

# An Introduction to Programming in R

Drew Schmidt

November 13, 2015



Notes

---

---

---

---

---

---

---

---

## About

### This tutorial:

This is not a “how to fit a linear model in R” tutorial.

Slides and exercises are available at:

<http://wrathematics.info/handouts/bas2015.html>

I try to stick to base R as much as possible, even when better alternatives exist.

Mostly independent; we can skip something that’s boring!

### Me:

I am a very productive R package developer.

I mostly do not show you the things I make.

My expertise is R in HPC (contact me if interested!)



Notes

---

---

---

---

---

---

---

---

## Introduction

What is R?

- 1 Introduction
  - Installing R
  - Resources and Advice
- 2 R Basics
- 3 Programming
- 4 Closing



Notes

---

---

---

---

---

---

---

---


Introduction

What is R?

# What is R?

1 Introduction

- Installing R
- Resources and Advice



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R

Notes

---

---

---

---

---

---

---


---

Introduction

What is R?

# What is R?

*lingua franca* for data analytics and statistical computing.  
Part programming language, part data analysis package.  
Dialect of S (Bell Labs).  
Syntax designed for data.



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 3/119

Notes

---

---

---

---

---

---

---

---

Introduction

What is R?

# Who uses R?





[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 4/119

Notes

---

---

---

---

---

---

---

---

Introduction
What is R?

Language Paradigms

[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)
Drew Schmidt
An Introduction to Programming in R
5/119

Notes

---

---

---

---

---

---

---

---

---

---

Introduction
What is R?

Very Popular for a DSL!

IEEE Spectrum's 2014 Ranking of Programming Languages

Language Rank	Types	Spectrum Ranking
1. Java		100.0
2. C		99.3
3. C++		95.5
4. Python		93.4
5. C#		92.4
6. PHP		84.7
7. Javascript		84.4
8. Ruby		78.8
9. R		74.2
10. MATLAB		72.9

See: <http://spectrum.ieee.org/static/interactive-the-top-programming-languages#index>

[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)
Drew Schmidt
An Introduction to Programming in R
6/119

Notes

---

---

---

---

---

---

---

---

---

---

Introduction
What is R?

At Build 2015 Microsoft CVP Joseph Sirosh called R the "language of data" and said "if there is a single language that you choose to learn today .. let it be R".

[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)
Drew Schmidt
An Introduction to Programming in R
7/119

Notes

---

---

---

---

---

---

---

---

---

---


IntroductionInstalling R

What is R?

1 Introduction

Installing R

Resources and Advice



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)Drew SchmidtAn Introduction to Programming in R

Notes

---

---

---

---

---

---


---

---

IntroductionInstalling R

Installing R

Go to <http://cran.r-project.org/> for a download (binary or source)



CRAN

[Mirrors](#)

[What's new?](#)

[Task Views](#)

[Search](#)

[About R](#)

[R Homepage](#)

[The R Journal](#)

[Software](#)

[R Sources](#)

[R Binaries](#)

[Packages](#)

[Other](#)

[Documentation](#)

[Manuals](#)

[FAQs](#)

[Contributed](#)

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2013-05-16, Masked Marvel): [R-3.0.1.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)Drew SchmidtAn Introduction to Programming in R8/119

Notes

---

---

---

---

---

---

---


---

IntroductionInstalling R

Installing R

Windows users should also install Rtools  
<http://cran.r-project.org/bin/windows/Rtools/>

For a complete set of installation instructions, see  
<http://www.r-pbd.org/install.html>



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)Drew SchmidtAn Introduction to Programming in R9/119

Notes

---

---

---

---

---

---

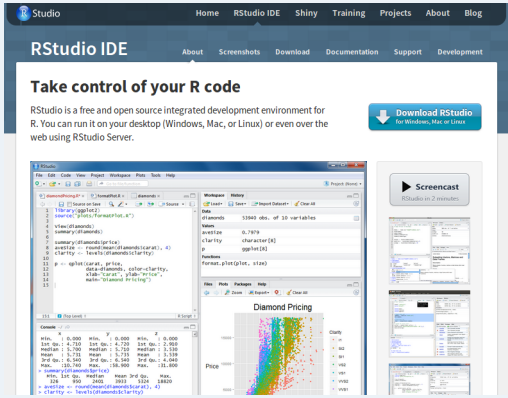
---

---

IntroductionInstalling R

Installing Rstudio

Go to <http://www.rstudio.com/ide/>



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)
Drew Schmidt
An Introduction to Programming in R
10/119

Notes

---

---

---

---

---

---

---

---

---

---

IntroductionResources and Advice

What is R?

1 Introduction

Installing R

Resources and Advice

[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)
Drew Schmidt
An Introduction to Programming in R

Notes

---

---

---

---

---

---

---

---

---

---

IntroductionResources and Advice

Important things we can't cover

R and version control

Developing R packages

Performance/profiling

Graphics/visualization

[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html)
Drew Schmidt
An Introduction to Programming in R
11/119

Notes

---

---

---

---

---

---

---

---

---

---

Introduction

Resources and Advice

R Resources

*The Art of R Programming* by Norm Matloff:  
<http://nostarch.com/artofr.htm>


*An Introduction to R* by Venables, Smith, and the R Core Team:  
<http://cran.r-project.org/doc/manuals/R-intro.pdf>

*The R Inferno* by Patrick Burns:  
[http://www.burns-stat.com/pages/Tutor/R\\_inferno.pdf](http://www.burns-stat.com/pages/Tutor/R_inferno.pdf)

Mathesaurus: <http://mathesaurus.sourceforge.net/>

R programming for those coming from other languages: [http://www.johndcook.com/R\\_language\\_for\\_programmers.html](http://www.johndcook.com/R_language_for_programmers.html)

