PROG103: Branches and Loops

Conditions in R

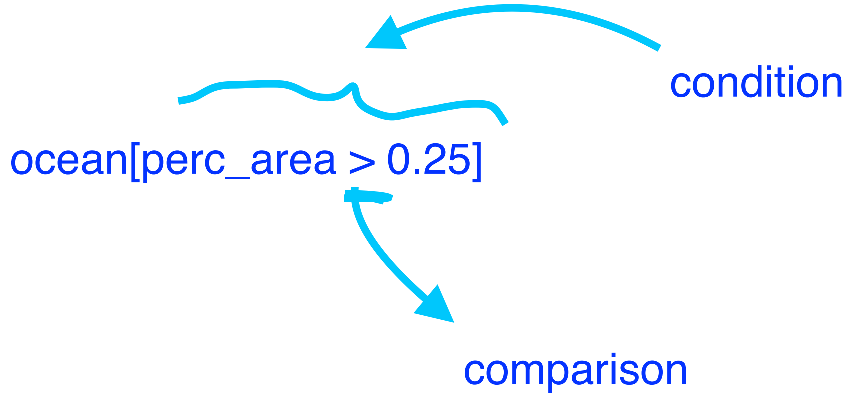
MARINCS 100B | Intro to Marine Data Science | Winter 2025

Key concepts

1) logical vector
2) comparisons -> conditions
3) combine comparisons using and, or, not

What you already know

```
ocean <- c(“Atlantic”, “Pacific”, “Indian”)
      perc_area <- c(0.17, 0.32, 0.14)
```



```
ocean[perc_area > 0.25]
```

condition

comparison

Logical vectors

TRUE FALSE ✓

True True “TRUE” ✗

```
x <- 1:3
x>2 -> c(FALSE, FALSE, TRUE)
```

Comparisons

== equality

>, >= greater

<, <= less

!= not equal

%in%

3 %in% c(1,3,5) -> TRUE
(is this value contained in that?)

2^2 == 4
TRUE

“Apple” > “Banana”
FALSE
(alphabetical)

Combining comparisons

(vertical line)

&
and

|
or

!
not

```
x <- 1:3

x > 1 & x != 2          1 2 3

    FALSE TRUE FALSE
```

Recap

1 - logical vectors TRUE FALSE
2 - comparisons build up our conditions
e.g. == > %in%
3 - combine comparisons logically
a & b etc

New vocabulary and lingering questions

New vocabulary

conditions

Lingering questions

how to make the vertical line for “or”

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

1. 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 the structure
else if offers specific alternatives
else offers general alternatives

Demo in R

Recap

if, else if, and else allow code to respond to conditions

New vocabulary and lingering questions

New vocabulary	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

there are 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)
```

no need to specify repetition because it’s 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

New vocabulary and lingering questions

New vocabulary

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

vectorization sometimes isn’t enough

when we need more control, we can use For loops

What they look like

```
for (item in a collection) {
    do_something(item)
}
```

need to ID our collection, name our iterator, write the body

What’s an iterator?

2 forms

elements themselves

collection = LETTERS (example)

```
for (L in LETTERS) {
… L = “A”
L = “B”
...
}
```

indices of the collection

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

useful for multiple vectors

Demo in R

Recap

vectorization is sometimes insufficient

For loops are more customizable but they require more work

New vocabulary and lingering questions

New vocabulary	Lingering questions

Exercises

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