538
## sd     0.9394098 1.04233228 0.8662510  1.04729328 0.99701500  1.003330261
##            [,13]       [,14]       [,15]      [,16]     [,17]       [,18]
## mean  0.1463964 -0.09876786 0.004957266 -0.0134712 0.0563613  -0.1201412
## sum  14.6396392 -9.87678632 0.495726644 -1.3471196 5.6361295 -12.0141159
## sd    0.9395562  0.95851023 1.123398071  1.0546025 0.9506805   1.0258731
##            [,19]       [,20]      [,21]     [,22]       [,23]       [,24]
## mean 0.04196866  -0.1206471  0.1672957 0.0825767  -0.1502577  -0.1109822
## sum  4.19686604 -12.0647136 16.7295736 8.2576700 -15.0257677 -11.0982154
## sd   0.95984503   1.0071233  1.0316454 0.8048382   1.0826439   0.9633243
##            [,25]       [,26]      [,27]       [,28]      [,29]       [,30]
## mean  0.1571000  -0.2436824 0.01536761 -0.08547891 0.04631827  -0.1859302
## sum  15.7099954 -24.3682359 1.53676069 -8.54789097 4.63182690 -18.5930224
## sd    0.9399437   1.0544696 1.03621862  0.86386824 1.01749137   1.0129779
##            [,31]       [,32]          [,33]      [,34]      [,35]      [,36]
## mean 0.01758185 -0.05495701 -1.593242e-06 0.08270861  0.1254629  0.1350680
## sum  1.75818511 -5.49570107 -1.593242e-04 8.27086112 12.5462926 13.5068020
## sd   1.12040394  1.03881759  9.577977e-01 1.06934484  1.0089437  0.9923664
##            [,37]       [,38]       [,39]     [,40]      [,41]      [,42]
## mean  0.1303743  -0.2186567 -0.01480827  0.123328 0.00632953 0.05548793
## sum  13.0374296 -21.8656704 -1.48082693 12.332799 0.63295303 5.54879272
## sd    0.9040058   0.9234217  0.95012932  1.087704 1.04970392 0.94215382
##            [,43]      [,44]      [,45]       [,46]      [,47]       [,48]
```

```
## mean 0.02607311 0.09381683 0.06295803 -0.01550976 0.01473325  -0.1020233
## sum  2.60731145 9.38168258 6.29580289 -1.55097594 1.47332538 -10.2023296
## sd   1.04239648 1.01105660 1.01904577  0.93088628 0.85712071   1.1097638
##            [,49]       [,50]       [,51]      [,52]        [,53]       [,54]
## mean -0.03830333 -0.09554149 -0.07669489  0.1399763 -0.003801175  -0.1534343
## sum  -3.83033286 -9.55414906 -7.66948941 13.9976288 -0.380117504 -15.3434299
## sd    0.79432161  0.96070990  0.92980004  0.8771107  1.036387956   1.0309703
##           [,55]      [,56]      [,57]       [,58]       [,59]      [,60]
## mean 0.03165933 0.07464777  0.1299174 -0.05289055  -0.1126933 0.01186857
## sum  3.16593307 7.46477731 12.9917414 -5.28905482 -11.2693270 1.18685744
## sd   0.96274416 1.12989518  0.9686657  0.96665714   0.9792885 0.96635692
##           [,61]      [,62]      [,63]      [,64]      [,65]      [,66]
## mean 0.01393459 0.01778689  -0.147004  0.1237466 0.06290203 -0.0284338
## sum  1.39345919 1.77868918 -14.700400 12.3746571 6.29020347 -2.8433802
## sd   1.05676576 1.05679957   1.024160  0.9045221 0.96244191  0.9322427
##           [,67]       [,68]      [,69]     [,70]      [,71]      [,72]
## mean  0.1228141  -0.1209915 0.09296355 0.0856157 0.08647106 0.04658601
## sum  12.2814073 -12.0991453 9.29635543 8.5615700 8.64710624 4.65860115
## sd    1.0740917   0.9304074 0.85987228 0.9974854 0.94022756 1.03883259
##            [,73]       [,74]     [,75]      [,76]      [,77]      [,78]
## mean  -0.1125660 -0.06858778 0.0340014 0.04746721 -0.0195227 0.07223008
## sum  -11.2565963 -6.85877836 3.4001399 4.74672106 -1.9522698 7.22300752
## sd     0.9709917  1.01645858 1.0134294 0.99654792  1.0097013 1.01255883
##           [,79]       [,80]       [,81]       [,82]      [,83]       [,84]
## mean  0.1402312 0.001307867 -0.04980046 -0.05345778 -0.0843901 0.007183895
## sum  14.0231215 0.130786678 -4.98004579 -5.34577784 -8.4390105 0.718389511
## sd    0.9598838 0.907860942  0.96959282  1.08018892  1.0490608 1.014749043
##           [,85]      [,86]       [,87]       [,88]       [,89]       [,90]
## mean 0.04271054 0.05627819 -0.03009682 -0.06346351  -0.1515026 -0.07417148
## sum  4.27105368 5.62781861 -3.00968162 -6.34635068 -15.1502613 -7.41714791
## sd   1.02193114 0.98283992  0.96898696  1.01817155   1.0198756  0.93869728
##          [,91]      [,92]      [,93]      [,94]      [,95]      [,96]
## mean 0.0567994  0.1024616  0.2170326  0.2736635  -0.113303 0.01964512
```

