

**Probability and Statistics (IT302) Class No. 24**  
**12<sup>th</sup> October 2020 Monday 09:45AM - 10:15AM**

# Introduction to R Programming Language

R is a programming language and environment commonly used in statistical computing, data analytics and scientific research.

It is one of the most popular languages used by statisticians, data analysts, researchers and marketers to retrieve, clean, analyze, visualize and present data.

Due to its expressive syntax and easy-to-use interface, it has grown in popularity in recent years.

- R is a programming language and software environment for statistical analysis, graphics representation and reporting.
- The core of R is an interpreted computer language which allows branching and looping as well as modular programming using functions.
- R allows integration with the procedures written in the C, C++, .Net, Python or FORTRAN languages for efficiency.
- R is freely available under the GNU General Public License, and pre-compiled binary versions are provided for various operating systems like Linux, Windows and Mac.
- R is free software distributed under a GNU-style copy left, and an official part of the GNU project called GNU S

# Features of R

As stated earlier, R is a programming language and software environment for statistical analysis, graphics representation and reporting. **The following are the important features of R:**

- R is a well-developed, simple and effective programming language which includes conditionals, loops, user defined recursive functions and input and output facilities.
- R has an effective data handling and storage facility,
- R provides a suite of operators for calculations on arrays, lists, vectors and matrices.
- R provides a large, coherent and integrated collection of tools for data analysis.
- R provides graphical facilities for data analysis and display either directly at the computer or printing at the papers.
- As a conclusion, R is world's most widely used statistics programming language. It's the #1 choice of data scientists and supported by a vibrant and talented community of contributors. R is taught in universities and deployed in mission critical business applications.

# Why use R

- R is an open source programming language and software environment for statistical computing and graphics.
- R is an object oriented programming environment, much more than most other statistical software packages.
- R is a comprehensive statistical platform, offering all manner of data-analytic techniques – any type of data analysis can be done in R.
- R has state-of-the-art graphics capabilities- visualize complex data.
- R is a powerful platform for interactive data analysis and exploration. • Getting data into a usable form from multiple sources .
- R functionality can be integrated into applications written in other languages, including C++, Java, Python , PHP, SAS and SPSS.
- R runs on a wide array of platforms, including Windows, Unix and Mac OS X. • R is extensible; can be expanded by installing “packages”

# Why use R for statistical computing and graphics?

## **R is open source and free!**

- R is free to download as it is licensed under the terms of GNU General Public license.

## **R is popular - and increasing in popularity**

- IEEE publishes a list of the most popular programming languages each year. R was ranked 5th in 2016, up from 6th in 2015. It is a big deal for a domain-specific language like R to be more popular than a general purpose language like C#.
- This not only shows the increasing interest in R as a programming language, but also of the fields like Data Science and Machine Learning where R is commonly used.

# Why use R for statistical computing and graphics? Contd.

## **R runs on all platforms**

You can find distributions of R for all popular platforms - Windows, Linux and Mac.

R code that you write on one platform can easily be ported to another without any issues.

## **R is being used by the biggest tech giants**

Adoption by tech giants is always a sign of a programming language's potential. Today's companies don't make their decisions on a whim. Every major decision has to be backed by concrete analysis of data.

