Julia for R Lovers 1

July 15, 2021

1 Julia for R-Lovers

Part one: Basics of working with R & Julia

1. Julia basics
2. RCall & using R in Julia
3. Communicating between R and Julia
4. (Some) essential Julia packages for data science

1.1 1. Julia basics

1.1.1 Basic code conventions

- #: comment
  - #= =#: multi-line comment
- basic math symbols as expected (+, -, /, *, ^, %)
- assign variables with =
- typeof(): gives variable type (Int64, Float64, String, etc.)
  - concrete types (Int64, Float64) belong to more abstract types (Float, Number, Any)
  - convert(Float64, 3): changes variable type
- can use unicode characters like emojis and mathematical symbols

[1]:
```
#comment

#= long comment
over multiple lines =#

1247 + 42
```

[1]:
```
1289
```

[2]:
```
number = 10
```

[2]:
```
10
```

[3]:
```
number_2 = 1.5
```

[3]:
```
1.5
```

[4]:
```
typeof(number)
```

[4]:
```
Int64
```

[5]:
```
typeof(number_2)
```

[5]:
```
Float64
```

[6]:
```
number + number_2
```

[6]:
```
11.5
```

[7]:
```
number * number_2
```

[7]:
```
15.0
```

[8]:
```
convert(Float64, number)
```

[8]:
```
10.0
```

1.1.2 Strings

- Define strings with " " or """ """
- Define characters (char) with ' '
- string(): convert to string and concatenate strings
- *: to concatenate strings
  - Ex: varname * varname

Note: using ; at end of line suppresses output in Julia

[9]:
```
hashtag = "rstats";
language = 'R';
```

[10]:
```
typeof(hashtag)
```

[10]:
```
String
```

[11]:
```
typeof(language)
```

[11]:
```
Char
```

[12]:
```
greeting = 'hello'
```

```
syntax: character literal contains multiple characters
```

```
Stacktrace:

 [1] top-level scope at In[12]:1

 [2] include_string(::Function, ::Module, ::String, ::String) at ./
↪loading.jl:1091
```

[13]:
```
hashtag * language
```

[13]:
```
"rstatsR"
```

1.1.3 Getting help

- Get function docs with ?

[14]:
```
?println()
```

[14]:
```
println([io::IO], xs...)
```

Print (using print) xs followed by a newline. If io is not supplied, prints to stdout.

2 Examples

```
julia> println("Hello, world")
Hello, world

julia> io = IOBuffer();

julia> println(io, "Hello, world")

julia> String(take!(io))
"Hello, world\n"
```

2.1 2. R-Call & using R in Julia

https://juliainterop.github.io/RCall.jl/latest/gettingstarted.html

- Support for dual-language workflow with R & Julia
- Comparable packages exist for working with Julia in R but are less efficient

2.1.1 R-Call essentials

- R" " or R""" """

[15]:
```
#using Pkg
#Pkg.add("RCall") #analogous to install.packages("package") in R

using RCall #analogous to library(package) in R
```

[16]:
```
R"nchar('Julia is great!')"
```

[16]:
```
RObject{IntSxp}
[1] 15
```

[17]:
```
R"round(rnorm(10,100,15), 1)"
```

[17]:
```
RObject{RealSxp}
 [1]  98.2  91.7  92.9 113.2  86.8  95.5 107.6  92.1  98.9  78.9
```

[18]:
```
R"""
fav_langs <- c("Julia", "R", "Python")

paste("I have", length(fav_langs), "favorite languages!")
"""
```

[18]:
```
RObject{StrSxp}
[1] "I have 3 favorite languages!"
```

[19]:
```
R"fav_langs"
```

[19]:
```
RObject{StrSxp}
[1] "Julia"  "R"      "Python"
```

- You can also load packages to R in the Julia environment and continue to use functions from them

[20]:
```
R"library(tidyverse)"
```

```
 Warning: RCall.jl: Warning: replacing previous import ‘vctrs::data_frame’ by
‘tibble::data_frame’ when loading ‘dplyr’
   Attaching packages                     tidyverse 1.3.0

   ggplot2 3.3.5      purrr   0.3.4
   tibble  3.1.2      dplyr   1.0.0
   tidyr   1.1.2      stringr 1.4.0
   readr   1.3.1      forcats 0.5.0
   Conflicts                      tidyverse_conflicts()

   dplyr::filter() masks stats::filter()
   dplyr::lag()    masks stats::lag()
 @ RCall /Users/kylamcconnell/.julia/packages/RCall/Qzssx/src/io.jl:160
```

