## R Programming Cheat Sheet by skydlins via cheatography.com/194596/cs/40633/

Data Structures			
Vector	ordered array of elements of the same data type a< c(3,1,5)		
Vector Naming	a<-c("desks" = 1, "tables" = 3, "chairs" = 4)		
Vector Coercion	a<-c(TRUE, FALSE, TRUE) = 1 0 1		
	seq(1,9,2) and rep(c(2,3,4), 3)		
Vector Subsetting	materials <- c(wood = 17, cloth = 36, silver = 24, gold = 3)		
	materials[1] = wood = 17		
Matrix	vector of elements arranged in two dimensions		
	m1<-matrix(3:8,ncol=3,nr- ow=2)		
	m2<-3:8 and dim(m2)<-c(3,2)		
Factor	used to store categorical variables (numeric or character)		
	a<-c(0,1,0,0,1)		
	a.f<-factor(a,labels = c("M- ale","Female"))		
	a.f = Male Female Male Male Female		
gl() function	generate factors by specifying the pattern of their levels		
	gl(2,8,labels=c("male","fem- ale"))		
List	multiple types of elements ()list		
	Mike<-list(Name="Mike",Sal- ary=10000,Age=43,Childre- n=c("Tom","Lily","Alice"))		
#\$	is a convenient way to retrieve element by element name.		
str()	display the internal structure		
$\mathbf{C}$	By skydlins cheatography.com/skydlins/		

### Data Structures (cont)

c()	combine several lists into one
Array	multi-dimensional arrangement of
	data in a vector.

### Exploring Data

Missing Data	a
Causes	human error, system error, loopholes
Dealing	summary() - how much data is missing
missing catego- rical data	set a new category called "Unknown"
missing numerical data	assign mean value or assign a value based on its relationship to other related variables
Other Data Problems	data entry, logical errors, outdated, inconsistent

# Data Visualization Principles Simplify, Compare, Attend (Details), Explore (Visual), View diversely, Ask why, Be skeptical, Respond GGPlot2 (+) allows us to make complex and aesthetically pleasing plots quickly and intuitively (-) work exclusively with data tables Components data data table in the example plot

lala	uata table in the example plot
	is summarized.

Not published yet. Last updated 5th October, 2023. Page 1 of 4.

Data Visualization (cont)		
geometry scatter plot, histograms, smooth densities, q-q plots, and blocks plots.		
aesthetic mapping	x and y axis	
scale	range of x-axis and y-axis appear to be defined by the range of the data	
labels, title, l	egend,	
Creating a N	ew Plot	
ggplot() function	specify the graph's data component.	
df %>% ggplot()	associates the dataset with the plotting object	
geom_p- oint()	add a layer, assigning population to x and total to y	
aes()	recognizes variables from the data component	
geom_l- abel() and geom_t- ext()	functions to add text to the plot.	
Size Color	geom_point(size = 3, color = "blue")	
geom_histogram()		
geom_d- ensity()	create smooth densities	
Programming	g Structure and Functions	
Basic		

if-else	else use curly braces "{}					
if(boolean condition){ expressions } else{						
alternative expressions }						
any() (similar	returns TRUE if any of the					
to OR " ")	logicals are true					

# Sponsored by Readable.com Measure your website readability!

### R Programming Cheat Sheet by skydlins via cheatography.com/194596/cs/40633/

can be faster if we know that our output can use a atomic data type that takes up less

a specified function to all elements of a list recursively

take multiple vectors as inputs.

applies the specified FUNC to each group of an array, grouped based on levels of certain

grouping data by different fields

summarize the data with your own function for specific

split a dataframe into a list of data frames based on a factor

group data by multiple factors

x <- list(A=2,B=list(-1,3),C=list(-2,list(-5,6))) rapply(x, function(x){x^2}) #returns a vector

factors.

purposes data(murders) tapply(murders\$total,

tapply(murders\$total/murders\$population,

murders\$region, sum)

murders\$region, mean)

array.

**Programming Structure and Functions** 

memory space.

(cont)

rapply()

mapply() tapply()

Pivot

Table

split()

tapply()

Programming Structure and Functions (cont)		
z <- c(TRUE, TRUE, FALSE) any(z)	TRUE	
all() (similar to &)	returns TRUE if all of the logicals are true	
Basic Funct	ions	
my_function <- function(x){ operations that operate on x which is defined by user of function value of final line is returned }		
For Loops		
for (i in rang i, which is c values }	e of values){ operations that use hanging across the range of	
for (i in 1:5){ print(i) }	## [1] 1 ## [1] 2 ## [1] 3 ## [1] 4 ## [1] 5	
apply()	apply a function to the margin of a matrix or a dataframe	
apply(x, MA	RGIN, FUNC,)	
z <- cbind(A	x=1:3,B=4:6,C=7:9,D=10:12)	
apply(z,2,sum)		
lapply()	works on list or vector inputs instead of matrix/dataframe input.	
	returns a list of the same length as the given list or array.	
x <- list(A=1	:4, B=seq(0.1,1,by=0.1))	
lapply(x, mean)		
sapply()	wrapper of the lapply() function. It also takes in a list or vector, however it returns a vector instead of a list	
vapply()	performs exactly like lapply() except that we can specify the return value type from FUNC	

#### By skydlins

cheatography.com/skydlins/

Not published yet. Last updated 5th October, 2023. Page 2 of 4.