*aRrgh: a newcomer's (angry) guide to R*, by Tim Smith and Kevin Ushey: <http://tim-smith.us/arrgh/>



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 12/119

Notes

---

---

---

---

---

---

---

---

Introduction


Resources and Advice

Tutorials

*R Programming*, Coursera course through Johns Hopkins  
<https://www.coursera.org/course/rprog>

*Statistics One* Coursera course through Princeton  
<https://www.coursera.org/course/stats1>

*High Performance Computing with R*  
<https://github.com/wrathematics/2015hpcRworkshop/blob/master/README.md>



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 13/119

Notes

---

---

---

---

---

---

---

---

Introduction

Resources and Advice

Other Invaluable Resources

*R Installation and Administration*:  
<http://cran.r-project.org/doc/manuals/R-admin.html>


*Task Views*: <http://cran.at.r-project.org/web/views>

*Writing R Extensions*:  
<http://cran.r-project.org/doc/manuals/R-exts.html>

Mailing list archives: <http://tolstoy.newcastle.edu.au/R/>

The [R] stackoverflow tag.

The #rstats hashtag on Twitter.



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 14/119

Notes

---

---

---

---

---

---

---

---

Introduction

Resources and Advice

Comments and Advice

R is part statistics package, part programming language.

R is slow; if you don't know what you're doing, it's *really* slow.

There is an R help mailing list. Use stackoverflow instead....

Learn to love the R help system.

If something appears broken in core R, it's (probably) not them, it's you.

Try to avoid "super functions".

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

15/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

1 Introduction

R Basics

2 R Basics

All\* About R Packages

I/O

Strings

Dates

Dealing with Dataframes

3 Programming

4 Closing

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

R Basics

R Basics

2 R Basics

All\* About R Packages

I/O

Strings

Dates

Dealing with Dataframes

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

Notes

---

---

---

---

---

---

---

---

---

---


R Basics

R Basics

Interacting with the R Terminal

The default thing for R to do is print/show:

```
1 1
2 1+1
3 print(1+1)
4 sum
5
6 + # ctrl+c to break
7 "+"
8 `+`
```



[wrthematics.info/handouts/bas2015.html](http://wrthematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 16/119

Notes

---

---

---

---

---

---

---

---

---


---

R Basics

R Basics

Arithmetic

```
1 7+4
2 7-4
3 7*4
4 7/4
5 7^4
6 7%4
```



[wrthematics.info/handouts/bas2015.html](http://wrthematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 17/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics


R Basics

Assignment

R naming rules can be quite lax. For all practical purposes:  
Start with a letter  
Should consist of letters (any case), numbers, . 's, and \_ 's  
Very strange things are possible, however...

```
1 rm(list=ls())

1 "&*" <- 3
2
3 "2" <- 1
4 `2` + 1
5 ## [1] 2
6
7 "\U0001f431" <- "even unicode"
8 cat(ls())
9 ## &*() 🐼 2
```



[wrthematics.info/handouts/bas2015.html](http://wrthematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 18/119

Notes

---

---

---

---

---

---

---

---

---

---




R Basics

R Basics

Assignment

```
1 # Local
2 myvar <- 1
3 myvar = 4
4
5 # Global
6 myvar <<- 2
7
8 # Whoever you want
9 assign(x="myvar", value=3)
```



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R19/119

Notes

---

---

---

---

---

---

---

---

---

---


R Basics

R Basics

Case and Spacing

R is case sensitive, but fairly lax about spacing:

```
1 x <- 1:5
2 x
3 X
4
5 1      + 1
6 sum(x)
7 sum ( x  )
8 s um(x)
```



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R20/119

Notes

---

---

---

---

---

---

---

---

---

---


R Basics

R Basics

Finding Help

R has its own manual system.  
Most of the answers to your questions lie within.  
Find help using ? or help(), or search across all help with ??.

```
1 ?sum
2 ??sum
3 help( "sum" )
4
5 ?+
6 ? "+"
```



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R21/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

R Basics

Ways to Run R

Interactive: typing commands into the console.  
Batch: Rscript, R CMD BATCH.  
From an IDE/GUI.  
Batch from a GUI: source()

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

22/119

Notes

R Basics

R Basics

On the note of directories...

Windows paths use / or  
, e.g. c:/myfile.r  
Get current working directory: getwd()  
Set current working directory: setwd("path/to/my/dir")

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

23/119

Notes

R Basics

All About R Packages

R Basics

2 R Basics

All\* About R Packages

I/O

Strings

Dates

Dealing with Dataframes

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

Notes

R Basics

All

About R Packages

R Extensions

R is great, but limited.  
Has a great package extension system.  
R doesn't do what you want? Make it!  
  
“...if I don't know how to fix it, I can hire somebody else to fix it for me.”  
— Matt Dowle, developer of the `data.table` package.

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

24/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

All

About R Packages

Packages

“R extensions”  
Comprehensive R Archive Network (CRAN).  
7474 packages (`nrow(available.packages())`)  
CRAN Task Views <http://cran.r-project.org/web/views/>

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

25/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

All

About R Packages

Terminology

A package is a collection of code usable by R.  
A library is a collection of packages.  
`library()` loads a package.  
Don't think too much about this...

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

26/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

All


About R Packages

Terminology

We call the extension a *package*.

*Packages* go into a *library*.

**This is dumb and confusing, but there's nothing we can ever do about it.**



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 27/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

All

About R Packages


Example Confusion

I wrote a library.

I put the library in a package.

I install the package ...into a library.

I load the package with `library()` ???



