Data Manipulation with the `tidyverse`

HES 505 Fall 2025: Session 5

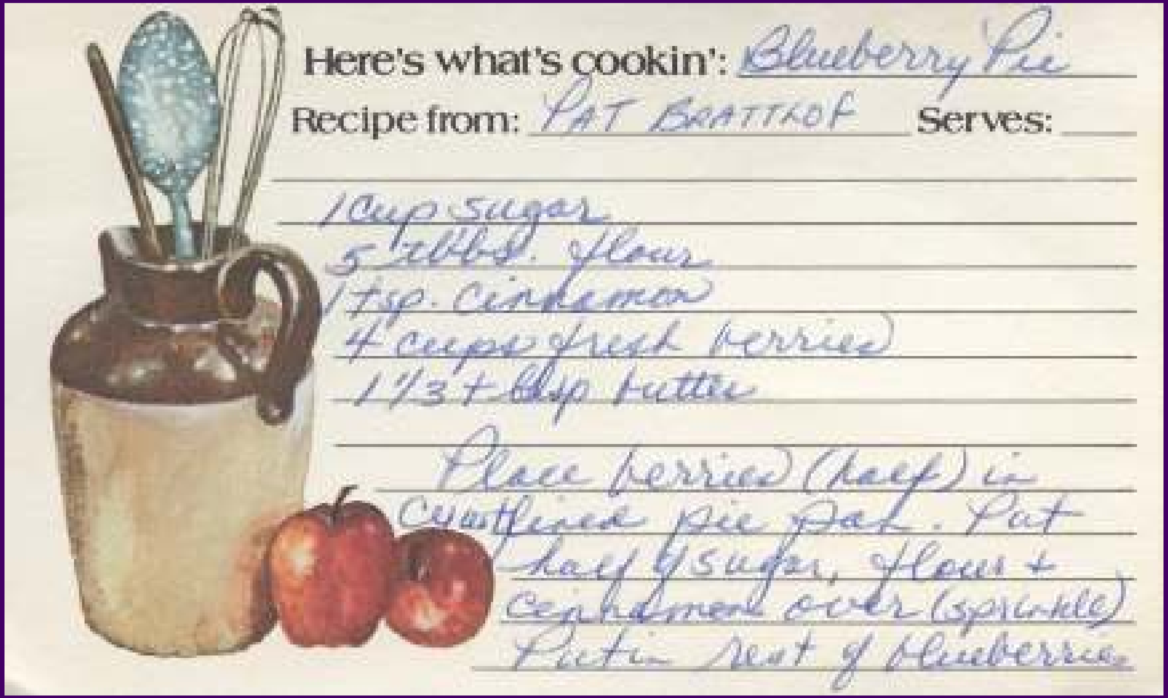
Matt Williamson

What is literate programming?

- Documentation containing code (not vice versa!)
- Direct connection between code and explanation
- Convey meaning to humans rather than telling computer what to do!

What is a script?

- Clarity
- Automation
- Minimal Documentation



Here's what's cookin': Blueberry Pie
Recipe from: PAT BRATTKOF Serves:

1 cup sugar
5 tbls. flour
1 tsp. cinnamon
4 cups fresh berries
1 1/3 tbsp butter

Place berries (half) in crust lined pie pan. Put half of sugar, flour & cinnamon over (sprinkle) Put in rest of blueberries

