DataFrames

INTRODUCTION TO JULIA

James Fulton
Climate informatics researcher

Tabular data

	Day	Distance	Time	Raining
1	Wednesday	2000	14.99	true
2	Monday	5000	31.68	false
3	Thursday	3500	22.02	true
4	Tuesday	3000	17.25	true
5	Thursday	4500	25.47	false
6	Monday	5000	30.77	true

Tabular data

	Day	Distance	Time	Raining
1	Wednesday	2000	14.99	true
2	Monday	5000	31.68	false
3	Thursday	3500	22.02	true
4	Tuesday	3000	17.25	true
5	Thursday	4500	25.47	false
6	Monday	5000	30.77	true
	String	**Int**	**Float**	**Bool**

DataFrames

```
using DataFrames
```

```
# Create DataFrame
df = DataFrames.DataFrame(




)
```

DataFrames

```
using DataFrames
```

```
# Create DataFrame
df = DataFrame(
    day = ["Wednesday", "Monday", "Thursday", "Tuesday", "Thursday", "Monday"]
    distance = [2000, 5000, 3500, 3000, 4500, 5000]
    time = [14.99, 31.68, 22.02, 17.25, 25.47, 30.77]
    raining = [true, false, true, true, false, true]
)
```

DataFrames

```
using DataFrames
```

```
# Create DataFrame
df = DataFrame(
    day = ["Wednesday", "Monday", "Thursday", "Tuesday", "Thursday", "Monday"],
    distance = [2000, 5000, 3500, 3000, 4500, 5000],
    time = [14.99, 31.68, 22.02, 17.25, 25.47, 30.77],
    raining = [true, false, true, true, false, true],
)
```

DataFrames

```
println(df)
```

```
6×4 DataFrame
 Row | day        distance  time     raining
     | String     Int64     Float64  Bool
_____|_____________________________________
   1 | Wednesday      2000    14.99     true
   2 | Monday         5000    31.68    false
   3 | Thursday       3500    22.02     true
   4 | Tuesday        3000    17.25     true
   5 | Thursday       4500    25.47    false
   6 | Monday         5000    30.77     true
```

CSV files

- Comma separated variable
- Common format for tabular data

Inside `run.csv`:

```
day,distance,time,raining
Wednesday,2000,14.99,true
Monday,5000,31.68,false
Thursday,3500,22.02,true
Tuesday,3000,17.25,true
Thursday,4500,25.47,false
Monday,5000,30.77,true
```

Loading CSV files

```
using CSV
```

```
# Load the run data
file = CSV.File("run.csv")

# Convert the CSV file into the DataFrame
df = DataFrame(file)
```

- Cannot use `File("run.csv")` only `CSV.File("run.csv")`

Printing DataFrames

```
# Print the first 3 rows
println(first(df, 3))
```

```
3×4 DataFrame
 Row | day        distance  time     raining
     | String     Int64     Float64  Bool
_____|_____________________________________
   1 | Wednesday      2000    14.99     true
   2 | Monday         5000    31.68    false
   3 | Thursday       3500    22.02     true
```

Basic properties of DataFrames

```
# Print column names
println(names(df))
```

```
["day", "distance", "time", "raining"]
```

```
# Print number of rows and columns
println(size(df))
```

```
(6, 4)
```

Let's practice!

INTRODUCTION TO JULIA

Sorting and slicing data

INTRODUCTION TO JULIA

James Fulton
Climate informatics researcher

Selecting an element from the DataFrame

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000       14.99     true
   2 | Monday       5000       31.68    false
   3 | Thursday     3500       22.02     true
   4 | Tuesday      3000       17.25     true
   5 | Thursday     4500       25.47    false
   6 | Monday       5000       30.77     true
```

```
# df[rownum, colnum]
t = df_run[6, 3]
println(t)
```

```
30.77
```

```
# df[rowrange, colnum]
ts = df_run[5:6, 3]
println(ts)
```

```
[25.47, 30.77]
```

Selecting an element from the DataFrame

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000       14.99     true
   2 | Monday       5000       31.68    false
   3 | Thursday     3500       22.02     true
   4 | Tuesday      3000       17.25     true
   5 | Thursday     4500       25.47    false
   6 | Monday       5000       30.77     true
```

```
# df[rownum, colnum]
t = df_run[6, 3]
println(t)
```

```
30.77
```

```
# df[rowrange, colnum]
ts = df_run[end-1:end, 3]
println(ts)
```

```
[25.47, 30.77]
```