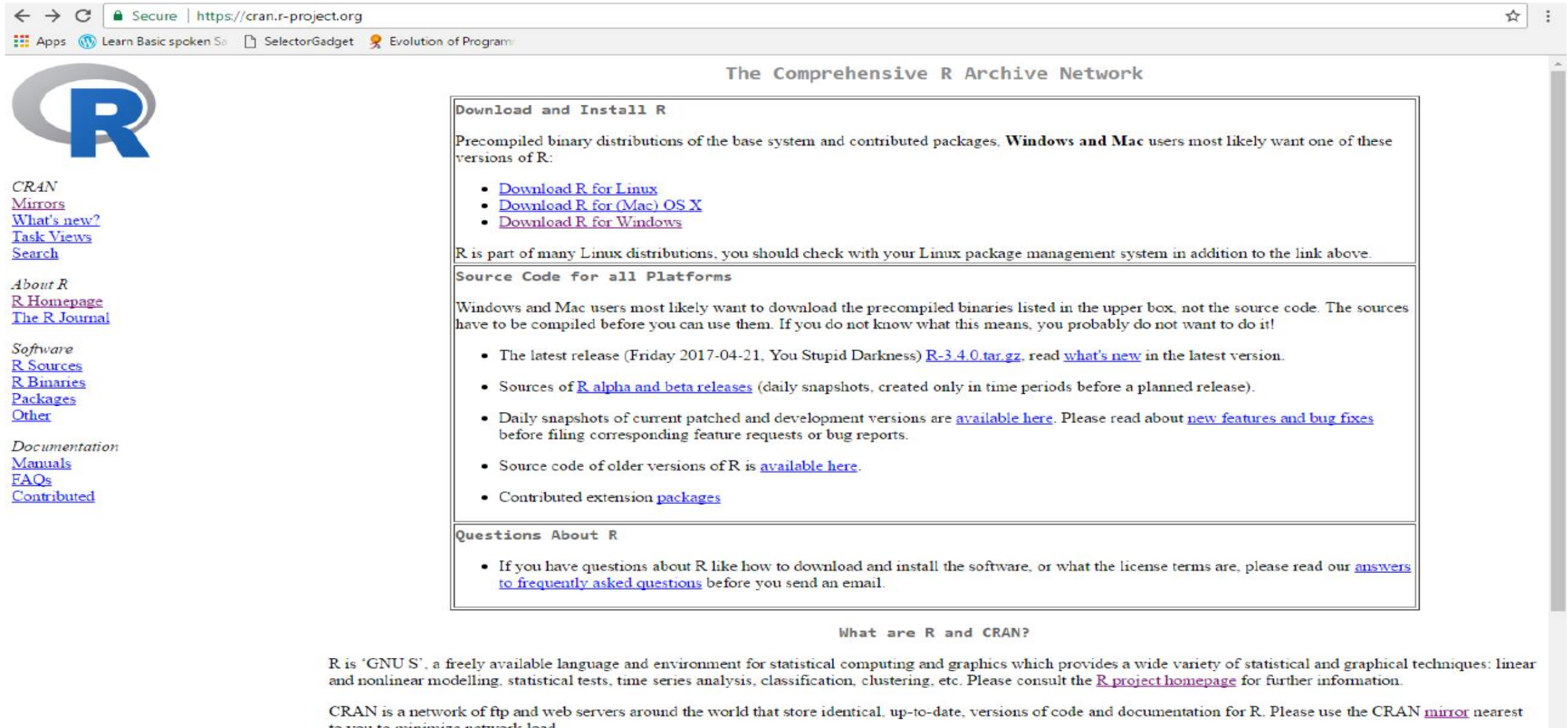
### **Companies Using R**

R is the right mix of simplicity and power, and companies all over the world use it to make calculated decisions. Here are a few ways industry stalwarts are using R and contributing to the R ecosystem.

# Downloading and Installing R

- R is free available from the comprehensive R Archive Network (CRAN) at <http://cran.r-project.org>
- Precompiled binaries are available for Linux, Mac OS X and windows.
- R latest release R-3.4.0
- Installing R on windows and Mac is just like installing any other program.
- Install R Studio: a free IDE for R at <http://www.rstudio.com/>
- If we install R and R Studio, then we need to run R Studio only.
- R is case-sensitive.
- R scripts are simply text files with a .R extension.

# Downloading and Installing R Contd.



The screenshot shows the CRAN website with the following content:

**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 (Friday 2017-04-21, You Stupid Darkness) [R-3.4.0.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 [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

**Questions About R**

- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

**What are R and CRAN?**

R is 'GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the [R project homepage](#) for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN [mirror](#) nearest to you to minimize network load.

