

R for Data Science

Bài 2: R cơ bản

Phòng LT & Mạng

https://csc.edu.vn/lap-trinh-va-cSDL/R-Programming-Language-for-Data-Science_190

2020



Nội dung



1. Kiểu dữ liệu
2. Biến
3. Toán tử
4. Nhập/ xuất trong R
5. String
6. Date

Kiểu dữ liệu

□ Đều là R-object, có các kiểu dữ liệu sau:

- Vector
- Factor
- List
- Matrix
- Array
- Data Frame



Kiểu dữ liệu

□ Vector: các kiểu dữ liệu hay dùng

- Logical (TRUE/ FALSE)

```
> pass <- TRUE
> print(class(pass))
[1] "logical"
```

- Numeric

```
> a <- 12.3
> b <- 150
> print(class(a))
[1] "numeric"
```

- Integer

```
> int <- 2L
> print(class(int))
[1] "integer"
```

- Character

```
> ky_tu <- 'a'
> print(class(ky_tu))
[1] "character"
```



Kiểu dữ liệu

☐ Vector

- Chú ý: khi tạo ra một vector có nhiều hơn một phần tử thì sử dụng phương thức **c()** để kết hợp các phần tử vào vector

- Ví dụ:

```
> fruit <- c("apple", "pear", "orange", "banana")
> print(class(fruit))
[1] "character"
```

(Sau có 1 phần tử có class là character thì
một số
tổng vector
có class
là character)



Kiểu dữ liệu

☐ Factor

- Được tạo nên từ vector nhưng chỉ chứa các phần tử duy nhất trong vector
- Phương thức **nlevels()** trả ra số lượng các mức trong factor

```
> # Create a vector.
> apple_colors <- c('green', 'green', 'yellow', 'red', 'red', 'red', 'green', 'white')
> # Create a factor object.
> factor_apple <- factor(apple_colors)
> # Print the factor.
> print(factor_apple)
[1] green green yellow red     red     red     green white
Levels: green red white yellow
> print(nlevels(factor_apple))
[1] 4
```



Kiểu dữ liệu

☐ List

List trong R index bắt đầu từ 1

- Là một danh sách có thể chứa nhiều phần tử với các kiểu khác nhau (vector, function, list)

```
> # Create a list.
> list1 <- list(c(2,5,3), 21.3, "spring")
>
> # Print the list.
> print(list1)
[[1]]
[1] 2 5 3

[[2]]
[1] 21.3

[[3]]
[1] "spring"
```

R programming language for Data Science

7

Kiểu dữ liệu

☐ Matrix

- Là một ma trận hai chiều (vuông, chữ nhật)

```
> # Create a matrix.
> M = matrix( c(1,2,3,4,5,6,7,8), nrow = 2, ncol = 4, byrow = TRUE)
> print(M)
 [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
> M = matrix( c(1,2,3,4,5,6,7,8,9), nrow = 3)
> print(M)
 [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

R programming language for Data Science

8

Kiểu dữ liệu

❑ Array

- Là một ma trận nhiều chiều

```
> a <- array(c('red', 'green', 'yellow'), dim = c(3, 3, 2))
> print(a)
, , 1

[,1]      [,2]      [,3]
[1,] "red"    "red"    "red"
[2,] "green"   "green"   "green"
[3,] "yellow"  "yellow"  "yellow"

, , 2

[,1]      [,2]      [,3]
[1,] "red"    "red"    "red"
[2,] "green"   "green"   "green"
[3,] "yellow"  "yellow"  "yellow"
```

R programming language for Data Science

9



Kiểu dữ liệu

❑ Data Frame

- Là các đối tượng dữ liệu dạng bảng, được sử dụng nhiều nhất. Gồm danh sách các vector có cùng độ dài.
- Sử dụng phương thức `data.frame()` để tạo.