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 28/119

Notes

---

---

---

---

---

---

---

---

---


---


R Basics

All

About R Packages

\*BOOM\*





[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 29/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics


AllAbout R Packages

Installing R Packages

```
1 install.packages("devtools")
```

```
1 install.packages("devtools", lib="some/place/on/disk")
```

```
1 R CMD INSTALL devtools_1.6.tar.gz -l /some/place/on/disk
```



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

30/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

AllAbout R Packages


Repositories

This basically assumes you're using CRAN.

Lots of exciting development is happening outside of CRAN these days.

Other binary package repositories: Bioconductor, R-forge (Windows)

Other source repositories: GitHub, Bitbucket, ...



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

31/119

Notes

---

---

---

---

---

---

---

---


---


---


R Basics





AllAbout R Packages


GitHub Binaries?

**Karl Broman** @kwbroman · Sep 27  
.@hadleywickham @millerdl It would be great to have a service that would compile windows/mac binaries of #rstats pkgs on github.

**Hadley Wickham** @hadleywickham  
@kwbroman @millerdl agreed. Plans are in motion







wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

32/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

AllAbout R Packages


Installing Packages from Source

To install packages from source, you need some compilers:

**Windows:** Install [Rtools](#).

**Mac:** Install Xcode from the app store, possibly some other things from [here](#). If you need OpenMP, god help you.

**Linux and FreeBSD:** You're good to go.



[wrathematics.info/handouts/bas2015.html](#)Drew SchmidtAn Introduction to Programming in R33/119

Notes

---

---

---

---

---

---

---

---


R Basics

AllAbout R Packages

Installing R Packages

```
1 devtools::install_github("wrathematics/lineSampler")
```

Devtools Package Install Functions		
install_bitbucket	install_github	install_svn
install_deps	install_gitorious	install_url
install_git	install_local	install_version



[wrathematics.info/handouts/bas2015.html](#)Drew SchmidtAn Introduction to Programming in R34/119

Notes

---

---

---

---

---

---

---

---

R Basics

I/O

R Basics

2 R Basics


All\* About R Packages

I/O

Strings

Dates

Dealing with Dataframes



[wrathematics.info/handouts/bas2015.html](#)Drew SchmidtAn Introduction to Programming in R

Notes

---

---

---

---

---

---

---


---

R Basics

I/O

I/O

Input	Output
readcsv()	writecsv()
readtable()	writetable()
readBin()	writeBin()
readLines()	writeLines()
save()	load()
saveRDS()	readRDS()
scan()	



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R35/119

Notes

---

---

---

---

---

---

---

---

---


---

R Basics

I/O

"What about my really weird use-case?"

NUMEROUS R packages  
XML, json, databases, ...  
See also: *R Data Import/Export* manual <https://cran.r-project.org/doc/manuals/r-release/R-data.html>



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R36/119

Notes

---

---

---

---

---

---

---

---

---


---

R Basics

I/O

Reading and Writing a CSV

```
1 x <- matrix(1:30, 10)
2 write.csv(x, file="x.csv")
3 read.csv("x.csv")
4
5 write.csv(x, file="x.csv", row.names=FALSE)
6 read.csv("x.csv", header=TRUE)
```



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R37/119

Notes

---

---

---

---

---

---

---

---

---

---


R BasicsI/O

### The dreaded stringsAsFactors

Default for R functions (e.g., `read.csv()`) is `stringsAsFactors=TRUE`.

If you do not need to modify the “strings” (going straight to modeling), `stringsAsFactors=TRUE`.

If you **DO** need to modify, `stringsAsFactors=FALSE`



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R38/119

Notes

---

---

---

---

---

---

---

---

---


---

R BasicsI/O

### Serializing

```
1 x <- letters
2 y <- 1:5
3 z <- list(list("a"), b=matrix(0))
4
5 save(x, y, z, file="robjects.RData")
6 rm(x)
7 rm(y)
8 rm(z)
9 x
10 load("robjects.RData")
```

See also `saveRDS()`.



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R39/119

Notes

---

---

---

---

---

---

---

---

---

---

R BasicsI/O


### Why/Why Not Serialize?

Why:

- Serializing is a binary “as-is” format.
- Performance!

Why not:

- Only works with R (or something that can read R binary formats!)
- Endianness...



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R40/119

Notes

---

---

---

---

---

---

---

---

---


---



R BasicsI/O

Some Notable Packages

General:  
rio <https://cran.r-project.org/web/packages/rio/index.html>  
CSV and friends:  
data.table (fread()) <https://cran.r-project.org/web/packages/data.table/index.html>  
readr <https://cran.r-project.org/web/packages/readr/index.html>  
Readers of proprietary things:  
haven (SAS, SPSS, and STATA) <https://cran.r-project.org/web/packages/haven/index.html>  
readxl (Excel) <https://cran.r-project.org/web/packages/readxl/index.html>



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R41/119

Notes

---

---

---

---

---

---


---

---

R BasicsI/O

The data() Command

Many packages (and R itself) bundle data.  
Load data with `data()`  
For a list of R's: `library(help="datasets")`



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R42/119

Notes

---

---

---

---

---

---


---

---

R BasicsI/O

Example

```
1 head(iris)
2 head(mtcars)
3
4 data(package="ggplot2")
5 diamonds # error
6 data(diamonds)
7 head(diamonds)
```



wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R43/119

Notes

---

---

---

---

---

---

---

---

R Basics

Strings

R Basics

2 R Basics

All\* About R Packages

I/O

Strings

Dates

Dealing with Dataframes

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

UT

Notes

---

---

---

---

---

---

---

---

R Basics

Strings

Strings

"character" data (text).

Internal storage scheme is complicated...

Managing character data is its own course!

Just the baby basics...

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

44/119

UT

Notes

---

---

---

---

---

---

---

---

R Basics

Strings

Quotes

Three kinds: single quote ', double quote ", backtick `.

Single and double create strings.