Selecting a column

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000       14.99     true
   2 | Monday       5000       31.68    false
   3 | Thursday     3500       22.02     true
   4 | Tuesday      3000       17.25     true
   5 | Thursday     4500       25.47    false
   6 | Monday       5000       30.77     true
```

```
# df[:, colnum]
distances = df_run[:, 2]
```

```
println(distances)
```

```
[2000, 5000, 3500, 3000, 4500, 5000]
```

Selecting a column

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000       14.99     true
   2 | Monday       5000       31.68    false
   3 | Thursday     3500       22.02     true
   4 | Tuesday      3000       17.25     true
   5 | Thursday     4500       25.47    false
   6 | Monday       5000       30.77     true
```

```
# df[:, colnum]
distances = df_run[:, 2]
```

```
# df[:, "colname"]
distances = df_run[:, "distance"]
```

```
# df.colname
distances = df_run.distance
```

```
println(distances)
```

```
[2000, 5000, 3500, 3000, 4500, 5000]
```

Selecting an element from the DataFrame

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000       14.99     true
   2 | Monday       5000       31.68    false
   3 | Thursday     3500       22.02     true
   4 | Tuesday      3000       17.25     true
   5 | Thursday     4500       25.47    false
   6 | Monday       5000       30.77     true
```

```
# df[rownum, colnum]
d = df_run[6, 2]
```

```
# df[rownum, "colname"]
d = df_run[6, "distance"]
```

```
# df.colname[rownum]
d = df_run.distance[6]
```

```
println(d)
```

```
5000
```

Slicing multiple columns

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday  2000         14.99    true
   2 | Monday     5000         31.68    false
   3 | Thursday   3500         22.02    true
   4 | Tuesday    3000         17.25    true
   5 | Thursday   4500         25.47    false
   6 | Monday     5000         30.77    true
```

```
df_3cols = df_run[:, 1:3]
println(df_3cols)
```

```
6×3 DataFrame
 Row | day        distance     time
     | String     Int64        Float64
_____|_____________________________
   1 | Wednesday  2000         14.99
   2 | Monday     5000         31.68
   3 | Thursday   3500         22.02
   4 | Tuesday    3000         17.25
   5 | Thursday   4500         25.47
   6 | Monday     5000         30.77
```

Selecting rows

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000     14.99       true
   2 | Monday       5000     31.68      false
   3 | Thursday     3500     22.02       true
   4 | Tuesday      3000     17.25       true
   5 | Thursday     4500     25.47      false
   6 | Monday       5000     30.77       true
```

```
# df[rownum, :]
println(df_run[4, :])
```

```
DataFrameRow
 Row | day    distance     time     raining
_____|_____________________________________
   4 | Tuesday    3000     17.25       true
```

Selecting multiple rows

```
df_run = DataFrame(CSV.File("run.csv"))
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday  2000         14.99     true
   2 | Monday     5000         31.68     false
   3 | Thursday   3500         22.02     true
   4 | Tuesday    3000         17.25     true
   5 | Thursday   4500         25.47     false
   6 | Monday     5000         30.77     true
```

```
# df[rowrange, :]
println(df_run[2:4, :])
```

```
3×4 DataFrame
 Row | day        distance     time     raining
     | String     Int64        Float64  Bool
_____|_____________________________________
   1 | Monday     5000         31.68     false
   2 | Thursday   3500         22.02     true
   3 | Tuesday    3000         17.25     true
```

Sorting DataFrames

```
df_sort = sort(df_run, "time")
println(df_sort)
```

```
6×4 DataFrame
 Row | day       distance     time     raining
     | String    Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000     14.99      true
   2 | Tuesday      3000     17.25      true
   3 | Thursday     3500     22.02      true
   4 | Thursday     4500     25.47     false
   5 | Monday       5000     30.77      true
   6 | Monday       5000     31.68     false
```