```
> # Create the data frame.
> BMI <- data.frame(
+   gender = c("Male", "Male", "Female"),
+   height = c(152, 171.5, 165),
+   weight = c(81, 93, 78),
+   age = c(42, 38, 26)
+ )
> print(BMI)
  gender height weight age
1  Male   152.0     81   42
2  Male   171.5     93   38
3 Female  165.0     78   26
```



10

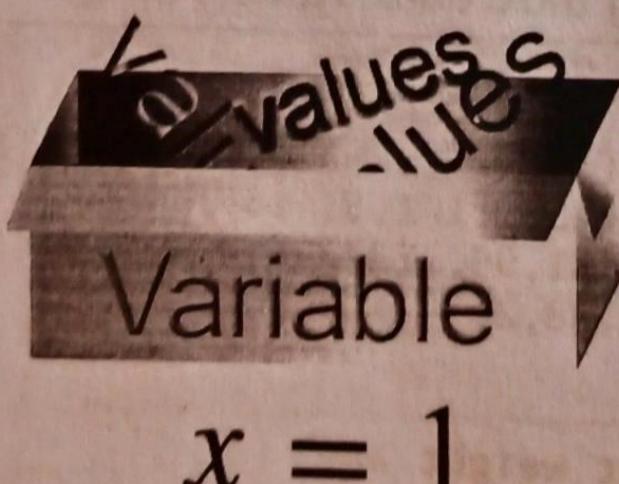
Nội dung

1. Kiểu dữ liệu
2. Biến
3. Toán tử
4. Nhập/ xuất trong R
5. String
6. Date



Biến

☐ Biến là một đơn vị lưu trữ trong máy tính. Khi khai báo biến sẽ có một bộ nhớ tương ứng được cấp để lưu trữ.



Biến

□ Khai báo biến

- Cách 1: Tên_variabile <- Dữ_lieu_khởi_tạo
- Cách 2: Tên_variabile = Dữ_lieu_khởi_tạo
- Cách 3: Dữ_lieu_khởi_tạo -> Tên_variabile

▪ Ví dụ

```
> # cách 1
> colors <- c("red", "green", "blue")
> print(colors)
[1] "red"   "green" "blue"
> # cách 2
> numbers = c(10, 15.5, 7)
> print(numbers)
[1] 10.0 15.5 7.0
> # cách 3
> c(TRUE, FALSE, TRUE, TRUE) -> pass
> print(pass)
[1] TRUE FALSE TRUE TRUE
```

Quy tắc đặt tên biến:

- Biến hợp lệ chứa ký tự, số, dấu ., dấu _,
- Tên biến không nên bắt đầu bằng _

Science

13

Biến

- Tên biến không sử dụng các từ khóa của R

▪ Gõ **?reserved** tại console để biết các từ khóa:

R: Reserved Words in R ▾ Find in Topic

Reserved {base} R Documentation ^

Reserved Words in R

Description

The reserved words in R's parser are

[if](#) [else](#) [repeat](#) [while](#) [function](#) [for](#) [in](#) [next](#) [break](#)

[TRUE](#) [FALSE](#) [NULL](#) [Inf](#) [NaN](#) [NA](#) [NA_integer_](#) [NA_real_](#) [NA_complex_](#) [NA_character_](#)

... and ..1, ..2 etc, which are used to refer to arguments passed down from a calling function. See the [Introduction to R](#) manual for usage of these syntactic elements, and [dotsMethods](#) for their use in formal methods.

□ Đọc danh sách các biến đang tồn tại trong workspace: dùng phương thức `ls()`