**Left sidebar links:**


- 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](#)



# Downloading and Installing R Contd.

← → ↻ Secure | https://cran.r-project.org ☆ ⋮

Apps Learn Basic spoken Sa SelectorGadget Evolution of Program



## R for Windows

Subdirectories:

<a href="#">base</a>	Binaries for base distribution (managed by Duncan Murdoch). This is what you want to <a href="#">install R for the first time</a> .
<a href="#">contrib</a>	Binaries of contributed CRAN packages (for R $\geq$ 2.11.x; managed by Uwe Ligges). There is also information on <a href="#">third party software</a> available for CRAN Windows services and corresponding environment and make variables.
<a href="#">old contrib</a>	Binaries of contributed CRAN packages for outdated versions of R (for R $<$ 2.11.x; managed by Uwe Ligges).
<a href="#">Rtools</a>	Tools to build R and R packages (managed by Duncan Murdoch). This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Duncan Murdoch or Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](#) and [R for Windows FAQ](#).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

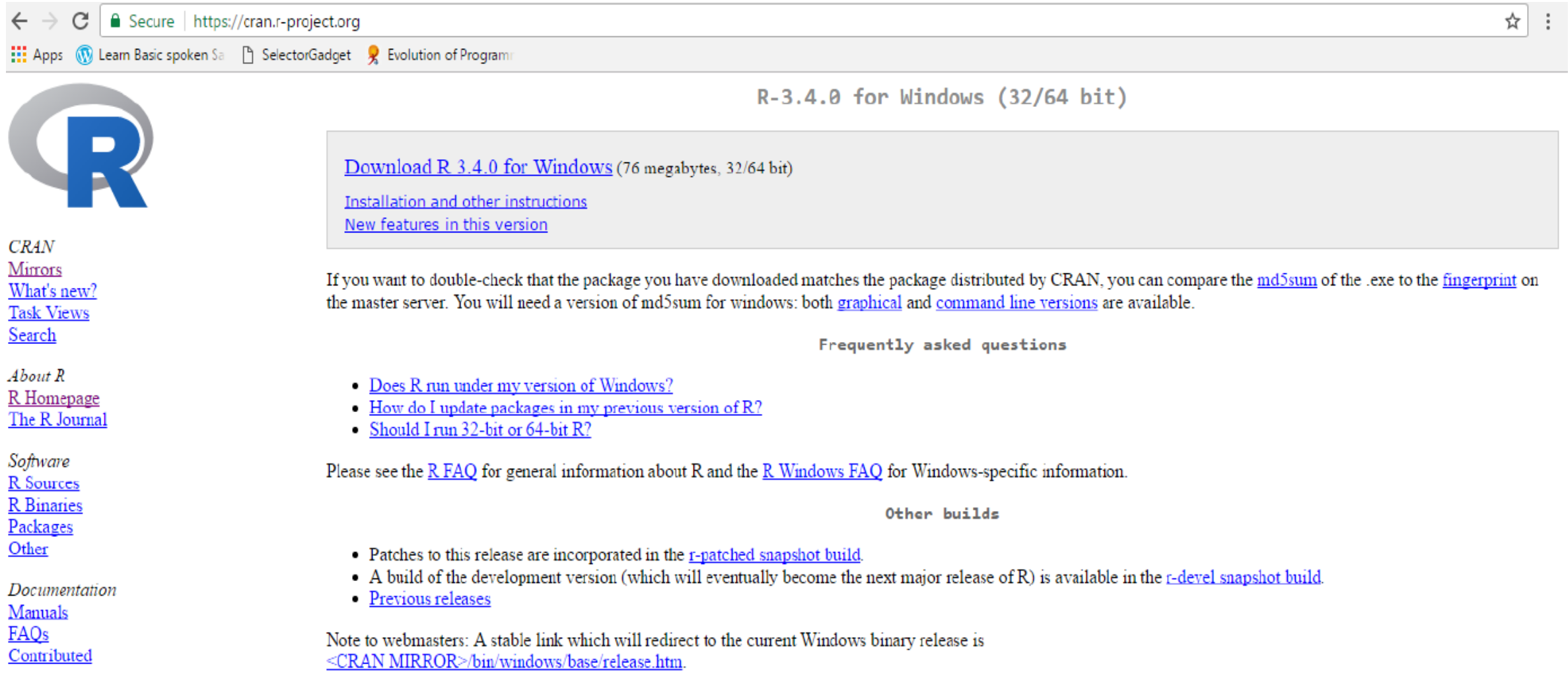
*CRAN*  
[Mirrors](#)  
[What's new?](#)  
[Task Views](#)  
[Search](#)

*About R*  
[R Homepage](#)  
[The R Journal](#)

*Software*  
[R Sources](#)  
[R Binaries](#)  
[Packages](#)  
[Other](#)