Backticks access language objects (e.g., `+`).

Escape within a string with a *single backslash*:

```
1 quoted_quote <- "\"\\n"
2 quoted_quote
3 ## [1] "\"\\n"
4 cat(quoted_quote)
5 ## "
```

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

45/119

UT

Notes

---

---

---

---

---

---

---

---

R Basics

Strings

Example 1

```
1 letters
2 ## [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o"
3 ## [20] "p" "q" "r" "s"
4 ## [20] "t" "u" "v" "w" "x" "y" "z"
5
6 toupper(letters)
7 ## [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O"
8 ## [20] "P" "Q" "R" "S"
9 ## [20] "T" "U" "V" "W" "X" "Y" "Z"
10
11 LETTERS
12 ## [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O"
13 ## [20] "P" "Q" "R" "S"
14 ## [20] "T" "U" "V" "W" "X" "Y" "Z"
15
16 tolower(LETTERS)
17 ## [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o"
18 ## [20] "p" "q" "r" "s"
19 ## [20] "t" "u" "v" "w" "x" "y" "z"
```

wrthematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

46/119

Notes

---

---

---

---

---

---

---

---

R Basics

Strings

Example 2

```
1 x <- "Star Trek is objectively better than Star Wars"
2 strsplit(x, split=" ")
3 ## [[1]]
4 ## [1] "Star" "Trek" "is" "objectively"
5 ## [6] "better"
6 ## [6] "than" "Star" "Wars"
7
8 y <- unlist(strsplit(x, split=" "))
9 y
10 ## [1] "S" "t" "a" "r" " " "T" "r" "e" "k" " " "i" "s" " " "o" "b"
11 ## [20] "j" "e" "c" "t"
12 ## [20] "i" "v" "e" "l" "y" " " "b" "e" "t" "t" "e" "r" " " "t" "h"
13 ## [39] "a" "n" " " "S"
14 ## [39] "t" "a" "r" " " "W" "a" "r" "s"
15
16 paste(rev(y), collapse=" ")
17 ## [1] "sraW ratS naht retteb ylevitcejbo si kerT ratS"
```

wrthematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

47/119

Notes

---

---

---

---

---

---

---

---

R Basics

Strings

Example 3

```
1 paste(letters, letters)
2 ## [1] "a a" "b b" "c c" "d d" "e e" "f f" "g g" "h h" "i i" "j j"
3 ## [13] "m m" "n n" "o o" "p p" "q q" "r r" "s s" "t t" "u u" "v v"
4 ## [25] "w w" "x x"
5 ## [25] "y y" "z z"
6
7 paste(letters, letters, sep=" ")
8 ## [1] "aa" "bb" "cc" "dd" "ee" "ff" "gg" "hh" "ii" "jj" "kk" "ll"
9 ## [16] "mm" "nn" "oo"
10 ## [16] "pp" "qq" "rr" "ss" "tt" "uu" "vv" "ww" "xx" "yy" "zz"
11
12 paste(letters, letters, sep=" ", collapse=" ")
13 ## [1] "aabbccddeeffgghhiijjkkllmmnnnooppqqrrssttuvwxyz"
14
15 paste(paste(letters, collapse=" "), paste(LETTERS, collapse=" "),
16 sep=" ")
17 ## [1] "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
```

wrthematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

48/119

Notes

---

---

---

---

---

---

---

---

R Basics

Strings

Example 4

```
1 x <- rnorm(1000, mean=10)
2
3 paste("The mean of 'x' is:", mean(x))
4 ## [1] "The mean of 'x' is: 10.0157377724829"
5
6 cat(sprintf("mean:\t%.2f\nvar:\t%.2f\n", mean(x), var(x)))
7 ## mean:    10.02
8 ## var:      0.95
```

UT

wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R49/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

Strings

Regular Expressions

grep(), grepl()  
sub(), gsub()  
regexpr(), gregexpr()  
Some others...

UT

wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R50/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

Strings

Some Notable Packages

stringi  
<https://cran.r-project.org/web/packages/stringi/index.html>  
tm <https://cran.r-project.org/web/packages/tm/index.html>

UT

wrathematics.info/handouts/bas2015.htmlDrew SchmidtAn Introduction to Programming in R51/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

Dates

R Basics

2 R Basics

All\* About R Packages

I/O

Strings

Dates

Dealing with Dataframes

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

UT

Notes

---

---

---

---

---

---

---

---

R Basics

Dates

Dates

Dates managed in the UNIX style.  
Probably very alien for you...  
Just use lubridate.

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

52/119

UT

Notes

---

---

---

---

---

---

---

---

R Basics

Dates

Formatting Dates with Lubridate

```
1 rightnow <- Sys.time()
2 rightnow
3
4 library(lubridate)
5 year(rightnow)
6 ## [1] 2015
7
8 month(rightnow)
9 ## [1] 11
10 month(rightnow, label=TRUE)
11 ## [1] Nov
12
13 wday(rightnow, label=TRUE)
14 ## [1] Mon
```

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

53/119

UT

Notes

---

---

---

---

---

---

---


---

R Basics

Dates

And much more!