[20]:
```
RObject{StrSxp}
 [1] "forcats"   "stringr"   "dplyr"     "purrr"     "readr"     "tidyr"
 [7] "tibble"    "ggplot2"   "tidyverse" "stats"     "graphics"  "grDevices"
[13] "utils"     "datasets"  "methods"   "base"
```

[21]:
```
R"ratings <- tibble(language = c('Julia', 'R', 'Python', 'Java'), rating = c(9,␣
↪10, 8, 1))"
```

[21]:
```
RObject{VecSxp}
# A tibble: 4 x 2
  language rating
  <chr>     <dbl>
1 Julia         9
2 R            10
3 Python        8
4 Java          1
```

[22]:
```
R"""
ratings %>%
   arrange(desc(rating))
"""
```

[22]:
```
RObject{VecSxp}
# A tibble: 4 x 2
  language rating
  <chr>     <dbl>
1 R            10
2 Julia         9
3 Python        8
4 Java          1
```

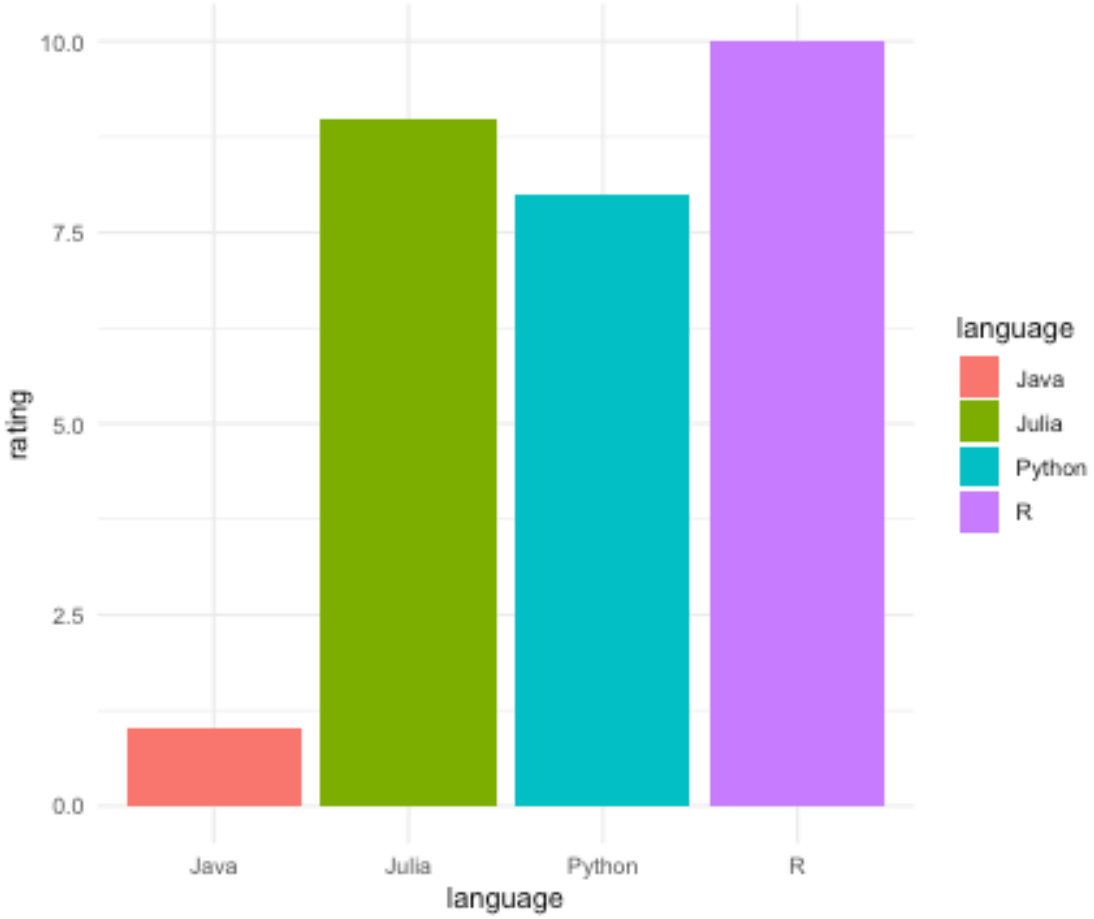
[23]:
```
R"""
ratings <- ratings %>%
   mutate(percent = rating * 10)

ratings
"""
```