```
> print(ls())
[1] "a"                 "age"                "apple_colors" "b"
[6] "c"                 "colors"              "d"                  "date"
[11] "factor_apple"    "fruit"               "fun"                "int"
[16] "list1"              "M"                  "my.name"            "myString"
[21] "Nhap_so"            "numbers"             "pass"                "year"
                                         "BMI"                "e1"                  "ky_tu"
                                         "name"
```



R programming language for Data Science

Nội dung

1. Kiểu dữ liệu
2. Biến
3. Toán tử
4. Nhập/ xuất trong R
5. String
6. Date



R programming language for Data Science

Toán tử

□ Toán tử số học (Arithmetic Operators)

Toán tử	Diễn giải	Ví dụ
+	Cộng	<pre>> v <- c(2, 5.5, 6) > t <- c(8, 3, 4) > print(v+t) [1] 10.0 8.5 10.0</pre>
-	Trừ	<pre>> print(v-t) [1] -6.0 2.5 2.0</pre>
*	Nhân	<pre>> print(v*t) [1] 16.0 16.5 24.0</pre>
/	Chia	<pre>> print(v/t) [1] 0.250000 1.833333 1.500000</pre>
%%	Chia lấy phần dư	<pre>> print(v%%t) [1] 2.0 2.5 2.0</pre>
%/%	Chia lấy phần nguyên	<pre>> print(v%/%t) [1] 0 1 1</pre>
^	vector 1 mũ vector 2	<pre>> print(v^t) [1] 256.000 166.375 1296.000</pre>

R programming language for Data Science

17

Toán tử

□ Toán tử quan hệ (Relational Operators)

Toán tử	Diễn giải	Ví dụ
v <- c(2, 5.5, 6, 7)		
t <- c(8, 3, 4, 7)		
>	Lớn hơn	<pre>> print(v>t) [1] FALSE TRUE TRUE FALSE</pre>
<	Nhỏ hơn	<pre>> print(v<t) [1] TRUE FALSE FALSE FALSE</pre>
==	Bằng	<pre>> print(v==t) [1] FALSE FALSE FALSE TRUE</pre>
>=	Lớn hơn hoặc bằng	<pre>> print(v>=t) [1] FALSE TRUE TRUE TRUE</pre>
<=	Nhỏ hơn hoặc bằng	<pre>> print(v<=t) [1] TRUE FALSE FALSE TRUE</pre>
!=	Khác	<pre>> print(v!=t) [1] TRUE TRUE TRUE FALSE</pre>

R programming language for Data Science

18

Toán tử

☐ Toán tử logic (Logical Operators)

- Chỉ xét cho vector kiểu logic, numeric
- Tất cả số các giá trị ≥ 1 đều có giá trị Logic là TRUE
- Lần lượt cho từng cặp/giá trị của element trong vector

Toán tử	Điễn giải	Ví dụ
&	Chỉ TRUE khi tất cả là TRUE	<pre>> v <- c(3, 1, TRUE, 0) > t <- c(4, 1, FALSE, 2) > print(v&t) [1] TRUE TRUE FALSE FALSE</pre>
	Chỉ FALSE khi tất cả là FALSE	<pre>> print(v t) [1] TRUE TRUE TRUE TRUE > print(!v) [1] FALSE FALSE FALSE TRUE > print(!t) [1] FALSE FALSE TRUE FALSE</pre>
!	Phủ định	

R programming language for Data Science

19

Toán tử

☐ Toán tử logic (Logical Operators)

- Xem xét giá trị đầu tiên của hai vector

Toán tử	Điễn giải	Ví dụ
&&	Chỉ TRUE khi tất cả là TRUE	<pre>> v <- c(FALSE, 1, TRUE, 0) > t <- c(4, 1, FALSE, 2) > print(v&&t) [1] FALSE</pre>
	Chỉ FALSE khi tất cả là FALSE	<pre>> print(v t) [1] TRUE</pre>

R programming language for Data Science

20

Toán tử

□ Toán tử gán (Assignment Operators)

Toán tử	Điễn giải	Ví dụ
<- = <<-	Gán trái (Left Assignment)	<pre>> v1 <- c(3,1,TRUE) > v2 <-< c(3,1,TRUE) > v3 = c(3,1,TRUE) > print(v1) [1] 3 1 1 > print(v2) [1] 3 1 1 > print(v3) [1] 3 1 1</pre>
-> ->>	Gán phải (Right Assignment)	<pre>> c(3,1,TRUE) -> v1 > c(3,1,TRUE) ->> v2 > print(v1) [1] 3 1 1 > print(v2) [1] 3 1 1</pre>

Toán tử

□ Các toán tử khác (Miscellaneous Operators)

Toán tử	Điễn giải	Ví dụ
:	Tạo ra loạt các số từ bắt đầu : kết thúc	<pre>> v <- 1:10 > print(v) [1] 1 2 3 4 5 6 7 8 9 10</pre>
%in%	Kiểm tra xem một giá trị có nằm trong vector hay không	<pre>> v1 <- 8 > v2 <- 12 > v <- c(2, 4, 3, 8, 9, 11) > print(v1 %in% v) [1] TRUE > print(v2 %in% v) [1] FALSE</pre>
%*%	Nhân và cộng tổng các phép nhân trong ma trận nxm * mxn=> ma trận mới nxn	<pre>> M = matrix(c(1, 1, 1, 2, 2, 2), nrow = 2, ncol = 3, byrow = TRUE) > print(M) [,1] [,2] [,3] [1,] 1 1 1 [2,] 2 2 2 > print(t(M)) [,1] [,2] [1,] 1 2 [2,] 1 2 [3,] 1 2 > t = M %*% t(M) > print(t) [,1] [,2] [1,] 3 6 [2,] 6 12 > 32</pre>

Nội dung

1. Kiểu dữ liệu
2. Biến
3. Toán tử
4. Nhập/ xuất trong R
5. String
6. Date



Nhập/ xuất trong R

☐ Nhập

`Tên_biến <- readline(prompt="Nhập ...")`

▪ Ví dụ: Nhập tên