See the package vignette: <https://cran.r-project.org/web/packages/lubridate/vignettes/lubridate.html>



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 54/119

Notes

---

---

---

---

---

---

---

---

---


---

R Basics

Dates

Example: Timezone Lookup

```
1 timezone <- function()
2 {
3   time <- Sys.time()
4   ret <- list(timezone=format(time, format="%Z"),
5               UTC.offset=format(time, format="%z"))
6   class(ret) <- "tzlookup"
7   return(ret)
8 }
9
10 print.tzlookup <- function(x)
11 {
12   maxlen <- max(sapply(names(x), nchar))
13   spacenames <- c("timezone: ", "UTC Offset:")
14   cat(paste(spacenames, x, sep=" ", collapse="\n"), "\n")
15 }
16
17 timezone()
18 ## timezone: EST
19 ## UTC Offset: -0500
```



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 55/119

Notes

---

---

---

---

---

---

---

---

---


---

R Basics

Dates

Some Notable Packages

lubridate <https://cran.r-project.org/web/packages/lubridate/index.html>



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 56/119

Notes

---

---

---

---

---

---

---

---

---

---

R Basics

Dealing with Dataframes

R Basics

2 R Basics

All\* About R Packages

I/O

Strings

Dates

Dealing with Dataframes

wrthematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

UT

Notes

---

---

---

---

---

---

---

---

R Basics

Dealing with Dataframes

Motivation

For many, dataframes are the most important/common object.  
ENORMOUS topic.  
We cover some important techniques.

wrthematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

57/119

UT

Notes

---

---

---

---

---

---

---

---

R Basics

Dealing with Dataframes

Modifying Elements

```
1 ### Matrix-like notation
2 iris[1,1] <- 0
3 iris[, 1] <- 0
4
5 edit(iris)
```

wrthematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

58/119

UT

Notes

---

---

---

---

---

---

---


---

R Basics

Dealing with Dataframes

Adding Variables

```
1 index <- 1:nrow(iris)
2 iris$index <- index
3 head(iris)
4
5 m(iris)
6 iris <- cbind(iris, index)
7 head(iris)
8
9 m(iris)
10 iris[, ncol(iris)+1] <- index
11 head(iris)
12
13 m(iris)
```



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

59/119

Notes

---

---

---

---

---

---

---

---

---


---

R Basics

Dealing with Dataframes

Adding Rows

```
1 nrow(iris)
2 set.seed(12345)
3 newrow <- iris[sample(nrow(iris), size=1), ]
4
5 iris <- rbind(iris, newrow)
6 nrow(iris)
7
8 iris[nrow(iris)+1, ] <- newrow
9 nrow(iris)
```



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

60/119

Notes

---

---

---

---

---

---

---

---

---


---

R Basics

Dealing with Dataframes

Adding Variables

```
1 index <- 1:nrow(iris)
2 iris$index <- index
3 head(iris)
4
5 m(iris)
6 head(iris)
7
8 iris <- cbind(iris, index)
9 head(iris)
10
11 m(iris)
```



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

61/119

Notes

---

---

---

---

---

---

---

---

---

---




R Basics

Dealing with Dataframes

Subsetting

```
1 subset(iris, Sepal.Length > 7)
2
3 subset(iris, Sepal.Length > 7 & Petal.Length < 6)
4
5 subset(iris, Species == "setosa" | Species == "versicolor")
```



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

62/119

Notes

---

---

---

---

---

---

---


---

R Basics

Dealing with Dataframes

Sorting/Ordering Rows

```
1 df1 <- iris[order(iris$Sepal.Length), ]
2 head(df1)
3
4 df2 <- iris[order(-iris$Sepal.Length), ]
5 head(df2)
6
7 df3 <- iris[order(-iris$Sepal.Length, iris$Sepal.Width), ]
8 head(df3)
```



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

63/119

Notes

---

---

---

---

---

---

---


---

R Basics

Dealing with Dataframes

Sorting/Ordering Columns

```
1 iris <- iris[1:5, ]
2
3 iris[5:1]
4 set.seed(12345)
5 iris[sample(1:ncol(iris))]
6
7 iris[1]
8
9 iris[sort(colnames(iris))]
10
11 iris[sort(colnames(iris), decreasing=TRUE)]
12
13 rm(iris)
```



wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

64/119

Notes

---

---

---

---

---

---

---


---

R Basics

Dealing with Dataframes

Dealing with Duplicates

```
1 any(duplicated(iris))
2 which(duplicated(iris))
3
4 nrow(unique(iris))
```



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 65/119

Notes

---

---

---

---

---

---

---


---

R Basics

Dealing with Dataframes

Applying Functions

```
1 colMeans(iris)
2 colMeans(iris[, -5])
3
4 apply(iris[, -5], MARGIN=2, FUN=median)
```



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 66/119

Notes

---

---

---

---

---

---

---

---


R Basics

Dealing with Dataframes

Some Notable Packages

Dataframe-like things:  
data.table <https://cran.r-project.org/web/packages/data.table/index.html>  
dplyr <https://cran.r-project.org/web/packages/dplyr/index.html>

Restructuring helpers:  
reshape2 <https://cran.r-project.org/web/packages/reshape2/index.html>  
tidyr <https://cran.r-project.org/web/packages/tidyr/index.html>  
broom <https://cran.r-project.org/web/packages/broom/index.html>



[wrathematics.info/handouts/bas2015.html](http://wrathematics.info/handouts/bas2015.html) Drew Schmidt An Introduction to Programming in R 67/119

Notes

---

---

---

---

---

---

---

---

- 1 Introduction
- 2 R Basics  
Type and Structure
- 3 Programming  
Control Flow  
Loops  
Functions  
Debugging
- 4 Closing



Notes

---

---

---

---

---

---

---

---

## Type and Structure

- 3 Programming  
Control Flow  
Loops  
Functions  
Debugging



Notes

---

---

---

---

---

---

---

---

### The R Language

Storage: logical, int, double, double complex, character (strings)

Structures: vector, matrix, array, list, dataframe, environment (hashtable)

Caveats: (Logical) TRUE, FALSE, NA

In fact, there's an NA for each type:

Integer:  $-(2^{31} - 1)$

Double: value at address 0x7FF00000000007A2LL

3 official OOP systems, several unofficial ones.

Several unofficial packages supporting C++.