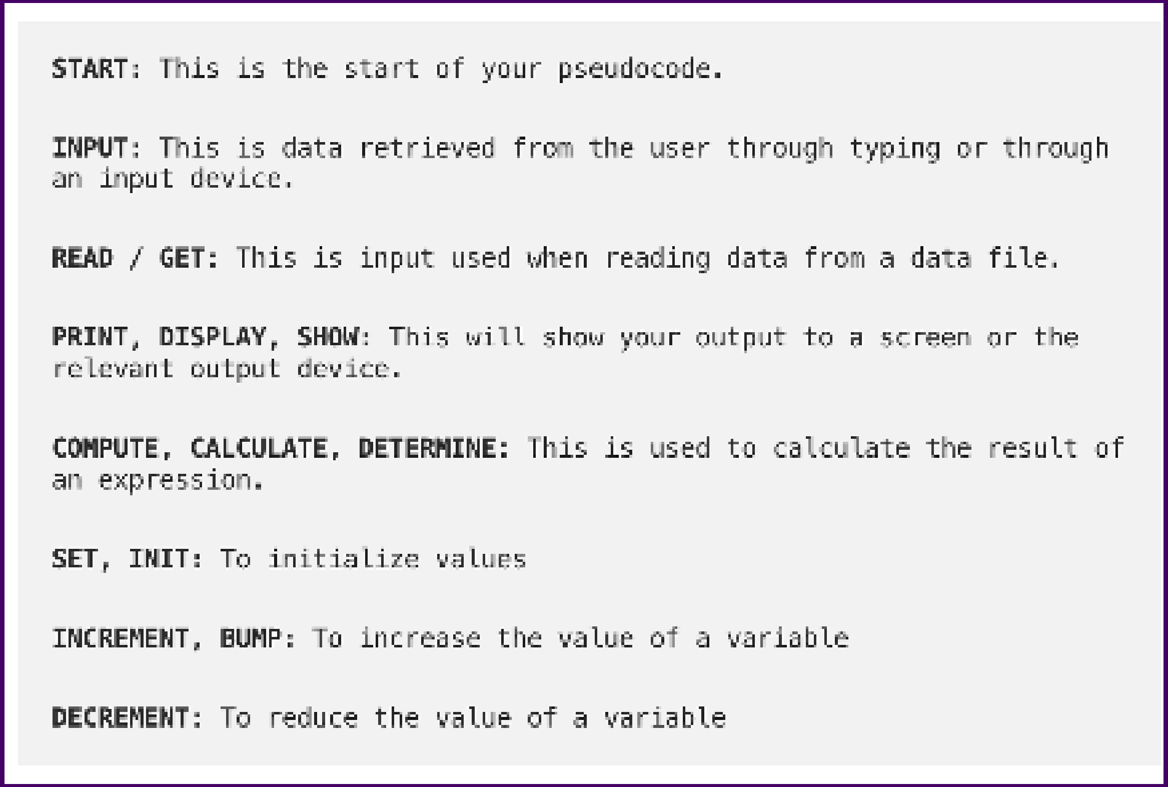
If it’s for a computer, it’s great for a script. If it’s for a human, we need more.

Pseudocode

- An informal way of writing the ‘logic’ of your program
- Balance between readability and precision
- Avoid *syntactic drift*

Writing pseudocode

- Focus on statements
- Mathematical operations
- Conditionals
- Iteration
- Exceptions



```
START: This is the start of your pseudocode.

INPUT: This is data retrieved from the user through typing or through
an input device.

READ / GET: This is input used when reading data from a data file.

PRINT, DISPLAY, SHOW: This will show your output to a screen or the
relevant output device.

COMPUTE, CALCULATE, DETERMINE: This is used to calculate the result of
an expression.

SET, INIT: To initialize values

INCREMENT, BUMP: To increase the value of a variable

DECREMENT: To reduce the value of a variable
```

Pseudocode

```
Start function
Input information
Logical test: if TRUE
  (what to do if TRUE)
else
  (what to do if FALSE)
End function
```

Why care about style?

- Easier for you, future you, and others to read
- Easier to get help

Introducing the `tidyverse`

What? Why?

- A group of task-specific packages built on shared grammar
- Reduced dependencies on other packages
- Consistent logic and shared style
- Allows us to code “out loud”

`dplyr` and a grammar for data transformation

- functions as verbs
- functions work on and return data frames
- first argument is always a data frame
- subsequent arguments say what to do with it

Basic structure: Verbs

Basic structure: Helpers

Can be combined with `select` to apply verbs to multiple columns

- `starts_with()`: Starts with a prefix
- `contains()`: Contains a literal string
- `one_of()`: Matches variable names in a character vector
- `everything()`: Matches all variables
- `last_col()`: Select last variable, possibly with an offset

Basic structure: Pipes

In programming, a pipe is a technique for passing information from one process to another.

- You can think about the following sequence of actions - find keys, unlock car, start car, drive to work, park.
- Expressed as a set of nested functions in R pseudocode this would look like:

```r
park(drive(start_car(find("keys")), to = "work"))
```

Basic structure: Pipes

- You can think about the following sequence of actions - find keys, unlock car, start car, drive to work, park.
- Writing it out using pipes give it a more natural (and easier to read) structure:

```r
find("keys") %>%
  start_car() %>%
  drive(to = "work") %>%
  park()
```

Extensions

- `tidyr` for cleaning and reshaping data
- `dplyr::xxxx_join` for combining data

Scripts and your workflow

- Analysis = many functions, many scripts
- Literate document (Quarto) focuses on communication
- Integrates inputs and outputs alongside code
- Focus is on research questions, decisions, and interpretation