```
name <- readline(prompt="Enter name: ")
```

● Lưu ý: khi nhập liệu, dữ liệu nhập vào là kiểu chuỗi, khi cần thì chuyển đổi kiểu dữ liệu cho phù hợp

▪ Ví dụ: Nhập và chuyển tuổi thành kiểu integer

```
age <- readline(prompt="Enter age: ")
```

```
age <- as.integer(age)
```



□ Xuất

`print("Nội dung cần xuất")`

- Ví dụ: Xuất tên

```
print(name)
```

- Ráp chuỗi: tạo một chuỗi từ nhiều chuỗi => dùng `paste(chuỗi 1, chuỗi 2, ...)`

- Ví dụ: Xuất tuổi

```
print(paste("Hi,", name, "next year you will be",
age + 1, "years old.))
```



□ Xuất

- In nhiều chuỗi: => dùng `cat(chuỗi 1, chuỗi 2, ...)`

- Ví dụ:

```
> colors <- c("red", "green", "blue")
> cat ("colors are ", colors ,"\n")
colors are red green blue
```



Nhập/ xuất trong R

□ Ghi chú

- Sử dụng # để ghi chú trên một dòng
- # ghi chú trên 1 dòng
- R không hỗ trợ ghi chú trên nhiều dòng



Nội dung

1. Kiểu dữ liệu
2. Biến
3. Toán tử
4. Nhập/ xuất trong R
5. String
6. Date



String

❑ Nguyên tắc tạo chuỗi

- Sử dụng “” hoặc ” để bao chuỗi
- Trong chuỗi bắt đầu và kết thúc bằng ” có thể có nội dung “”
- Trong chuỗi bắt đầu và kết thúc bằng “” có thể có nội dung “”
- Trong chuỗi “” không có nội dung “”
- Trong chuỗi “” không có nội dung “”



String

❑ Ví dụ

```
> # demo chuoi
> s1 <- "This is first sentence."
> s2 <- 'This is second sentence.'
> s3 <- "This is 'third' sentence"
> s4 <- 'This is "fourth" sentence'
> print(s1)
[1] "This is first sentence."
> print(s2)
[1] "This is second sentence."
> print(s3)
[1] "This is 'third' sentence"
> print(s4)
[1] "This is \"fourth\" sentence"
```



❑ parse()

- Tạo một chuỗi từ nhiều chuỗi khác nhau

- Ví dụ:

```
> a <- "one"
> b <- 'two'
> c <- "three"
>
> print(paste(a,b,c))
[1] "one two three"
>
> print(paste(a,b,c, sep = "-"))
[1] "one-two-three"
>
> print(paste(a,b,c, sep = ""))
[1] "onetwothree"
```

R programming language for Data Science



❑ nchar()

- Đếm số ký tự trong chuỗi

- Ví dụ:

```
> s <- "This is a string"
> print(nchar(s))
[1] 16
```



R programming language for Data Science

String

❑ toupper() & tolower()

- Đổi chuỗi thành chữ hoa/ chữ thường

- Ví dụ:

```
> s <- "This is a string"
> print(toupper(s))
[1] "THIS IS A STRING"
> print(tolower(s))
[1] "this is a string"
```



String

❑ substring(s, first, last)

- Tạo chuỗi con từ chuỗi s, có nội dung từ first đến last

- Ví dụ:

```
> s <- "This is big string"
> s_sub <- substring(s, 5, 12)
> print(s_sub)
[1] " is big "
```



❑ **chartr(old, new, string_name)**

- Tìm kiếm và thay thế ký tự trong chuỗi
- Ví dụ:

```
> s1 <- "Today is New year day"  
> print(chartr(" ", "-", s1))  
[1] "Today-is-New-year-day"
```



❑ **strsplit(string_name, delimiter)**

- Cắt chuỗi thành danh sách các phần tử
- Ví dụ:

```
> print(s1)  
[1] "Today is New year day"  
> character_list <- strsplit(s1, " ")  
> word_list <- unlist(character_list)  
> print(word_list)  
[1] "Today" "is" "New" "year" "day"
```



String

❑ sort()

- Sắp xếp các phần tử chuỗi

- Ví dụ:

```
> print(word_list)
[1] "Today" "is"      "New"     "year"    "day"
> sorted_list <- sort(word_list)
> print(sorted_list)
[1] "day"    "is"     "New"    "Today"   "year"
```



Nội dung

1. Kiểu dữ liệu
2. Biến
3. Toán tử
4. Nhập/ xuất trong R
5. String
6. Date



□ date()

- Lấy ngày hiện hành trên hệ thống

- Ví dụ:

```
> print(date())
[1] "Sun Feb 11 15:19:46 2018"
> print(Sys.Date())
[1] "2018-02-11"
```

□ as.Date("chuỗi ngày"[, format("chuỗi định dạng")))

- Tạo ngày từ chuỗi ngày

```
> d1 <- "20-10-2018"
> print(d1)
[1] "20-10-2018"
> print(as.Date("20/10/2018", format("%d/%m/%Y")))
[1] "2018-10-20"
```

R programming language for Data Science

39

□ Định dạng hiển thị datetime

Symbol	Meaning	Example
%d	day as a number (0-31)	01-31
%a %A	abbreviated weekday unabbreviated weekday	Mon Monday
%m	month (01-12)	01-12
%b %B	abbreviated month unabbreviated month	Jan January
%y %Y	2-digit year 4-digit year	18 2018

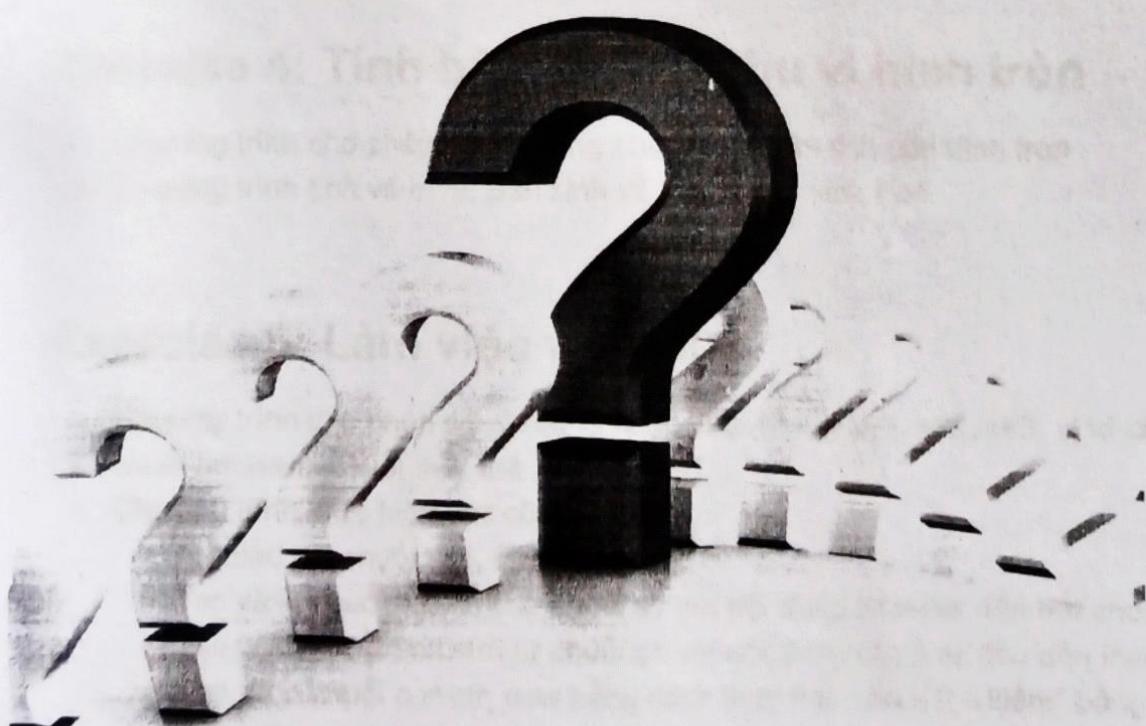
```
> print(format(Sys.Date(), format="%B %d %Y"))
[1] "February 11 2018"
> print(format(Sys.Date(), format="%d/%m/%Y"))
[1] "11/02/2018"
```



□ Khoảng cách giữa 2 ngày

● Ví dụ:

```
> distance <- Sys.Date() - as.Date("2018-01-01")
> print(distance)
Time difference of 41 days
> difftime(Sys.Date(), as.Date("2018-01-01"), units = "weeks")
Time difference of 5.857143 weeks
```





Chapter 2 - R cơ bản

Exercise 1: Thông tin cá nhân

- Chương trình cho phép người dùng nhập vào: tên, năm sinh, cân nặng (kg), chiều cao (m).
- Tính BMI theo công thức: $BMI = \text{cân nặng} / (\text{chiều cao} * \text{chiều cao})$
- Xuất ra thông tin của người dùng theo định dạng: "Name: xxx, Year of Birth: xxxx, Height: x.xx (m), Weight: xxx (kg), BMI = xx.xx

Exercise 2: Tính tiền hóa đơn ăn uống

- Chương trình cho phép người dùng nhập vào: Tổng số tiền các món ăn & nước uống, Thuế phải trả cho hóa đơn : từ 10 – 20 (%) (của Tổng số tiền các món ăn & nước uống), Tip: từ 5 – 10 (%) (của Tổng số tiền các món ăn & nước uống)
- Chương trình tính và in ra: Thuế phải trả, Tip, Tổng số tiền cần thanh toán = Tổng số tiền các món ăn & nước uống + thuế + tip

Exercise 3: Tính tiền lãi gửi tiết kiệm

- Chương trình cho phép người dùng nhập vào: Lãi suất một năm, số tiền gửi và số ngày gửi.
- Biết Lãi suất ngày = Lãi suất năm/365/100, Tiền lãi = ($\text{Số tiền gửi} * \text{số ngày gửi}$) * Lãi suất ngày, Tổng số tiền = Số tiền gửi + Tiền lãi
- Chương trình tính và in ra: Tiền lãi và tổng số tiền nhận được sau khi hết thời hạn gửi tiền.

Exercise 4: Tính bán kính và chu vi hình tròn

- Chương trình cho phép người dùng nhập vào: Diện tích của hình tròn
- Chương trình tính và in ra: Bán kính và chu vi của hình tròn

Exercise 5: Làm việc với chuỗi

- Chương trình cho phép người dùng nhập vào: Chuỗi str1, str2, str3, vị trí cắt chuỗi (index), chuỗi tìm kiếm, chuỗi thay thế
- Chương trình thực hiện các công việc:
 - In chiều dài chuỗi str1, str2, str3
 - Tạo và in chuỗi con str4 từ chuỗi s1 với nội dung từ index đến hết chuỗi
 - Tạo và in chuỗi con str5 từ chuỗi s1 với nội dung chuỗi từ đầu đến index
 - Tạo và in chuỗi con str_new bằng cách thay thế "chuỗi tìm kiếm" bằng "chuỗi thay thế" cho str1

Exercise 6: Khoảng cách giữa 2 ngày

- Chương trình cho phép người dùng nhập vào: chuỗi ngày tháng năm theo mẫu dd-mm-YYYY
- Chương trình tính và in ra: Khoảng cách giữa ngày hiện tại (lấy trên hệ thống) và ngày vừa nhập

Exercise 7: Bảy ngày tiếp theo

- Chương trình cho phép người dùng nhập vào: Chuỗi ngày tháng năm theo mẫu dd-mm-YYYY
- Chương trình kiểm tra nếu ngày nhập vào hợp lệ thì cho biết thứ của ngày này và in 7 ngày liên tục tính từ ngày này. Nếu không hợp lệ thì thông báo.

Gợi ý:

Exercise 1: Thông tin cá nhân