Notes

---

---

---

---

---

---

---

---

## Data Types

- logical
- int
- double
- double complex
- character



Notes

---

---

---

---

---

---

---

---

## Data Types

R is *dynamically typed*. You do not have to declare what kind of data a variable is before you start using it:

```
1 x <- 1
2 typeof(x)
3 x <- 1:2
4 typeof(x)
```



Notes

---

---

---

---

---

---

---

---

## Other

There are 4 “other” data “types”:

- Inf
- NaN
- NULL
- NA



Notes

---

---

---

---

---

---

---

---

## Inf

## Numerical infinity

```
1 Inf
2 ## [1] Inf
3
4 typeof(Inf)
5 ## [1] "double"
6
7 is.finite(Inf)
8 ## [1] FALSE
9
10 is.infinite(Inf)
11 ## [1] TRUE
12
13 Inf+Inf
14 ## [1] Inf
15
16 1/0
17 ## [1] Inf
```



Notes

---

---

---

---

---

---

---

---

---

---

## NaN

## Not a Number; numerical undefinedness.

```
1 NaN
2 ## [1] NaN
3
4 typeof(NaN)
5 ## [1] "double"
6
7 is.nan(NaN)
8 ## [1] TRUE
9
10 Inf - Inf
11 ## [1] NaN
12
13 sin(Inf)
14 ## [1] NaN
15 ## Warning message:
16 ## In sin(Inf) : NaNs produced
```



Notes

---

---

---

---

---

---

---

---

---

---

## NULL

The null object; a sort of placeholder for something undefined. Like a non-numeric NaN.

```
1 NULL
2 ## NULL
3
4 typeof(NULL)
5 ## [1] "NULL"
6
7 is.null(NULL)
8 ## [1] TRUE
9
10 NULL+NULL
11 ## numeric(0)
```



Notes

---

---

---

---

---

---

---

---

---

---

## NA

Missingness; not merely undefined, but *unknown*.

Each type (logical, int, double, ...) has its own NA

R is thus not boolean: TRUE, FALSE, NA

Most R methods have ways of removing NA's



Notes

---

---

---

---

---

---

---

---

---

---

## Data Structures

vector

matrix

array

factor

list

dataframe



Notes

---

---

---

---

---

---

---

---

---

---

## Vectors

```
1 1:10
2 10:1
3
4 c(1, 3, 5, 7, 9)
5 seq(from=1, to=10, by=2)
6
7 x <- 1:5
8 length(x)
```



Notes

---

---

---

---

---

---

---

---

---

---

## Matrices

```
1 matrix(1:10)
2 matrix(1:10, nrow=5)
3 matrix(1:10, ncol=5)
4
5 x <- 1:10
6 y <- as.matrix(x)
7 dim(x)
8 dim(y)
9 dim(x) <- c(1, 10)
10 x
```



Notes

---

---

---

---

---

---

---

---

---

---

## Extraction

```
1 x <- matrix(1:30, 10)
2 x
3
4 x[-1, ]
5 x[, -1]
6 x[1:5, -1]
7 x[c(2,5,7), c(1,3)]
8
9 y <- x[, -2]
10 dim(y) <- NULL
11 y
```



Notes

---

---

---

---

---

---

---

---

---

---

## Replacement

```
1 x <- matrix(1:30, 10)
2
3 x[1:5, ] <- 0
4 x[7, 3] <- NA
5 x
```



Notes

---

---

---

---

---

---

---

---

---

---

## Factors

```
1 factor(1:5)
2 factor(c("a", "b", "b", "a", "c"))
3
4 x <- factor(-1:1)
5 x
6 as.numeric(x)
7 as.numeric(as.character(factor(-1:1)))
```



Notes

---

---

---

---

---

---

---

---

---

---

## Dataframes

```
1 c(1, "a")
2 matrix(c(1, "a"))
3
4 x <- data.frame(1, "a")
5 x
6 x[1, 1]
7 is.numeric(x[1,1])
8 x[1, 2]
9 is.numeric(x[1,2])
10
11 data.frame(a=1:5,b=5:1)
```



Notes

---

---

---

---

---

---

---

---

---

---

## Lists

Super structures

Items can be any structure (even other lists)

Dataframe is really just a special list



Notes

---

---

---

---

---

---

---

---

---

---



## Lists

```
1 list(1)
2 x <- list(list(1), "a")
3 x
4 x[[1]]
5
6 x <- list(a=list("b", 1), z=1:5)
7 x$z
```



Notes

---

---

---

---

---

---

---

---

---

---

## Other Structures?

stacks, heaps, queues, graphs, ...  
 tl;dr: mostly no  
 Hash table - environments  
 Example deque <https://github.com/wrathematics/dequer>



Notes

---

---

---

---

---

---

---

---

---

---

Type and Structure

- 3 Programming
- Control Flow
- Loops
- Functions
- Debugging



Notes

---

---

---

---

---

---

---

---

---

---

## Logic

Possible values are TRUE, FALSE, and NA

Operations: &&, ||, &, |

Comparators: ==, !=, <, <=, >, and >=



Notes

---

---

---

---

---

---

---

---

---

---

## Logic

```
1 1==1
2 1!=1
3 1<1
4 1<=1
5
6 TRUE==FALSE
7
8 TRUE==1
9 FALSE==0
10
11 TRUE==T
12 FALSE==F
```



Notes

---

---

---

---

---

---

---

---

---

---

## Logic

```
1 NA==NA
2 is.na(NA)
3
4 NULL==NULL
5 is.null(NULL)
6
7 NaN==NaN
8 is.nan(NaN)
9
10 Inf==Inf
11 is.infinite(Inf)
```



Notes

---

---

---

---

---

---

---

---

---

---

## Logic

Vectors of logicals are evaluated element-wise:

```
1 x <- c(TRUE, FALSE, FALSE, TRUE)
2 x==TRUE
3 !x
```



Notes

---

---

---

---

---

---

---

---

---

---

## Logic

Some very important functions for logical evaluations:

```
1 x <- c(TRUE, FALSE, FALSE, TRUE)
2 any(x)
3 all(x)
4 which(x)
5
6 y <- -5:5
7 which(y%%2==0)
8 y[which(y%%2==0)]
```



Notes

---

---

---

---

---

---

---

---

---

---

## Comparisons

Comparing to...