```
## sum  5.6799398 10.2461597 21.7032645 27.3663531 -11.330295 1.96451207
## sd   0.9225007  1.0078425  1.1148935  0.9979616   1.043460 0.87546729
##            [,97]      [,98]       [,99]     [,100]
## mean -0.06910142 0.06120836 -0.03527931 0.01193073
## sum  -6.91014176 6.12083556 -3.52793139 1.19307343
## sd    0.97121452 0.96818799  0.87383856 0.94030218
```

0.5.2 用 mtcars 进行练习

用 tapply 练习：

1. 用 **汽缸数**分组，计算 **油耗**的 **平均值**；
2. 用 **汽缸数**分组，计算 **wt** 的 **平均值**；

用 dplyr 的函数实现上述计算

```
## 代码写这里，并运行；
library(magrittr)
```

```
##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
##     set_names

## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
# 1.
mtcars %$% tapply(mpg, cyl, mean)
```

```
##        4        6        8
## 26.66364 19.74286 15.10000
```

```r
# 2.
mtcars %$% tapply(wt, cyl, mean)
```

```
##        4        6        8
## 2.285727 3.117143 3.999214
```

0.5.3 练习 lapply 和 sapply

1. 分别用 lapply 和 sapply 计算下面 list 里每个成员 vector 的长度:

```r
list( a = 1:10, b = letters[1:5], c = LETTERS[1:8] );
```

2. 分别用 lapply 和 sapply 计算 mtcars 每列的平均值;

```r
## 代码写这里, 并运行;
list <- list(a = 1:10, b = letters[1:5], c = LETTERS[1:8])

# 1.
list %>% lapply(length)
```

```
## $a
## [1] 10
##
## $b
```

```
## [1] 5
##
## $c
## [1] 8
```

```
list %>% sapply(length)
```

```
##  a  b  c
## 10  5  8
```

```
# 2.
mtcars %>% lapply(mean)
```

```
## $mpg
## [1] 20.09062
##
## $cyl
## [1] 6.1875
##
## $disp
## [1] 230.7219
##
## $hp
## [1] 146.6875
##
## $drat
## [1] 3.596563
##
## $wt
## [1] 3.21725
##
## $qsec
## [1] 17.84875
##
```

```
## $vs
## [1] 0.4375
##
## $am
## [1] 0.40625
##
## $gear
## [1] 3.6875
##
## $carb
## [1] 2.8125
```

```
mtcars %>% sapply(mean)
```

```
##        mpg        cyl       disp         hp       drat         wt       qsec
##  20.090625   6.187500 230.721875 146.687500   3.596563   3.217250  17.848750
##         vs         am       gear       carb
##   0.437500   0.406250   3.687500   2.812500
```

0.6 练习与作业 3：loop 进阶，purr 包的函数

0.6.1 map 初步

生成一个变量：

```
df <- tibble(
  a = rnorm(10),
  b = rnorm(10),
  c = rnorm(10),
  d = rnorm(10)
)
```

用 map 计算：

- 列平均值、总和和中值

```
## 代码写这里，并运行；
df %>% map(~c(mean(.), sum(.), median(.)))
```

```
## $a
## [1] 0.1011186 1.0111857 0.3470321
##
## $b
## [1] -0.6576212 -6.5762116 -0.7543649
##
## $c
## [1]  0.07390339  0.73903387 -0.20994951
##
## $d
## [1] -0.2460700 -2.4606996 -0.4215982
```

0.6.2 map 进阶

用 map 配合 purr 包中其它函数，用 mtcars：

为每一个 **汽缸数**计算燃油效率 mpg 与重量 wt 的相关性（Pearson correlation），得到 p 值和 correlation coefficient 值。