```
In [1]: # input personality information
name <- readline(prompt="Enter name: ")

yob <- readline(prompt="Enter year of birth: ")
yob <- as.integer(yob)

height <- readline(prompt="Enter height: ")
height <- as.double(height)

weight <- readline(prompt="Enter weight: ")
weight <- as.double(weight)

Enter name: John
Enter year of birth: 2000
Enter height: 1.75
Enter weight: 70
```

```
In [2]: bmi = weight/(height^2)
```

```
In [3]: print(paste("Name:", name, ", Year of Birth:", yob))
print(paste("Height:", round(height,2), "(m), Weight:", 
           round(weight,2), "(kg), BMI: ", round(bmi,2)))
[1] "Name: John , Year of Birth: 2000"
[1] "Height: 1.75 (m), Weight: 70 (kg), BMI: 22.86"
```

Exercise 2: Tính tiền hóa đơn ăn uống



```
In [4]: # Information
sum_of_food_and_drink <- readline(prompt="Total of foods and drinks: ")
sum_of_food_and_drink <- as.integer(sum_of_food_and_drink)

tax <- readline(prompt="Tax (10-20 %): ")
tax <- as.integer(tax)

tip <- readline(prompt="Tip (5-10 %)/:")
tip <- as.integer(tip)

Total of foods and drinks: 500000
Tax (10-20 %): 10
Tip (5-10 %)/:5

In [5]: tax.m = sum_of_food_and_drink * tax/100
tip.m = sum_of_food_and_drink * tip/100

In [6]: total = sum_of_food_and_drink + tax.m + tip.m

In [7]: print(paste("Total of foods and drinks: ",
                  sum_of_food_and_drink, "VND"))
print(paste("Tax: ", tax.m, "VND"))
print(paste("Tip: ", tip.m, "VND"))
print(paste("You need to pay (total): ",
            format(total, scientific = FALSE), "VND"))

[1] "Total of foods and drinks: 500000 VND"
[1] "Tax: 5000 VND"
[1] "Tip: 2500 VND"
[1] "You need to pay (total): 575000 VND"
```

Exercise 3: Tính tiền lãi gửi tiết kiệm

```
In [8]: lai_suat_nam <- readline(prompt = "Input interest rate/year (%): ")
lai_suat_nam <- as.numeric(lai_suat_nam)

so_tien_1 <- readline(prompt = "Input amount of money:")
so_tien <- as.numeric(so_tien_1)

so_ngay_gui <- readline(prompt = "Input number of days:")
so_ngay_gui <- as.integer(so_ngay_gui)

Input interest rate/year (%): 7
Input amount of money:100000000
Input number of days:365
```

```
In [9]: lai_suat_ngay = (lai_suat_nam/365)/100
```

```
In [10]: tien_lai = so_tien * lai_suat_ngay * so_ngay_gui
tien_gui_va_lai = so_tien + tien_lai
```