[23]:
```
RObject{VecSxp}
# A tibble: 4 x 3
  language rating percent
  <chr>     <dbl>   <dbl>
1 Julia         9      90
2 R            10     100
```

```
3 Python        8       80
4 Java          1       10
```

[24]:
```
R"""
ggplot(ratings, aes(x = language, y = rating, fill = language)) +
  geom_col() +
  theme_minimal()
"""
```



[24]:
```
RObject{VecSxp}
```

2.2 3. Communiating between R and Julia

What about Julia?

- Create variables in Julia and transfer them to R with $

[25]:
```
greeting = "Hello there users of R!"
```

[25]:
```
"Hello there users of R!"
```

[26]:
```
R"toupper($greeting)"
```

[26]:
```
RObject{StrSxp}
[1] "HELLO THERE USERS OF R!"
```

[27]:
```
hrs = 3

R"mns <- 60 * $hrs"
```

[27]:
```
RObject{RealSxp}
[1] 180
```

[28]:
```
R"paste('There are', mns, 'minutes in', $hrs, 'hours.')"
```

[28]:
```
RObject{StrSxp}
[1] "There are 180 minutes in 3 hours."
```

- Can also substitute Julia operations into your R code with $

[29]:
```
findfirst("love", "We all love R!") #example function I know in Julia but not␣
↪R, returns index of substring
```

[29]:
```
8:11
```

[30]:
```
R"""
solitude <- 'Many years later, as he faced the firing squad, Colonel Aureliano␣
↪Buendía was to remember that distant afternoon when his father took him to␣
↪discover ice.'

$findfirst("afternoon", solitude)
"""
```

[30]:
```
RObject{IntSxp}
[1] 105 106 107 108 109 110 111 112 113
```

2.2.1 How to get objects from R back to Julia?

Macros: @rput & @rget - Macros = metacode: Julia code that alters Julia code - Move the variable from R to Julia (or vice versa) with the same name (and keep it there)

[31]:
```
@rget solitude

solitude
```

[31]:
```
"Many years later, as he faced the firing squad, Colonel Aureliano Buendía was
to remember that distant afternoon when his father took him to discover ice."
```

[32]:
```
solitude[105:113]
```

[32]:
```
"afternoon"
```

[33]:
```
R"dist <- round(rnorm(100, 25, 5), 1)"
```

[33]:
```
RObject{RealSxp}
  [1] 20.9 22.8 22.4 26.6 29.6 20.4 23.4 25.8 25.6 26.4 27.3 33.5 25.9 21.3 33.1
 [16] 25.3 20.6 30.3 17.3 31.5 24.6 21.6 26.4 24.7 22.5 26.9 20.9 35.2 20.8 22.0
 [31] 32.7 21.6 23.4 17.3 17.8 26.2 26.1 25.8 21.5 23.4 27.1 25.3 20.8 23.5 20.8
 [46] 25.9 25.3 29.9 24.3 23.3 32.9 35.2 24.2 29.8 20.7 29.1 25.1 30.8 30.9 19.4
 [61] 22.2 15.2 21.9 26.0 27.8 18.0 15.3 24.0 33.0 29.4 25.0 22.0 30.1 25.5 30.8
 [76] 24.3 24.5 22.3 21.0 19.0 28.9 25.6 24.2 25.6 19.1 18.3 26.7 21.9 28.7 16.9
 [91] 26.1 26.1 21.7 28.3 25.3 14.1 19.9 27.5 25.6 17.7
```

[34]:
```
@rget dist

length(dist)
```

[34]:
```
100
```

[35]:
```
dist = sqrt.(dist) #elementwise square roots
```

[35]:
```
100-element Array{Float64,1}:
 4.571651780264984
 4.774934554525329
 4.732863826479693
 5.157518783291051
 5.440588203494177
 4.516635916254486
 4.8373546489791295
 5.079370039680118
 5.059644256269407
 5.138093031466052
 5.224940191045253
 5.787918451395113
 5.089204259999788

 5.357238094391549
```

```
4.110960958218893
5.10881590977792
5.10881590977792
4.658325879540846
5.319774431308154
5.029910535983717
3.7549966711037173
4.460941604639093
5.244044240850758
5.059644256269407
4.207136793592526
```