### Basic Data Wrangling

Data Frame	use the data.frame() function. elements in the same column should be of the same data type.
	name <- c("Anne"), age <- c(28), child <- c(FALSE)
	df <- data.frame(name, age, child)
Data Frame Naming	names(df) <- c("Name", "- Age", "Child")
Data Frame Structure	Data Frame in R is implem- ented as a list of vectors with an important restriction of equal length vectors.
	R stores the character data type as a factor instead
str()	prevents R from converting the characters to vectors
Data Frame Subsetting	"[]" and "[[]]" and "\$"
	df[3,2] #r3c2
c()	used to subset multiple portions of the Data Frame.
Data Frame Extension	adding new variables or observations to an existing Data Frame.
	height <- c(163, 177, 163, 162, 157)
	df\$height <- height
Sorting	sort(df\$age) #based on age

Sponsored by Readable.com Measure your website readability! https://readable.com

# R Programming Cheat Sheet by skydlins via cheatography.com/194596/cs/40633/

max(dfSage) #jetting the highest agemerge 2 right_join & left_joinImpo Via mData Frame indexingindex c- dfSheight > 171suffixadded to the column names from each data frame to make them unique in the result.indexData Frame indexingindex <- dfSheight > 171should be a vector with two elementsindexindex <- dfSheight > 171should be a vector with two elementsindexindex <- dfSheight > 171should be a vector with two elementsindexindex <- dfSage > 30 & older than 30 without children.index <- dfSage > 30 & olfschild == FALSEinner_join returns only the rows that have matching values in both data frame based on specified key columnsR-ba converture/ny.suffix = cit, dri- ver', "_constructor")Ibbrary(dplyr)inner_join mutate(df, bmi = weight/height*2*10000commo robid & bind_rows used for dffilter(f, bmi > 18.5 & bmi < 24.9)filter(fr, hame, height, weight, bmi) > 18.5 & bmi < 24.9)inter, join sheight, weight, bmi > 24.9)%>% filter(bmi > 18.5 & bmi < 24.9)chain these three functions together.via u sheight, weight, bmi > 24.9)%>% filter(chini > 18.5 & bmi < 24.9)chain these three functions together.via u sheight, weight, bmi > 24.9)%>% filter(bmi > 18.5 & bmi < 24.9)filter(filter(filter) 34.8 bmi < 24.9)via u sheight, weight, bmi > 24.9)	Basic Data Wrangling (cont)			Basic Data	Wrangling (cont)	Adva
which.max(dfSage) #index of the oldest person       on col         Data Frame Indexing       find specific cases in DF       suffix       added to the column names from each data frame to make them unique in the result.         index <- dfSheight > 171       should be a vector with two elements       should be a vector with two elements         graph       dfSname[index] #person who is taller: pete       index <- dfSage > 30 & dfSchild == FALSE       inner_join       returns only the rows that have matching values in both data frames based on specified key columns       R-ba         library(dplyr)       index <- dfSuge > 30 & dfSchild == FALSE       union       combine two or more data frames vertically, stacking them on top of each other.       R-ba         library(dplyr)       extend DF for row and col function       union       combine two or more data frame sectically, stacking them on top of each other.       QSV         filter(r)       subset rows       anti_join       filterig rows from the first data frame based on values that do not have matching values in tot df       via U         select(1)       health <- select(df, name, height, weight, bmi)       of the select(name, height, weight, bmi)       via U         %>%       chain these three functions together.       chain these three functions together.       via U         %>%       df %>% select(name, height, weight, bmi) %>% filter(bmi > 18.5 & bmi < 24.9)       via U		max(df\$age) #getting the highest age		merge 2 df based	right_join & left_join	<b>Impo</b> Via re
Data Frame Indexingfind specific cases in DFsum (index <-df\$height > 171should be a vector with two elementsindex <- df\$height > 171should be a vector with two elementsshould be a vector with two elementsindex <- df\$height > 171should be a vector with two elementsreadodf\$name[index] #person who is taller: peteright_join(driver_q2, constructor"), suffix = c["_driver", "_constructor"), suffix = c["_driver", "_constructor")readofinding those older than 30df\$child == FALSEinner_joinreturns only the rows that have matching values in both data frames based on specified key columnsR-ba convlibrary(dplyr)extend DF for row and col functionor df\$bmi <- df\$weight/df- \$height^2110000inter_joinfilterig rows from the first data frame based on values that do not pof each other.CSVfilter(1)subset rows the second data frame.via Lcommon on to pof each other.Via Lfilter(1)subset rows the second data frame.via Ltemp afilter(1)subset rows together.df *- **********************************		which.max(df\$age) #index		on col	added to the column names	