```
## 代码写这里，并运行；
mtcars %>%
  split(.$cyl) %>%
  map(~cor.test(.$mpg, .$wt)) %>%
  map(~c("p.value" = .$p.value, .$estimate))
```

```
## $`4`
##     p.value         cor
##  0.01374278 -0.71318483
##
## $`6`
##     p.value         cor
##  0.09175766 -0.68154982
##
## $`8`
##     p.value         cor
##  0.01179281 -0.65035801
```

0.6.3 keep 和 discard

1. 保留 iris 中有 factor 的列，并打印前 10 行；
2. 去掉 iris 中有 factor 的列，并打印前 10 行；

```
## 代码写这里，并运行；
# 保留 factor
iris %>%
  keep(is.factor) %>%
  head(n = 10)
```

```
##    Species
## 1   setosa
## 2   setosa
## 3   setosa
## 4   setosa
## 5   setosa
## 6   setosa
## 7   setosa
## 8   setosa
```

```
## 9   setosa
## 10  setosa
```

```
# 移除 factor
iris %>%
  discard(is.factor) %>%
  head(n = 10)
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1           5.1         3.5          1.4         0.2
## 2           4.9         3.0          1.4         0.2
## 3           4.7         3.2          1.3         0.2
## 4           4.6         3.1          1.5         0.2
## 5           5.0         3.6          1.4         0.2
## 6           5.4         3.9          1.7         0.4
## 7           4.6         3.4          1.4         0.3
## 8           5.0         3.4          1.5         0.2
## 9           4.4         2.9          1.4         0.2
## 10          4.9         3.1          1.5         0.1
```

0.6.4 用 reduce

用 reduce 得到以下三个 vector 中共有的数字：

```
c(1, 3, 5, 6, 10),
  c(1, 2, 3, 7, 8, 10),
  c(1, 2, 3, 4, 8, 9, 10)
```

```
## 代码写这里，并运行；
vec <- list(
  c(1, 3, 5, 6, 10),
  c(1, 2, 3, 7, 8, 10),
```

```
  c(1, 2, 3, 4, 8, 9, 10)
)

vec %>% reduce(intersect)
```

```
## [1]  1  3 10
```

0.6.5 运行以下代码，观察得到的结果，并用 tidyverse 包中的 spread 等函数实现类似的结果

```
dfs <- list(
  age = tibble(name = "John", age = 30),
  sex = tibble(name = c("John", "Mary"), sex = c("M", "F")),
  trt = tibble(name = "Mary", treatment = "A")
);

dfs %>% reduce(full_join);
```

```
## 代码写这里，并运行;
dfs <- list(
  age = tibble(name = "John", age = 30),
  sex = tibble(name = c("John", "Mary"), sex = c("M", "F")),
  trt = tibble(name = "Mary", treatment = "A")
)

dfs %>%
  bind_rows() %>%
  gather(key = "key", value = "value", -name, na.rm = TRUE) %>%
  spread(key, value)
```

```
## # A tibble: 2 x 4
```

```
##   name  age   sex   treatment
##   <chr> <chr> <chr> <chr>
## 1 John  30    M     <NA>
## 2 Mary  <NA>  F     A
```

0.7 练习与作业 4：并行计算

0.7.1 安装相关包，成功运行以下代码，观察得到的结果，并回答问题

* parallel
* foreach
* iterators

```
library(parallel); ##
library(foreach);
```

```
##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
library(iterators);

## 检测有多少个 CPU --
( cpus <- parallel::detectCores() );
```

```
## [1] 8
```

```
## 创建一个 data.frame
d <- data.frame(x=1:10000, y=rnorm(10000));

## make a cluster --
cl <- makeCluster( cpus - 1 );

## 分配任务 ...
res <- foreach( row = iter( d, by = "row" ) ) %dopar% {
  return ( row$x * row$y  );
}
```

```
## Warning: executing %dopar% sequentially: no parallel backend registered
```

```
## 注意在最后关闭创建的 cluster
stopCluster( cl );

summary(unlist(res));
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -27356.11  -2568.42      6.56     74.25   2614.83  41822.59
```

问：你的系统有多少个 CPU？此次任务使用了多少个？**答**：用代码打印出相应的数字即可：

```
## 代码写这里，并运行；
c(8, 7)
```

```
## [1] 8 7
```