[36]:
```
@rput dist;
```

[37]:
```
R"dist"
```

[37]:
```
RObject{RealSxp}
  [1] 4.571652 4.774935 4.732864 5.157519 5.440588 4.516636 4.837355 5.079370
  [9] 5.059644 5.138093 5.224940 5.787918 5.089204 4.615192 5.753260 5.029911
 [17] 4.538722 5.504544 4.159327 5.612486 4.959839 4.647580 5.138093 4.969909
 [25] 4.743416 5.186521 4.571652 5.932959 4.560702 4.690416 5.718391 4.647580
 [33] 4.837355 4.159327 4.219005 5.118594 5.108816 5.079370 4.636809 4.837355
 [41] 5.205766 5.029911 4.560702 4.847680 4.560702 5.089204 5.029911 5.468089
 [49] 4.929503 4.827007 5.735852 5.932959 4.919350 5.458938 4.549725 5.394442
 [57] 5.009990 5.549775 5.558777 4.404543 4.711688 3.898718 4.679744 5.099020
 [65] 5.272571 4.242641 3.911521 4.898979 5.744563 5.422177 5.000000 4.690416
 [73] 5.486347 5.049752 5.549775 4.929503 4.949747 4.722288 4.582576 4.358899
 [81] 5.375872 5.059644 4.919350 5.059644 4.370355 4.277850 5.167204 4.679744
 [89] 5.357238 4.110961 5.108816 5.108816 4.658326 5.319774 5.029911 3.754997
 [97] 4.460942 5.244044 5.059644 4.207137
```

2.3 4. (Some) essential Julia packages for data science

StatsBase: https://juliastats.org/StatsBase.jl/stable/ (or Statistics)

CSV: https://csv.juliadata.org/stable/

DataFrames: https://dataframes.juliadata.org/stable/

[38]:
```
#Pkg.add("StatsBase")
using StatsBase

mean(dist)
```

[38]:
```
4.939818612979829
```

[39]:
```
mean_and_std(dist)
```

[39]: (4.939818612979829, 0.4607767605781781)

[40]:
```
zscore(dist)
```

[40]:
```
100-element Array{Float64,1}:
 -0.799013457737914
 -0.35783935424088165
 -0.44914328196685055
  0.4724634333512258
  1.0867943728021068
 -0.9184115453095731
 -0.22237224783673798
  0.302861252215022
  0.26005140350225486
  0.43030472768945505
  0.6187846316460395
  1.8405872669253025
  0.32420395254420253

  0.9059039368390588
 -1.7988269497812648
  0.36676610293026696
  0.36676610293026696
 -0.6109091376174638
  0.8245984842021115
  0.19552184639442585
 -2.5713578531812438
 -1.0392820326698913
  0.6602451640338567
  0.26005140350225486
 -1.5901015026624636
```

[41]:
```
summarystats(dist)
```

[41]:
```
Summary Stats:
Length:         100
Missing Count:  0
Mean:           4.939819
Minimum:        3.754997
1st Quartile:   4.644887
Median:         4.984955
3rd Quartile:   5.191332
Maximum:        5.932959
```

2.4 DataFrames and CSV

[42]:
```
using CSV
using DataFrames

kart = CSV.read(download("https://raw.githubusercontent.com/rfordatascience/
 ↪tidytuesday/master/data/2021/2021-05-25/records.csv"));
```

[43]:
```
names(kart)
```

[43]:
```
9-element Array{String,1}:
 "track"
 "type"
 "shortcut"
 "player"
 "system_played"
 "date"
 "time_period"
 "time"
 "record_duration"
```

[44]:
```
describe(kart)
```

[44]:

	variable	mean	min	median	max	nunique	nmissing	
	Symbol	Union…	Any	Union…	Any	Union…	Nothing	
1	track		Banshee Boardwalk		Yoshi Valley	16		…
2	type		Single Lap		Three Lap	2		…
3	shortcut		No		Yes	2		…
4	player		ABE		iMathII	65		…
5	system_played		NTSC		PAL	2		…
6	date		1997-02-15		2021-02-25	1096		…
7	time_period		14.59S		6M 9.67S	1577		…
8	time	90.6238	14.59	86.19	375.83			…
9	record_duration	220.751	0	51.0	3659			…