index <- df\$height > 171should be a vector with two elementssum(index) #number of people taller than the male averageright_join(driver_q2, constr- uctors, by = c("constructor") = " constructor"), suffiz = c("_dri- ver", "_constructor"))reade R-bafinding those older than 30 without children.index <- df\$age > 30 & df\$child == FALSEinner_joinreturns only the rows that have matching values in both data frames based on specified key columnsR-ba convfilter(n.inner_joincombine two or more data frames vertically, stacking them on top of each other.R-ba convfilter(n)extend DF for row and col functionunion filtering rows from the first data frame based on values that do not have matching values in the second data frame.CSVfilter(1)subset rowsanti_joinfiltering rows from the first data frame based on values that do not have matching values in the second data frame.Via Lfilter(1)subset rowscommon sheight^2*10000rbind & bind_rowsVia Lfilter(1)subset rowscommon used for dfvia JSONfilter(1)health <- select(df, name, height, weight, bmi)height, weight, bmi)%% 	Data Frame Indexing	find specific cases in DF		Sum	from each data frame to make them unique in the result.	
sum(index) #number of people taller than the male average       immetris       immetris         df\$name[index] #person who is taller: pete       immetris       iconstructor").suffix = c("_dri-ver", "_constructor")         finding those       index <- df\$age > 30 & df\$child == FALSE       immetris       reads         without       df\$child == FALSE       inner_join       returns only the rows that have matching values in both data frames based on specified key columns       reba         library(dplyr)       mutate()       extend DF for row and col function       union       combine two or more data frames based on values that do not have matching values in both data frame based on values that do not have matching values in the second data frame.       CSV         filter()       subset rows       anti_join       filtering rows from the first data frame based on values that do not have matching values in the second data frame.       Via L         select()       height.veight.brii)       common       rbid & bind_rows       %         filter(d, bmi > 18.5 & bmi < 24.9)		index <- df\$height > 171			should be a vector with two elements right_join(driver_q2, constr- uctors, by = c("constructor" = "-	
df\$name[index] #person who is taller: peteconstructor"), suffix = c("_dri- ver", "_constructor"))R-barfinding those older than 30 oldf\$child == FALSEinner_joinreturns only the rows that have matching values in both data frames based on specified key columnsR-barlibrary(dplyr) </td <td>sum(inde&gt; people tal</td> <td>sum(index) #number of people taller than the male average</td> <td></td> <td>read</td>	sum(inde> people tal	sum(index) #number of people taller than the male average				read
finding those older than 30 without children.index <- df\$age > 30 & df\$child == FALSE without children.inner_join returns only the rows that have matching values in both data frames based on specified key 		df\$name[index] #person who is taller: pete			constructor"),suffix = c("_dri- ver", "_constructor"))	R-ba
older than 30 without children.df\$child == FALSE frames based on specified key columnsR-ba convilibrary(dplyr)extend DF for row and col functionunioncombine two or more data 	finding those	index <- df\$age > 30 &		inner_join	returns only the rows that have matching values in both data	
library(dplyr)       union       combine two or more data frames vertically, stacking them on top of each other.       CSV         mutate()       extend DF for row and col function       df <- mutate(df, bmi = weight/height^2*10000)	older than 30 without children.	er than 30 df\$child == FALSE nout dren			frames based on specified key columns	R-ba
mutate() functionextend DF for row and colfunctiondf <- mutate(df, bmi = weight/height^2*10000)or df\$bmi <- df\$weight/df- \$height^2*10000filter()subset rowsfilter(df, bmi > 18.5 & bmi < 24.9)select()health <- select(df, name, height, weight, bmi)filter(health, bmi > 18.5 & bmi < 24.9)	library(dplyr)		unio	union	combine two or more data	CSV
Initiation       df <- mutate(df, bmi = weight/height^2*10000)	mutate()	extend DF for row and col			on top of each other.	
or df\$bmi <- df\$weight/df- \$height^2*10000       Via U         filter()       subset rows         filter(df, bmi > 18.5 & bmi < 24.9)       common rbind & bind_rows         select()       health <- select(df, name, height, weight, bmi)         filter(health, bmi > 18.5 & bmi < 24.9)	uncion	df <- mutate(df, bmi = weight/height^2*10000)		anti_join	filtering rows from the first data frame based on values that do not have matching values in	
filter()       subset rows       used for       a         filter(df, bmi > 18.5 & bmi < 24.9)		or df\$bmi <- df\$weight/df- \$beight^2*10000			the second data frame.	Via L
filter(df, bmi > 18.5 & bmi < 24.9)dfselect()health <- select(df, name, height, weight, bmi)	filter()	subset rows		common used for	rbind & bind_rows	&
select()health <- select(df, name, height, weight, bmi)Viafilter(health, bmi)ISONfilter(health, bmi > 18.5 & bmi < 24.9)		filter(df, bmi > 18.5 & bmi < 24.9)		df		temp
filter(health, bmi > 18.5 & bmi < 24.9)%>%chain these three functions together.df %>% select(name, height, weight, bmi) %>% filter(bmi > 18.5 & bmi < 24.9)	select()	health <- select(df, name, height, weight, bmi)				Via JSOI
%>%       chain these three functions together.         df %>% select(name, height, weight, bmi) %>% filter(bmi > 18.5 & bmi < 24.9)		filter(health, bmi > 18.5 & bmi < 24.9)				
df %>% select(name, height, weight, bmi) %>% filter(bmi > 18.5 & bmi < 24.9)	%>%	chain these three functions together.				
		df %>% select(name, height, weight, bmi) %>% filter(bmi > 18.5 & bmi < 24.9)				

