PROG103: Branches and Loops

Conditions in R

MARINCS 100B | Intro to Marine Data Science | Winter 2025

Key concepts

logical vector

comparisons $\longrightarrow$ conditions

combine comparisons using: and, or, not

What you already know

ocean <- c(Atlantic, Pacific, Indian)

perc_area <- c(0.17, 0.32, 0.14)

$$\text{ocean}\,\overbrace{[\text{perc\_area } \underbrace{>}_{\text{comparison}} 0.25]}^{\text{condition}}$$

Logical vectors

TRUE FALSE

```
x <- 1:3

x > 2   →  c(FALSE, FALSE, TRUE)
```

Comparisons

`==` equality

`>, >=` greater

`<, <=` less

`!=` not equal

`%in%` = inoperator

```
2^2 == 4
output: TRUE
```

```
"Apple" > "Banana"
output: FALSE
```

```
3 %in% c(1,3,5)
output: TRUE
```

Combining comparisons

&	|	!
and	or	not

```
x <- 1:3

x > 1 & x != 3
```

output: FALSE TRUE FALSE

Recap

logical vectors: TRUE FALSE

comparisons $\rightarrow$ conditions

e.g. `==` `>` `%in%`

combine comparisons logically

`a & b`	`a	b`	`!a`
and	or	not

New vocabulary and lingering questions

New vocabulary

logical vector: a vector that outputs a TRUE or FALSE response

comparisons: symbols used to find relationships between inputs + give you conditions

Lingering questions

Exercises

See section “Conditions in R” in prog103exercises.R

PROG103: Branches and Loops

Making choices with `if`, `else`, and `else if`

MARINCS 100B | Intro to Marine Data Science | Winter 2025

Key concepts

if   else if   else

allow code to respond to conditions

Syntax: how it's written

```
if (cond) {
    do_something
} else if (cond2) {
    do_something_else
} else {
    do_third_thing
}
```

if begins structure

else if offers specific alts.

else offers general alts.

Demo in R

Recap

if, else if, else :

allow code to respond to conditions

New vocabulary and lingering questions

New vocabulary

if: tells code to read for something specific and if found to be true to do anothur function.

else: used to broadly scan for a condition

Lingering questions

Exercises

See section “Making choices with if, else, and else if” in prog103exercises.R

PROG103: Branches and Loops

Repeating yourself with vectorized functions

MARINCS 100B | Intro to Marine Data Science | Winter 2025

Key concepts

Many ways to repeat yourself in R

Vectorized operations are the simplest

What you already know

```
x <- c(1,4,9,16)

x - 2          -1   2   7   14

sqrt(x)         1   2   3   4
```

Repitition is implied

Demo in R

if you write a function using only vectorized operations that function will also be vectorized

Recap

Many ways to repeat yourself in R

Vectorization is the simplest, use whenever possible

New vocabulary and lingering questions

New vocabulary

vectorized fonctions: will act with all the given elements to create output

Lingering questions

Exercises

See section “Repeating yourself with vectorized functions” in prog103exercises.R

PROG103: Branches and Loops

Repeating yourself with `for` loops

MARINCS 100B | Intro to Marine Data Science | Winter 2025

Key concepts

Vectorized sometimes isn't enough

When we need more controls, we use "For" loops

What they look like

```
For (item in collection) {
    do_something(item)
}
```

identify our collection

name our iterator

write the body; do something with iterator

What's an iterator?

2 forms:

elements themselves:

```
collection = LETTERS
for (L in LETTERS) {
  ... L = "A"
  ... L = "B"
}
```

indices of collection: useful for mult. vectors

```
for (i in 1:length (LETTERS)) {
  do_something(LETTERS[i])
}
```

Demo in R

Recap

Vectorization is sometimes insufficent

"For" loops are more customizable, but more work

New vocabulary and lingering questions

New vocabulary

Loops: allow the coder to specify what the code should search for the given elements.

Lingering questions

What does i or a stand for?

The "i" is just a place holder we can use any variable

Exercises

See section “~~Repeating yourself with vectorized functions~~” in prog103exercises.R

Repeating yourself with for loops