[45]:
```
first(kart, 5)
```

[45]:

	track	type	shortcut	player	system_played	date	time_period	
	String	String	String	String	String	Date…	String	
1	Luigi Raceway	Three Lap	No	Salam	NTSC	1997-02-15	2M 12.99S	…
2	Luigi Raceway	Three Lap	No	Booth	NTSC	1997-02-16	2M 9.99S	…
3	Luigi Raceway	Three Lap	No	Salam	NTSC	1997-02-16	2M 8.99S	…
4	Luigi Raceway	Three Lap	No	Salam	NTSC	1997-02-28	2M 6.99S	…
5	Luigi Raceway	Three Lap	No	Gregg G	NTSC	1997-03-07	2M 4.51S	…

[46]:
```
kart.track
```

[46]: 2334-element PooledArrays.PooledArray{String,UInt32,1,Array{UInt32,1}}:
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"
"Luigi Raceway"

"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"
"Rainbow Road"

- denote ``symbol'' with :
- used for columns in dataframes

[47]:
```
select(kart, Not(:time_period))
```

[47]:

	track	type	shortcut	player	system_played	date	time	
	String	String	String	String	String	Date…	Float64	
1	Luigi Raceway	Three Lap	No	Salam	NTSC	1997-02-15	132.99	…
2	Luigi Raceway	Three Lap	No	Booth	NTSC	1997-02-16	129.99	…
3	Luigi Raceway	Three Lap	No	Salam	NTSC	1997-02-16	128.99	…
4	Luigi Raceway	Three Lap	No	Salam	NTSC	1997-02-28	126.99	…
5	Luigi Raceway	Three Lap	No	Gregg G	NTSC	1997-03-07	124.51	…
6	Luigi Raceway	Three Lap	No	Rocky G	NTSC	1997-04-30	122.89	…
7	Luigi Raceway	Three Lap	No	Launspach	NTSC	1997-04-30	122.87	…
8	Luigi Raceway	Three Lap	No	Launspach	NTSC	1997-04-30	122.78	…
9	Luigi Raceway	Three Lap	No	Launspach	NTSC	1997-05-27	122.25	…
10	Luigi Raceway	Three Lap	No	Launspach	NTSC	1997-05-27	122.21	…
11	Luigi Raceway	Three Lap	No	Patrick Q	NTSC	1997-07-30	121.32	…
12	Luigi Raceway	Three Lap	No	Launspach	NTSC	1997-08-02	120.81	…
13	Luigi Raceway	Three Lap	No	Launspach	NTSC	1997-08-02	120.76	…
14	Luigi Raceway	Three Lap	No	Launspach	NTSC	1997-10-31	120.1	…
15	Luigi Raceway	Three Lap	No	Booth	NTSC	1998-03-12	120.04	…
16	Luigi Raceway	Three Lap	No	Booth	NTSC	1998-03-13	120.01	…
17	Luigi Raceway	Three Lap	No	Booth	NTSC	1998-05-26	119.96	…
18	Luigi Raceway	Three Lap	No	Booth	NTSC	1998-08-14	119.86	…
19	Luigi Raceway	Three Lap	No	Penev	PAL	1998-09-28	119.85	…
20	Luigi Raceway	Three Lap	No	Penev	PAL	1998-10-01	119.82	…
21	Luigi Raceway	Three Lap	No	Peter E	PAL	1998-10-02	119.76	…
22	Luigi Raceway	Three Lap	No	Penev	PAL	1998-10-24	119.64	…
23	Luigi Raceway	Three Lap	No	Booth	NTSC	1998-11-20	119.63	…
24	Luigi Raceway	Three Lap	No	Penev	PAL	1998-12-24	119.54	…
25	Luigi Raceway	Three Lap	No	Booth	NTSC	1999-01-03	119.49	…
26	Luigi Raceway	Three Lap	No	Penev	PAL	1998-12-28	119.48	…
27	Luigi Raceway	Three Lap	No	Penev	PAL	1998-12-31	119.39	…
28	Luigi Raceway	Three Lap	No	Penev	PAL	1999-05-17	119.33	…
29	Luigi Raceway	Three Lap	No	Zwartjes	PAL	1999-09-07	119.3	…
30	Luigi Raceway	Three Lap	No	Penev	PAL	1999-09-27	119.26	…
…	…	…	…	…	…	…	…	