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] "wchen"
```

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

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

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；
```

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;
```

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;
```

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:

```
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;

```
## 代码写这里，并运行;
```

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;
```

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;
```

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；
```

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；
```

0.6.5 运行以下代码，观察得到的结果，并用 tidyverse 包中的 pivot_wider 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；
```

0.7 练习与作业 4：pmap 和 map 的更多用法

请参考 https://r4ds.had.co.nz/iteration.html 的 Mapping over multiple arguments 部分

0.7.1 map2

运行以下代码，查看输出结果。用 for 循环重现计算结果。

```
mu <- list(5, 10, -3);
sigma <- list(1, 5, 10);
map2(mu, sigma, rnorm, n = 5)
```

```
## 代码写这里，并运行;
```

0.7.2 pmap

运行以下代码，查看输出结果。用 for 循环重现计算结果。

```
params <- tribble(
  ~mean, ~sd, ~n,
    5,     1,  1,
   10,     5,  3,
   -3,    10,  5
)
params %>%
  pmap(rnorm)
```

```
## 代码写这里，并运行;
```

0.8 练习与作业 5：并行计算

0.8.1 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);
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.
## -33153.21  -2601.69      1.87     78.36   2681.23  30596.30
```

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;
```