```
df_sort = sort(df_run, "time", rev=true)
println(df_sort)
```

```
6×4 DataFrame
 Row | day       distance     time     raining
     | String    Int64        Float64  Bool
_____|_____________________________________
   1 | Monday       5000     31.68     false
   2 | Monday       5000     30.77      true
   3 | Thursday     4500     25.47     false
   4 | Thursday     3500     22.02      true
   5 | Tuesday      3000     17.25      true
   6 | Wednesday    2000     14.99      true
```

Cheat sheet

Select a column

- `df[:, "colname"]`
- `df[:, colnum]`
- `df.colname`

Select a row

`df[rownum, :]`

Select multiple columns

`df[:, colnum1:colnum2]`

Select multiple rows

`df[rownum1:rownum2, :]`

Select a single value

- `df[rownum, "colname"]`
- `df[rownum, colnum]`
- `df.colname[rownum]`

Sorting by column

Ascending order

`sort(df, "colname")`

Descending order

`sort(df, "colname", rev=true)`

Let's practice!

INTRODUCTION TO JULIA

Descriptive statistics

INTRODUCTION TO JULIA

James Fulton
Climate informatics researcher

datacamp

Describe function

```
# Summarize the runs DataFrame
println(describe(df_run))
```

```
4×7 DataFrame
 Row | variable  mean       min     median  max        nmissing  eltype
     | Symbol    Union…     Any     Union…  Any        Int64     DataType
_____|_____________________________________________________________________
   1 | day                  Monday          Wednesday          0  String
   2 | distance  3833.33    2000    4000.0  5000               0  Int64
   3 | time      23.6967    14.99   23.745  31.68              0  Float64
   4 | raining   0.666667   false   1.0     true               0  Bool
```

Summary statistics on columns

```
using statistics
```

Functions in `Statistics`:

- `mean()` - Calculate mean of array
- `median()` - Calculate median value of array
- `std()` - Calculate standard deviation of array values
- `var()` - Calculate variance of array values

```
# Calculate average of distance column
average_distance = mean(df_run[:, "distance"])
```

Other builtin summary functions

- `sum()` - Calculate sum of array
- `minimum()` - Calculate minimum value in array
- `maximum()` - Calculate maximum value in array

```
total_distance = sum(df_run[:, "distance"])         # Returns 23000

minimum_distance = minimum(df_run[:, "distance"])  # Returns 2000

maximum_distance = maximum(df_run[:, "distance"])  # Returns 5000
```

Column operations

For columns `a` and `b` of DataFrame `df`

Operation	Scalar example	Array example
Addition	`df.a .+ 1`	`df.a .+ df.b` or `df.a + df.b`
Subtraction	`df.a .- 1`	`df.a .- df.b` or `df.a - df.b`
Multiplication	`2 .* df.a` or `2 * df.a`	`df.a .* df.b`
Division	`df.a ./ 2` or `df.a / 2`	`df.a ./ df.b`

Calculating run speed

```
# Convert distances to kilometers
distance_km = df_run.distance ./ 1000
```

```
# Convert run times to hours
time_hr = df_run.time ./ 60
```

```
println(distance_km)
println(time_hr)
```

```
[2.0, 5.0, 3.5, 3.0, 4.5, 5.0]
[0.25, 0.53, 0.37, 0.29, 0.42, 0.51]
```

```
6×4 DataFrame
 Row | distance     time  ...
     |    Int64  Float64  ...
_____|_____________________  ...
   1 |     2000    14.99  ...
   2 |     5000    31.68  ...
   3 |     3500    22.02  ...
   4 |     3000    17.25  ...
   5 |     4500    25.47  ...
   6 |     5000    30.77  ...
```

Calculating run speed

```
# Convert distances to kilometers
distance_km = df_run.distance ./ 1000
```

```
# Convert run times to hours
time_hr = df_run.time ./ 60
```

```
# Run speed in km/hr
speeds = distance_km ./ time_hr

println(speeds)
```

```
[8.01, 9.47, 9.54, 10.43, 10.60, 9.75]
```

```
6×4 DataFrame
 Row | distance     time  ...
     |    Int64  Float64  ...
_____|_____________________  ...
   1 |     2000    14.99  ...
   2 |     5000    31.68  ...
   3 |     3500    22.02  ...
   4 |     3000    17.25  ...
   5 |     4500    25.47  ...
   6 |     5000    30.77  ...
```

Column assignment

```
# Assign run speeds to new column named "speed"
df_run[:, "speed"] = distance_km ./ time_hr
```