Comparing to NULL: `is.null(x)`

Comparing to NA: `is.na(x)`

Comparing to TRUE: `isTRUE(x)`

Comparing two...

Tread carefully...: `x == y`

Numerics: `all.equal(x, y)`

EXACTLY THE SAME: `identical(x, y)`



Notes

---

---

---

---

---

---

---

---

---

---

Programming

Control Flow

Why all this trouble?

BECAUSE COMPUTERS ARE TERRIBLE

```
1 x <- 1
2 for (i in 1:10) x <- x - .1
3 x
```

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

92/119

Notes

---

---

---

---

---

---

---

---

---

---

Programming

Control Flow

Another famous example

```
1 sprintf("%.17f", 0.1+0.2)
2 ## [1] "0.30000000000000004"
```

This even has its own website! <http://0.30000000000000004.com/>

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

93/119

Notes

---

---

---

---

---

---

---

---

---

---

Programming

Loops

Type and Structure

3 Programming

Control Flow

Loops

Functions

Debugging

wrathematics.info/handouts/bas2015.html

Drew Schmidt

An Introduction to Programming in R

Notes

---

---

---

---

---

---

---

---

---

---

## Loops

Instruction set to be repeatedly executed until some condition is satisfied (possibly forever).

R has `for()` and `while()`.

`for()`: Iterates over a list or vector

`while()`: Performs operations until logical condition is satisfied.



Notes

---

---

---

---

---

---

---

---

---

---

## for Loop

```
1 for (i in 1:10){
2   cat(i, " ")
3 }
4 ## 1 2 3 4 5 6 7 8 9 10
5
6 x <- matrix(1:30, 10)
7 colmax <- numeric(3)
8 for (i in 1:3){
9   colmax[i] <- max(x[, i])
10 }
11
12 colmax
13 ## [1] 10 20 30
```



Notes

---

---

---

---

---

---

---

---

---

---

## while Loop

```
1 i <- 1
2 while (i < 11){
3   print(i)
4   i <- i+1
5 }
6
7 n <- 2
8 i <- 1
9 while (n < 1000){
10   n <- n^2
11   i <- i*2
12 }
13
14 n
15 ## [1] 65536
16 i
17 ## [1] 16
```



Notes

---

---

---

---

---

---

---

---

---

---

## The \*ply Family

`apply()`: Apply function across “margin” (dimension) of matrix.

`lapply()`: Apply function to input data object; returns a list.

`sapply()`: Same as `sapply()` but

`mapply()`: Multivariate `sapply()`.

`vapply()`: Essentially `sapply()`, sometimes faster.

`tapply()`: Applying a function to a subset of a vector.

...

We will only discuss the first 3.



Notes

## What is it?

Syntax to loop without writing a loop.

Inspired by functional programming.

**Not the same thing as vectorization** (but it looks vectorized).



Notes

## apply

```

1 x <- matrix(1:30, 10)
2 x
3
4 apply(X=x, MARGIN=1, FUN=min)
5 ## [1] 1 2 3 4 5 6 7 8 9 10
6 apply(X=x, MARGIN=2, FUN=min)
7 ## [1] 1 11 21
8
9 out <- numeric(3)
10 for (i in 1:3){
11   out[i] <- min(x[, i])
12 }
13 out
14 ## [1] 1 11 21

```



Notes

## lapply and sapply

```

1 x <- 1:5
2 lapply(X=x, FUN=sqrt)
3 ## [[1]]
4 ## [1] 1
5 ##
6 ## [[2]]
7 ## [1] 1.414214
8 ##
9 ## [[3]]
10 ## [1] 1.732051
11 ##
12 ## [[4]]
13 ## [1] 2
14 ##
15 ## [[5]]
16 ## [1] 2.236068
17
18 sapply(X=x, FUN=sqrt)
19 ## [1] 1.000000 1.414214 1.732051 2.000000 2.236068

```



Notes

---

---

---

---

---

---

---

---

---

---

## \*ply Functions Internally

**apply():** A for() loop.

**lapply():** Internal R voodoo; faster than a loop.

**sapply():** Essentially the same as lapply().



Notes

---

---

---

---

---

---

---

---

---

---

## When does all this choice matter?

loops are slow.

**apply()** is sugar for an R for loop.

**lapply()** different, often faster.

Vectorization is fastest of all.



Notes

---

---

---

---

---

---

---

---

---

---

## Loop Speeds

```

1 x <- 100000:1
2
3 system.time({ # No initialization
4   sin <- numeric(0)
5   for (i in 1:length(x)){
6     sin[i] <- sin(x[i])
7   }
8   sin
9 })
10 ##      user      system elapsed
11 ## 17.320    1.072   18.376
12
13 system.time({ # With initialization
14   sin <- numeric(length(x))
15   for (i in 1:length(x)){
16     sin[i] <- sin(x[i])
17   }
18   sin
19 })
20 ##      user      system elapsed
21 ##  0.172    0.000    0.171

```

Notes

---

---

---

---

---

---

---

---

---

---

## Loop Speeds

```

1 system.time(lapply(x, sin))
2 ##      user      system elapsed
3 ##  0.056    0.004    0.059
4
5 system.time(sapply(x, sin))
6 ##      user      system elapsed
7 ##  0.048    0.004    0.053
8
9 system.time(sin(x))
10 ##      user      system elapsed
11 ##  0.004    0.000    0.004

