충남대학교
Chungnam National University

투자론

- *R*과 Excel을 통한 금융데이터 분석 -

7주차
R(입문 및 데이터 포맷)과 Data Management

충남대학교
장호규 교수

충남대학교
Chungnam National University

Unit 01

Introduction to R

충남대학교
Chungnam National University

Overview

- What is R?
- Rstudio
- Setting Environment: R packages
- Basic R Coding

R as a programming environment

- R is a programming environment for statistical computing and graphics

- R
  - serves as a data analysis and storage facility
  - is designed to perform operations on vectors and matrices
  - uses a well-developed but simple programming language (called **s**)
  - allows for rapid development of new tools according to user demand

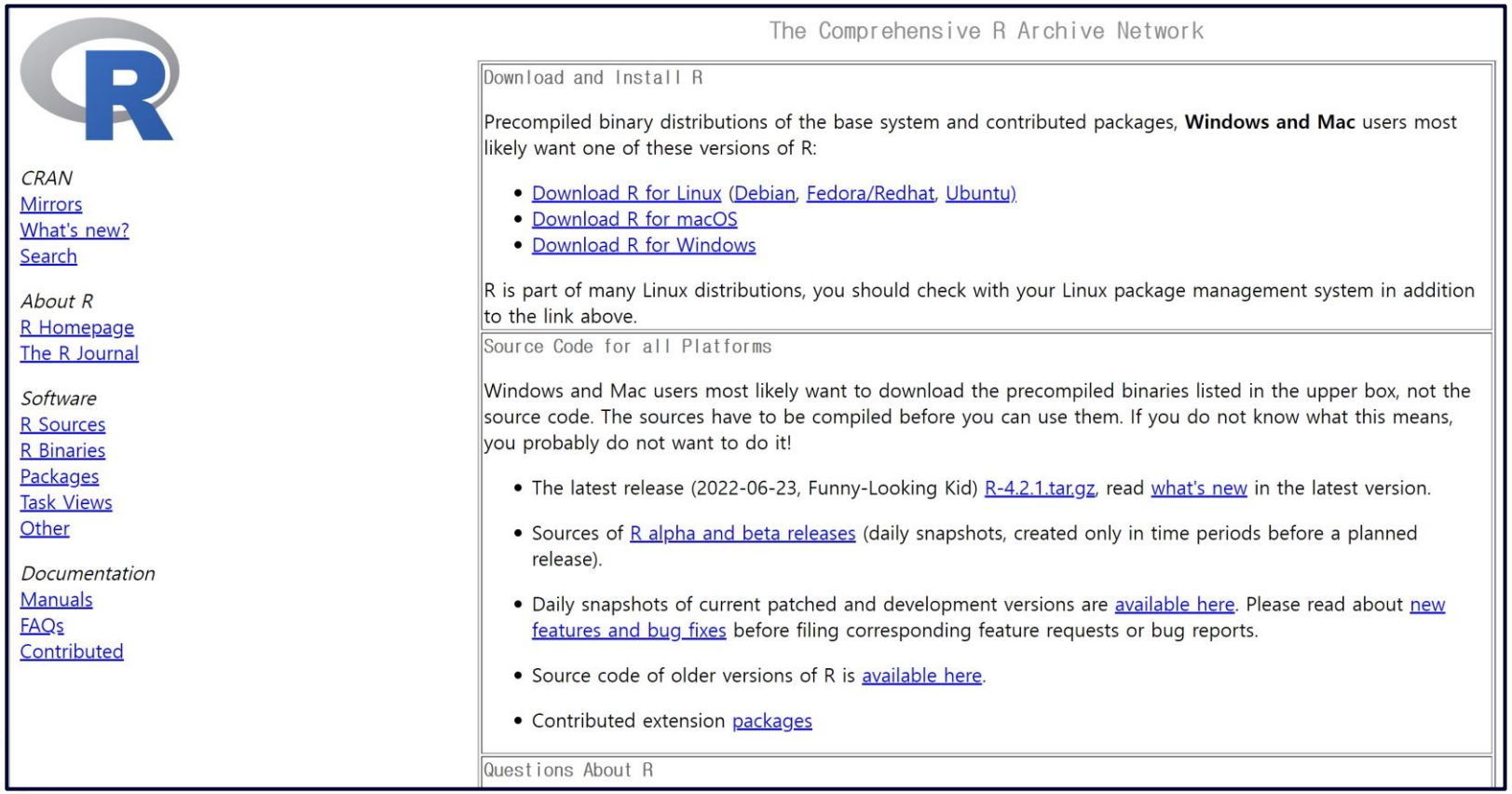
Why R?