### Advance Data Wrangling

Importing Data		
Via readr	read_csv: comma separated values	
	read_tsv: tab delimited separated values	
	read_delim: general text file format	
	head() function display it as a tibble.	
readxl	read_excel,xls,xlsx	
R-base	read.csv() and read.table() can be used without having to install any libraries	
R-base import function will automatically convert any character strings to factors		
CSV	widespread use in the data science community due to its efficiency at storing large amounts of data and also as it is platform agnostic.There is also no size limit with csv files.	
Via URL	read_csv(url)	
tempdir() & tempfile()	it is useful to have a temporary directory or filename auto generated to manage these URL imports	
Via JSON	provided via API, library(json- lite), fromJSON(url)	

#### By skydlins

cheatography.com/skydlins/

Not published yet. Last updated 5th October, 2023. Page 3 of 4. Sponsored by Readable.com Measure your website readability! https://readable.com

# R Programming Cheat Sheet by skydlins via cheatography.com/194596/cs/40633/

Advance Data Wrangling (cont)

Advance Data Wrangling (cont)		
Via XML	rawling a website, xmlParse("- books.xml")	
xmlRoot()	access the root node of the tree.	
xmlChi- ldren()	use the children nodes of the tree	
xmlToL- ist(data), xmlToD- ataFra- me(- books)	convert the XML file to list or data frame format	
Reshaping I	Data	
Wide to Tidy: gather()	convert the above wide data into tidy data	
country,y- ear,feart- ility	new_tidy_data <- wide_data %>% gather(year, fertility, '1960':'2015')	
Tidy to Wide: spread()	The first argument of the spread() function is to declare which variables are to be used as column names. While the second argument is to specify the variables used to fill out the cells.	
Separate and Unite		
separate()	requires the target column, the names for the new columns and the separator character.	
<pre>dat %&gt;% separate(key, c("year", "first_var- iable_name", "second_variable_name"), fill = "right")</pre>		
spread()		

# C

#### By skydlins

cheatography.com/skydlins/

dat %>% separate(key, c("year", "variable name"), extra = "merge") %>% spread(va- riable_name, value)			
unite()	first name & last name		
Combini	ng Data		
join()	combined so that matching rows are together		
Inner Join	eturns only the rows that have matching values in both tables		
Left Join	returns all the rows from the left table and the matching rows from the right table		
Full Join	all the rows from both tables, with NULL values in columns where there is no match in the other table		
Semi	keep the part of the first table for		

Semi keep the part of the first table for Join which we have information in the second table, but doesnt add the columns of the second.

#### Advance Data Wrangling (cont)

Anti Join	opposite of the semi_join()
	function. It allows us to keep
	the part of the first table for
	which we have NO information
	in the second table, but doesnt
	add the columns of the second.

#### Set Operators

1

Intersect: inds common elements shared among sets.	intersect(1:10, 6:15) = 6 7 8 9 10
Union: ombines sets into one, removing duplic- ates.	same with interse
Setequal	helps us check if two sets are the same regardless of order.
Setdiff	find the elements that are in one set (or vector) but not in another set.

Not published yet. Last updated 5th October, 2023. Page 4 of 4. Sponsored by Readable.com Measure your website readability! https://readable.com