```



Notes

---

---

---

---

---

---

---

---

---

---

## Type and Structure

## 3 Programming

Control Flow

Loops

**Functions**

Debugging



Notes

---

---

---

---

---

---

---

---

---

---



## Functions

Self-contained input/output machine.  
Reusable blocks of code.  
First class objects.  
All functions have returns!  
*Evaluating the Design of the R language,*  
<http://r.cs.purdue.edu/pub/ecoop12.pdf>



Notes

---

---

---

---

---

---

---

---

---

---

## Some Basic Rules

Can not modify data in place\* (multi-return with a list)  
Arguments can have defaults, specified with `=`.  
By default, no printing occurs (have to say `print(x)`).  
The last “thing done” is the return.  
People often use `NULL` or `invisible(NULL)` if return is unimportant.



Notes

---

---

---

---

---

---

---

---

---

---

## Functions: Example 1

```
1 f <- function(x)
2 {
3   ret <- x+1
4   return(ret)
5 }
6
7 f(1)
8 ## [1] 2
9 f(2)
10 ## [1] 3
11 f(5)
12 ## [1] 6
13 f
14 ## function(x)
15 ## {
16 ##   ret <- x+1
17 ##   return(ret)
18 ## }
```



Notes

---

---

---

---

---

---

---

---

---

---

## Functions: Example 2

```

1 f <- function (a, b)
2 {
3   a - b
4 }
5
6 f(a=1, b=2)
7 f(1, 2)
8 f(b=1, a=2)
9 f(b=1, 2)
10 f(1)
11 f(matrix(1:4, nrow=2), matrix(4:1, nrow=2))

```



Notes

## Functions: Example 3

For complicated returns (especially of mixed type/class), use a list:

```

1 g <- function (a, b)
2 {
3   plus <- a+b
4   minus <- a-b
5   return(list(plus, minus))
6 }
7
8 g(5, 2)
9 g(1, 0)
10 g(f(2, 6), 2)

```



Notes

## Functions: Example 4

R allows parameter defaults in functions

```

1 h <- function (a=1, b=2)
2 {
3   return(b-a)
4 }
5
6 h
7 h()
8 h(2, 1)

```



Notes

## Recursive Functions

A *recursive function* is one that calls itself:

```

1 f <- function(n)
2 {
3   if (n==1)
4     return(1)
5   else {
6     if (n%%2==0)
7       return(f(n/2))
8     else
9       return(f(3*n+1))
10  }
11 }
12
13 f(22)
14 ## [1] 1
15 f(237)
16 ## [1] 1
17 sapply(1:37, f)
18 ##      [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
      [1] 1 1 1 1 1 1 1

```

Notes

---

---

---

---

---

---

---

---

---

---

Type and Structure

### 3 Programming

Control Flow

Loops

Functions

Debugging



Notes

---

---

---

---

---

---

---

---

---

---

## Debugging R Code

Very broad topic ...

We'll hit the highlights.

For more examples, see:

[cran.r-project.org/doc/manuals/R-exts.html#Debugging](http://cran.r-project.org/doc/manuals/R-exts.html#Debugging)

Debugging compiled code called by R (valgrind, gdb, ...) also possible...



Notes

---

---

---

---

---

---

---

---

---

---

## Object Inspection Tools

print()  
str()  
unclass()



Notes

## Object Inspection Tools: print()

Basic printing:

```
1 x <- matrix(1:10, nrow=2)
2 print(x)
3 ##      [,1] [,2] [,3] [,4] [,5]
4 ## [1,]  1   3   5   7   9
5 ## [2,]  2   4   6   8  10
6
7 x
8 ##      [,1] [,2] [,3] [,4] [,5]
9 ## [1,]  1   3   5   7   9
10 ## [2,]  2   4   6   8  10
```



Notes

## Object Inspection Tools: str()

Examining the structure of an R object:

```
1 x <- matrix(1:10, nrow=2)
2
3 str(x)
4 ## int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
```



Notes

## Object Inspection Tools: unclass()

Exposing all data with unclass():

```
1 df <- data.frame(x=rnorm(10), y=rnorm(10))
2 mdl <- lm(y~x, data=df) ### That's a "tilde" character
3
4 mdl
5 print(mdl)
6
7 str(mdl)
8
9 unclass(mdl)
```



Notes

---

---

---

---

---

---

---

---

---

---

## The R Debugger

```
debug()
debugonce()
undebug()
```



Notes

---

---

---

---

---

---

---

---

---

---

## Using The R Debugger

- 1 Declare function to be debugged: `debug(foo)`
- 2 Call function: `foo(arg1, arg2, ...)`  
*next:* Enter or n followed by Enter.  
*break:* Halt execution and exit debugging: Q.  
*exit:* Continue execution and exit debugging: C.
- 3 Call `undebug()` to stop debugging



Notes

---

---

---

---

---

---

---

---

---

---

## Using the Debugger

### Example Debugger Interaction

```
> f <- function(x){y <- z+1;z <- y*2;z}
> f(1)
Error in f(1) : object 'z' not found
> debug(f)
> f(1)
debugging in: f(1)
debug at #1: {
  y <- z + 1
  z <- y * 2
  z
}
Browse[2]>
debug at #1: y <- z + 1
Browse[2]>
Error in f(1) : object 'z' not found
>
```



Notes

---

---

---

---

---

---

---

---

---

---

- 1 Introduction
- 2 R Basics
- 3 Programming
- 4 Closing



Notes

---

---

---

---

---

---

---

---

---

---

Thanks so much for attending!

# Questions?



Notes

---

---

---

---

---

---

---

---

---

---