Programming Logic for Non-Programmers

Kat Koziar and Stephanie Labou

2025-02-02

Contents

1 About **5**

1.1 Keeping this below for easy reference while we get used to the bookdown format 6

2 Introduction **7**

2.1 Building a Mental Model 7

3 Algorithms **9**

3.1 Breakout Activity 9

4 Loops **11**

4.1 Activity - loop trace 11

4.2 Other Types of Loops 11

4.3 math example 11

5 Applications **13**

5.1 Example one 13

5.2 Example two 13

6 Final Words **15**

7 BD Demo Introduction **17**

8 BD Demo Methods **19**

8.1 math example 19

Chapter 1

About

Bookdown reference: https://bookdown.org/yihui/bookdown/usage.html

```
BEGIN pitch()
```

Have you ever wondered how some of your colleagues can look at a computer programming script, with little prior knowledge of the language, and not only read it, but help fix the code? It's not because they know all programming languages, but because most programming languages use the same concepts and logic.

```
STRUCTURE(workshop)
```

In this interactive workshop, attendees will gain hands-on experience to understand and interpret programming logic. We will cover fundamental topics in programming including: conditional statements, loops, order of operations and logical flow, functions and arguments, and data types. Attendees will practice formulating programming arguments to accomplish common tasks, such as subsetting data based on a set of conditions.

```
WHERE prior_experience == FALSE
```

No coding experience required! Programming logic is transferable across specific languages, so learners will focus on concepts, rather than specific syntax from a specific language. Attendees will learn to interpret programming logic and build confidence to apply their understanding to various programming languages they may encounter.

```
FOR (x in example1:example5) {annotate(x)}
```

To provide real world examples of programming logic in practice, the workshop will integrate hands-on work time with examples of sample code written in R,

Python, SQL, Stata, and other languages. Attendees will practice annotating code in human understandable language and discuss the process, and any pitfalls, with their peers and the instructors.

```
IF attendee_need == "learn_programming_logic": print("register
for this workshop!")
```

1.1 Keeping this below for easy reference while we get used to the bookdown format

Prerequisites

This is a *sample* book written in **Markdown**. You can use anything that Pandoc's Markdown supports, e.g., a math equation $a^2 + b^2 = c^2$.

The **bookdown** package can be installed from CRAN or Github:

```
install.packages("bookdown")
# or the development version
# devtools::install_github("rstudio/bookdown")
```

Remember each Rmd file contains one and only one chapter, and a chapter is defined by the first-level heading `#`.

To compile this example to PDF, you need XeLaTeX. You are recommended to install TinyTeX (which includes XeLaTeX): https://yihui.org/tinytex/.

Chapter 2

Introduction

New intro for Programming Logic

2.1 Building a Mental Model

Chapter 3

Algorithms

overview of algorithms and computer code/scripting

3.1 Breakout Activity

Chapter 4

Loops

Drawing from Introduction to Programming Logic(Lynne O'Hanlon, 2000) Start with `for` loop as first function term "Populate an array from a for loop" as an early example; pg 376 blank table with good early exercise. Side note about infinite loop, importance of settling bounds What happens if you don't tell loop to increase? Examples of `for` loops in multiple languages (Python, C, R)? To decide: all theory and then all practice, or theory/practice/theory/practice? Also writing out result of each iteration otherwise you'll only get the last result (a common problem)

4.1 Activity - loop trace

4.2 Other Types of Loops

while(check at beginning) do-while(check at end) until(check at end)

4.3 math example

You can also use math in footnotes like this[1].

We will approximate standard error to 0.027

[1] where we mention $p = \frac{a}{b}$

Chapter 5

Applications

Some *significant* applications are demonstrated in this chapter.

5.1 Example one

5.2 Example two

Chapter 6

Final Words

We have finished a nice book.

Chapter 7

BD Demo Introduction

You can label chapter and section titles using `{#label}` after them, e.g., we can reference Chapter 2. If you do not manually label them, there will be automatic labels anyway, e.g., Chapter **??**.

Figures and tables with captions will be placed in `figure` and `table` environments, respectively.

```
par(mar = c(4, 4, .1, .1))
plot(pressure, type = 'b', pch = 19)
```

Reference a figure by its code chunk label with the `fig:` prefix, e.g., see Figure 7.1. Similarly, you can reference tables generated from `knitr::kable()`, e.g., see Table 7.1.

```
knitr::kable(
  head(iris, 20), caption = 'Here is a nice table!',
  booktabs = TRUE
)
```

You can write citations, too. For example, we are using the **bookdown** package (Xie, 2025) in this sample book, which was built on top of R Markdown and **knitr** (Xie, 2015).

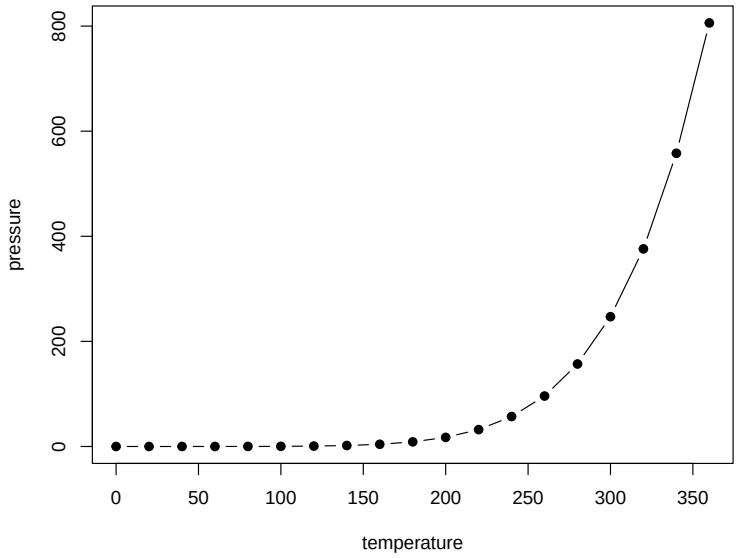

Figure 7.1: Here is a nice figure!

Table 7.1: Here is a nice table!

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa

Chapter 8

BD Demo Methods

We describe our methods in this chapter.

Math can be added in body using usual syntax like this

8.1 math example

p is unknown but expected to be around 1/3. Standard error will be approximated

$$SE = \sqrt{\frac{p(1-p)}{n}} \approx \sqrt{\frac{1/3(1-1/3)}{300}} = 0.027$$

You can also use math in footnotes like this[1].

We will approximate standard error to 0.027[2]

[1] where we mention $p = \frac{a}{b}$

[2] p is unknown but expected to be around 1/3. Standard error will be approximated

$$SE = \sqrt{\frac{p(1-p)}{n}} \approx \sqrt{\frac{1/3(1-1/3)}{300}} = 0.027$$

Bibliography

Lynne O'Hanlon (2000). *Introduction to computer programming logic*. Kendall/Hunt Pub. Co.

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2025). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.42.