*Documentation*  
[Manuals](#)  
[FAQs](#)  
[Contributed](#)

# Downloading and Installing R Contd.




The screenshot shows the CRAN website for downloading R 3.4.0 for Windows. The browser address bar shows 'https://cran.r-project.org'. The page features the R logo on the left and a navigation menu with links like 'About R', 'R Sources', 'R Binaries', 'Packages', 'Other', 'Documentation', 'Manuals', 'FAQs', and 'Contributed'. The main content area is titled 'R-3.4.0 for Windows (32/64 bit)' and contains a box with links to 'Download R 3.4.0 for Windows (76 megabytes, 32/64 bit)', 'Installation and other instructions', and 'New features in this version'. Below this, a paragraph explains how to verify the download using md5sum. A 'Frequently asked questions' section lists three common queries. An 'Other builds' section mentions patches and development versions. A note to webmasters provides a stable link to the current Windows binary release.

← → ↻ Secure | https://cran.r-project.org ☆ ⋮

Apps Learn Basic spoken Sa SelectorGadget Evolution of Program

## R-3.4.0 for Windows (32/64 bit)



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](#)

[Download R 3.4.0 for Windows](#) (76 megabytes, 32/64 bit)

[Installation and other instructions](#)

[New features in this version](#)

If you want to double-check that the package you have downloaded matches the package distributed by CRAN, you can compare the [md5sum](#) of the .exe to the [fingerprint](#) on the master server. You will need a version of md5sum for windows: both [graphical](#) and [command line versions](#) are available.

### Frequently asked questions

- [Does R run under my version of Windows?](#)
- [How do I update packages in my previous version of R?](#)
- [Should I run 32-bit or 64-bit R?](#)

Please see the [R FAQ](#) for general information about R and the [R Windows FAQ](#) for Windows-specific information.

### Other builds

- Patches to this release are incorporated in the [r-patched snapshot build](#).
- A build of the development version (which will eventually become the next major release of R) is available in the [r-devel snapshot build](#).
- [Previous releases](#)

Note to webmasters: A stable link which will redirect to the current Windows binary release is [<CRAN MIRROR>/bin/windows/base/release.htm](#).

Last change: 2017-04-21, by Duncan Murdoch

# Run R Programming on Your Computer

You will find the easiest way to run R programming on your system (Windows) in this section.

## Run R Programming in Windows

---

1. Go to [official site of R programming](https://www.r-project.org/)(https://www.r-project.org/)
2. Click on the CRAN link on the left sidebar
3. Select a mirror
4. Click "Download R for Windows"
5. Click on the link that downloads the base distribution
6. Run the file and follow the steps in the instructions to install R.

# Getting help in R

To get help on specific topics, we can use the `help()` function along with the topic we want to search. We can also use the `?` operator for this.

```
> help(Syntax)  
> ?Syntax
```

Also there is `help.search()` function to do a search engine type of search. One could use the `??` operator for this.

```
> help.search("histograms")  
> ?? "histograms"
```

# Starting an R session

The R programming can be done in two ways. Either type the command lines on the screen inside an "R-session", or can save the commands as a "script" file and execute the whole file inside R.

First learn the R-session.

To start an R session, type 'R' from the command line in windows or linux OS.

For example, from shell prompt '\$' in linux, type

```
$ R
```

This generates the following output before entering the '>>' prompt of R:

# Starting an R session Contd.

R version 3.1.1 (2014-07-10) -- "Sock it to Me"  
Copyright (C) 2014 The R Foundation for Statistical Computing  
Platform: x86\_64-unknown-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.

[Previously saved workspace restored]

>

# Working with R Session

Once we are inside the R session, we can directly execute R language commands by typing them line by line.

Pressing the enter key terminates typing of command and brings the > prompt again.

In the example session below, two variables 'a' and 'b' are declared to have values 5 and 6 respectively, and assign their sum to another variable called 'c':

```
> a = 5
```

```
> b = 6
```

```
> c = a + b
```

```
> c
```

The value of the variable 'c' is printed as,

```
[1] 11
```

**In R session, typing a variable name prints its value on the screen.**

## Get help inside R Session

To get help on any function of R, type `help(function-name)` in R prompt.