```
# Assign using dot form
df_run.speed = distance_km ./ time_hr
```

Column assignment

```
println(df_run)
```

```
6×4 DataFrame
 Row | day        distance     time     raining    speed
     | String     Int64        Float64  Bool       Float64
-----|-----------------------------------------------------
   1 | Wednesday    2000     14.99       true       8.01
   2 | Monday       5000     31.68      false       9.47
   3 | Thursday     3500     22.02       true       9.54
   4 | Tuesday      3000     17.25       true      10.43
   5 | Thursday     4500     25.47      false      10.60
   6 | Monday       5000     30.77       true       9.75
```

Let's practice!

INTRODUCTION TO JULIA

Filtering

INTRODUCTION TO JULIA

James Fulton
Climate informatics researcher

Example

```
6×4 DataFrame
 Row | day       distance     time     raining
     | String    Int64        Float64  Bool
_____|_____________________________________
   1 | Wednesday    2000     14.99      true
   2 | Monday       5000     31.68     false
   3 | Thursday     3500     22.02      true
   4 | Tuesday      3000     17.25      true
   5 | Thursday     4500     25.47     false
   6 | Monday       5000     30.77      true
```

The filter function

```
6×4 DataFrame
 Row | day       distance     time  raining
     | String    Int64  Float64     Bool
_____|_____________________________________
   1 | Wednesday   2000    14.99     true
   2 | Monday      5000    31.68    false
...
```

```
# Filter to Monday runs
df_monday = filter(row -> row.day=="Monday", df_run)
```

The filter function

```
println(df_monday)
```

```
6×4 DataFrame
 Row | day     distance     time  raining
     | String     Int64  Float64     Bool
_____|_____________________________________
   1 | Monday       5000    31.68    false
   2 | Monday       5000    30.77     true
```

Filtering on numerical columns

```
# Filter to shorter runs
df_short = filter(row -> row.distance<=3000, df_run)
println(df_short)
```

```
 Row | day       distance     time   raining
_____|_____________________________________
   1 | Wednesday   2000     14.99      true
   2 | Tuesday     3000     17.25      true
```

Filtering on boolean columns

```
# Filter to raining days
df_raining = filter(row -> row.raining, df_run)
println(df_raining)
```

```
 Row | day       distance     time   raining
_____|_____________________________________
   1 | Wednesday   2000     14.99      true
   2 | Thursday    3500     22.02      true
   3 | Tuesday     3000     17.25      true
   4 | Monday      5000     30.77      true
```

Filtering on all comparisons

- `row.col == b` filter to where `row.col` equals `b`
- `row.col != b` filter to where `row.col` does not equal `b`
- `row.col > b` filter to where `row.col` is greater than `b`
- `row.col >= b` filter to where `row.col` is greater than or equal to `b`
- `row.col < b` filter to where `row.col` is less than `b`
- `row.col <= b` filter to where `row.col` is less than or equal to `b`
- `row.col` filter to where `row.col` is `true`

Further analysis

```
# Distance run in rain
println(sum(df_raining.time))
```

```
13500
```

Let's practice!

INTRODUCTION TO JULIA

Final thoughts

INTRODUCTION TO JULIA

James Fulton
Climate informatics researcher

The exciting world of Julia

- Julia created 2012
- Expanding rapidly
- Stable version 1.0 released in 2018

Some things we've covered

- First steps
  - ```
    println(1+1)
    ```
  - ```
    x = 0
    ```
  - ```
    y = "fin"
    ```
- Arrays
  - ```
    x = [1,2,3,4,5]
    ```
  - ```
    x .* 2
    ```
  - ```
    x[2:end]
    ```
- Conditional statements

```
if enjoyment == 5
  println("Leave a rating?")
else
  println("Goodbye :)")
end
```

- Functions

```
function f(x)
  return x^2 + 2*x + 1
end
```

Some things we've covered

Broadcasting

```
x = [1,2,3]
y = f.(x)
```

Using Packages

```
import Statistics
using DataFrames
```

Multiple dispatch

```
function f(x::Int64)
    return x^2 + 2*x + 1
end

function f(x::Bool)
    return x
end
```

DataFrames

```
df = DataFrames(CSV.Files("run.csv"))
```

Congratulations!

INTRODUCTION TO JULIA