```
In [22]: result = paste("Interest rate/day: ", lai_suat_ngay,
                      "\nAmount of money: ", so_tien_1, " after ", so_ngay_gui,
                      " days => Saving interest money: ", round(tien_lai,2),
                      "\nSum:", format(tien_gui_va_lai, big.mark=".",
                      decimal.mark = ",",
                      nsmall = 2),sep="")
```

```
In [23]: cat(result)
```

```
Interest rate/day: 0.000191780821917808
Amount of money: 10000000 after 365 days => Saving interest money: 7e+05
Sum:10.700.000,00
```

Exercise 4: Tính bán kính và chu vi hình tròn

```
In [24]: S <- readline(prompt = "Input S of cricle:")
S <- as.double(S)

Input S of cricle:10
```

```
In [25]: PI <- 3.14
```

```
R <- sqrt(S/PI)
P <- 2 * PI * R
```

```
In [26]: print(paste("R of circle:", round(R,2)))
print(paste("P of circle:", round(P,2)))

[1] "R of circle: 1.78"
[1] "P of circle: 11.21"
```

Exercise 5: Làm việc với chuỗi

```
In [27]: str1 <- readline(prompt = "Input s1:")
str2 <- readline(prompt = "Input s2:")
str3 <- readline(prompt = "Input s3:")

index <- as.integer(readline(prompt = "Input index:"))
str_find <- readline(prompt = "Find_string:")
str_replace <- readline(prompt = "Replace_string:")

Input s1:This is R programming language
Input s2:for Data Science
Input s3:and Machine Learning
Input index:10
Find_string:language
Replace_string:LANGUAGE
```



```
In [28]: str4 <- substring(str1, index)
str5 <- substring(str1, 1, index)
# thay the 1 chuoi voi 1 chuoi dau tien thoai
str_new <- sub(str_find, str_replace, str1)

# thay the 1 chuoi cho tat ca cac chuoi thoai
str_new <- gsub(str_find, str_replace, str1)

In [29]: print(paste("str4 (index ->):", str4))
print(paste("str5 (->index):", str5))

print(paste("Length of str1:", nchar(str1),
           ", Length of str2:", nchar(str2),
           ", Length of str3:", nchar(str3)))
print(paste("After replace:", str_new))

[1] "str4 (index ->): programming language"
[1] "str5 (->index): This is R"
[1] "Length of str1: 30 , Length of str2: 16 , Length of str3: 20"
[1] "After replace: This is R programming LANGUAGE"
```

Exercise 6: Khoảng cách giữa 2 ngày

```
In [30]: date_1 <- readline(prompt = "Input date: dd-mm-YYYY:")
date_1 <- as.Date(date_1, format("%d-%m-%Y"))

Input date: dd-mm-YYYY:01-03-2020

In [31]: now_date <- Sys.Date()
distance <- now_date - date_1

In [32]: print(paste("Input date:", date_1))
print(paste("Now:", now_date))
print(paste("Distance between", now_date, "and", date_1, "is", distance))

[1] "Input date: 2020-03-01"
[1] "Now: 2020-03-03"
[1] "Distance between 2020-03-03 and 2020-03-01 is 2"
```

Exercise 7: Bảy ngày tiếp theo

```
In [33]: date_str <- readline(prompt = "Input your date dd-mm-YYYY:")
date_1 <- as.Date(date_str, format("%d-%m-%Y"))

Input your date dd-mm-YYYY:03-03-2020
```

```
In [34]: if(is.na(date_1)){
    print(paste(date_str, "Can not create a date"))
}else{
    print(paste(date_1, "is", weekdays.Date(date_1)))
    seven_days <- seq(date_1, length = 7, by = 1)
    print(paste("7 continuing days:",toString(seven_days)))
}

[1] "2020-03-03 is Tuesday"
[1] "7 continuing days: 2020-03-03, 2020-03-04, 2020-03-05, 2020-03-06, 2020-03-07, 2020-03-08, 2020-03-09"
```