For example, if help is needed on "if" logic, type,

```
> help("if")
```

then, help lines for the "if" statement are printed.

## Exit the R session

To exit the R session, type `quit()` in the R prompt, and say 'n' (no) for saving the workspace image. This means, we do not want to save the memory of all the commands we typed in the current session:

```
> quit()
```

```
Save workspace image? [y/n/c]: n
```

```
>
```



# Saving the R session

Note that by not saving the current session, we lose all the memory of current session commands and the variables and objects created when we exit R prompt.

When we work in R, the R objects we created and loaded are stored in a memory portion called workspace. When we say 'no' to saving the workspace, we lose all these objects are wiped out from the workspace memory. If we say 'yes', they are saved into a file called ".RData" is written to the present working directory.

In **Linux**, this "**working directory**" is generally the directory from where R was started through the command 'R'. In **windows**, it can be either "My Documents" or user's home directory.

# Listing the objects in the current R session

We can list the names of the objects in the current R session by `ls()` command.

For example, start R session fresh and proceed as follows:

```
>  
> a = 5  
> b = 6  
> c = 8  
> sum = a+b+c  
> sum  
[1] 19  
> ls()  
[1] "a" "b" "c" "sum"
```

Here, the objects we created have been listed.

# Removing objects from the current R session

Specific objects created in the current session can be removed using `rm()` command.

If we specify the name of an object, it will be removed. If we just say `rm(list = ls())` , all objects created so far will be removed. See below:

```
> a = 5
> b = 6
> c = 8
> sum = a+b+c
> sum
[1] 19
> ls()
[1] "a" "b" "c" "sum"
>
> rm(list=c("sum"))
> ls()
[1] "a" "b" "c"
>
> rm(list = ls())
> ls()
character(0)
```

# Getting and setting the current working directories

From R prompt, we can get information about the current working directory using `getwd()` command:

```
> getwd()  
[1] "/home/user"
```

Similarly, we can set the current working directory by calling `setwd()` function:

```
> setwd("/home/user/prog")
```

After this, `"/home/user/prog"` will be the working directory.

In Windows version of R, the working directory can be set from menu in R window.

# Comments

Comments are like helping text in your R program and they are ignored by the interpreter while executing your actual program.

Single comment is written using # in the beginning of the statement as follows:

```
# My first program in R Programming
```

R does not support multi-line comments

# R Reserved Words

Reserved words in R programming are a set of words that have special meaning and cannot be used as an identifier (variable name, function name etc.).

Here is a list of reserved words in the R's parser.

Reserved words in R

if	else	repeat	while	function
for	in	next	break	TRUE
FALSE	NULL	Inf	NaN	NA
NA_integer_	NA_real_	NA_complex_	NA_character_	...

This list can be viewed by typing `help(reserved)` or `?reserved` at the R command prompt as follows.

```
> ?reserved
```

# R Reserved Words Contd.

Among these words, **if**, **else**, **repeat**, **while**, **function**, **for**, **in**, **next** and **break** are used for conditions, loops and user defined functions.

They form the basic building blocks of programming in R.

TRUE and FALSE are the logical constants in R.

NULL represents the absence of a value or an undefined value.

Inf is for "Infinity", for example when 1 is divided by 0 whereas NaN is for "Not a Number", for example when 0 is divided by 0.

NA stands for "Not Available" and is used to represent missing values.

**R is a case sensitive language. Which mean that TRUE and True are not the same.**

# Variables in R

Variables are used to store data, whose value can be changed according to our need. Unique name given to variable (function and objects as well) is identifier.

## Rules for writing Identifiers in R

- 1) Identifiers can be a combination of letters, digits, period (.) and underscore (\_).
- 2) It must start with a letter or a period. If it starts with a period, it cannot be followed by a digit.
- 3) Reserved words in R cannot be used as identifiers.

## Valid identifiers in R

total, Sum, .fine.with.dot, this\_is\_acceptable, Number5

## Invalid identifiers in R

tot@1, 5um, \_fine, TRUE, .0ne



# Variables in R (Best Practices)

- Earlier versions of R used underscore (\_) as an assignment operator. So, the period (.) was used extensively in variable names having multiple words.
- Current versions of R support underscore as a valid identifier but it is good practice to use period as word separators.
- For example, **a.variable.name** is preferred over **a\_variable\_name** or alternatively we could use camel case as a **VariableName**

# Constants in R

- Constants, as the name suggests, are entities whose value cannot be altered. **Basic types of constant are numeric constants and character constants.**

## Numeric Constants

- All numbers fall under this category.
- They can be of type integer, double or complex.
- It can be checked with the `typeof()` function.
- Numeric constants followed by `L` are regarded as integer and those followed by `i` are regarded as complex.

```
> typeof(5)
[1] "double"

> typeof(5L)
[1] "integer"

> typeof(5i)
[1] "complex"
```

Numeric constants preceded by `0x` or `0X` are interpreted as hexadecimal numbers.

```
> 0xff
[1] 255

> 0XF + 1
[1] 16
```

# Character Constants

Character constants can be represented using either single quotes (') or double quotes (") as delimiters.

```
> 'example'
```

```
[1] "example"
```

```
> typeof("5")
```

```
[1] "character"
```

# Built-in Constants

Some of the built-in constants defined in R along with their values is shown below.

```
> LETTERS
```

```
[1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P"  
"Q" "R" "S"  
[20] "T" "U" "V" "W" "X" "Y" "Z"
```

```
> letters
```

```
[1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p"  
"q" "r" "s"  
[20] "t" "u" "v" "w" "x" "y" "z"
```

```
> pi
```

```
[1] 3.141593
```

# Built-in Constants Contd.

```
> month.name  
[1] "January" "February" "March"     "April"     "May"  
"June"  
[7] "July"    "August"   "September" "October"   "November"  
"December"  
  
> month.abb  
[1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct"  
"Nov" "Dec"
```

But it is not good to rely on these, as they are implemented as variables whose values can be changed.

```
> pi  
[1] 3.141593  
  
> pi <- 56  
> pi  
[1] 56
```

# Example: Hello World Program

```
> # We can use the print() function
```

```
> print("Hello World!")
```

```
[1] "Hello World!"
```

```
> # Quotes can be suppressed in the output
```

```
> print("Hello World!", quote = FALSE)
```

```
[1] Hello World!
```

```
> # If there are more than 1 item, we can concatenate using paste()
```

```
> print(paste("How", "are", "you?"))
```

```
[1] "How are you?"
```

# R - Data Types

You may like to store information of various data types like character, wide character, integer, floating point, double floating point, Boolean etc.

Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory.

In contrast to other programming languages like C and java in R, the variables are not declared as some data type. The variables are assigned with R-Objects and the data type of the R-object becomes the data type of the variable.

There are many types of R-objects. The frequently used ones are –

Vectors      ☐ Lists      ☐ Matrices      ☐ Arrays      ☐ Factors      ☐ Data  
Frames

# R - Data Types Contd.

The simplest of these objects is the vector object and there are six data types of these atomic vectors, also termed as six classes of vectors. The other R-Objects are built upon the atomic vectors.

Data Type	Example	Verify
Logical	TRUE, FALSE	<pre>v &lt;- TRUE print(class(v))</pre> <p>it produces the following result</p> <p>—</p> <pre>[1] "logical"</pre>



# R - Data Types Contd.

Numeric	12.3, 5, 999	<pre>v &lt;- 23.5</pre> <pre>print(class(v))</pre> <p>it produces the following result</p> <p>—</p> <pre>[1] "numeric"</pre>
Integer	2L, 34L, 0L	<pre>v &lt;- 2L</pre> <pre>print(class(v))</pre> <p>it produces the following result</p> <p>—</p> <pre>[1] "integer"</pre>

# R - Data Types Contd.

Complex	3 + 2i	<pre>v &lt;- 2+5i print(class(v))</pre>
		it produces the following result —
		<pre>[1] "complex"</pre>
Character	'a' , '"good"', "TRUE", '23.4'	<pre>v &lt;- "TRUE" print(class(v))</pre>
		it produces the following result —
		<pre>[1] "character"</pre>

# R - Data Types Contd.

Raw	"Hello" is stored as 48 65 6c 6c 6f	<pre>v &lt;- charToRaw("Hello") print(class(v))</pre> <p>it produces the following result</p> <p>—</p> <pre>[1] "raw"</pre>
-----	-------------------------------------	---