- **R has many advantages as data analysis software:**
  - Free
  - Free online books to learn R: **Home | Bookdown**
  - **Powerful, intuitive graphics systems** make it easy to produce publication-quality graphics
  - Easily create data analysis reports as documents and presentations for reproducibility with **R Markdown**
  - Many specialized packages featuring analysis tools not available in other software

- **Disadvantage: frequent updates require maintenance**

Installing R

- https://cran.r-project.org



The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Search

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Task Views
Other

Documentation
Manuals
FAQs
Contributed

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 (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- 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.

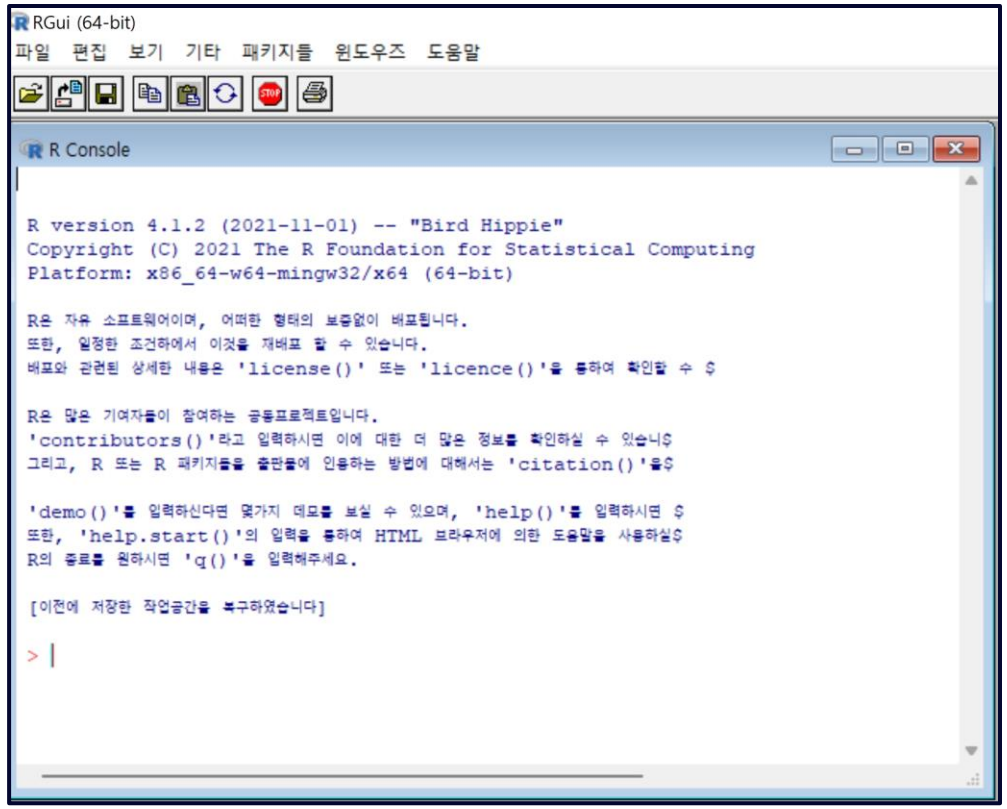
Source Code for all Platforms

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 (2022-06-23, Funny-Looking Kid) R-4.2.1.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 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

Native R GUI


RGui (64-bit)
파일 편집 보기 기타 패키지들 윈도우즈 도움말
R Console
R version 4.1.2 (2021-11-01) -- "Bird Hippie"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R은 자유 소프트웨어이며, 어떠한 형태의 보증없이 배포됩니다.
또한, 일정한 조건하에서 이것을 재배포 할 수 있습니다.
배포와 관련된 상세한 내용은 'license()' 또는 'licence()'을 통하여 확인할 수 $
R은 많은 기여자들이 참여하는 공동프로젝트입니다.
'contributors()'라고 입력하시면 이에 대한 더 많은 정보를 확인하실 수 있습니$
그리고, R 또는 R 패키지들을 출판물에 인용하는 방법에 대해서는 'citation()'을$
'demo()'를 입력하신다면 몇가지 데모를 보실 수 있으며, 'help()'를 입력하시면 $
또한, 'help.start()'의 입력을 통하여 HTML 브라우저에 의한 도움말을 사용하실$
R의 종료를 원하시면 'q()'을 입력해주세요.
[이전에 저장한 작업공간을 복구하였습니다]
>

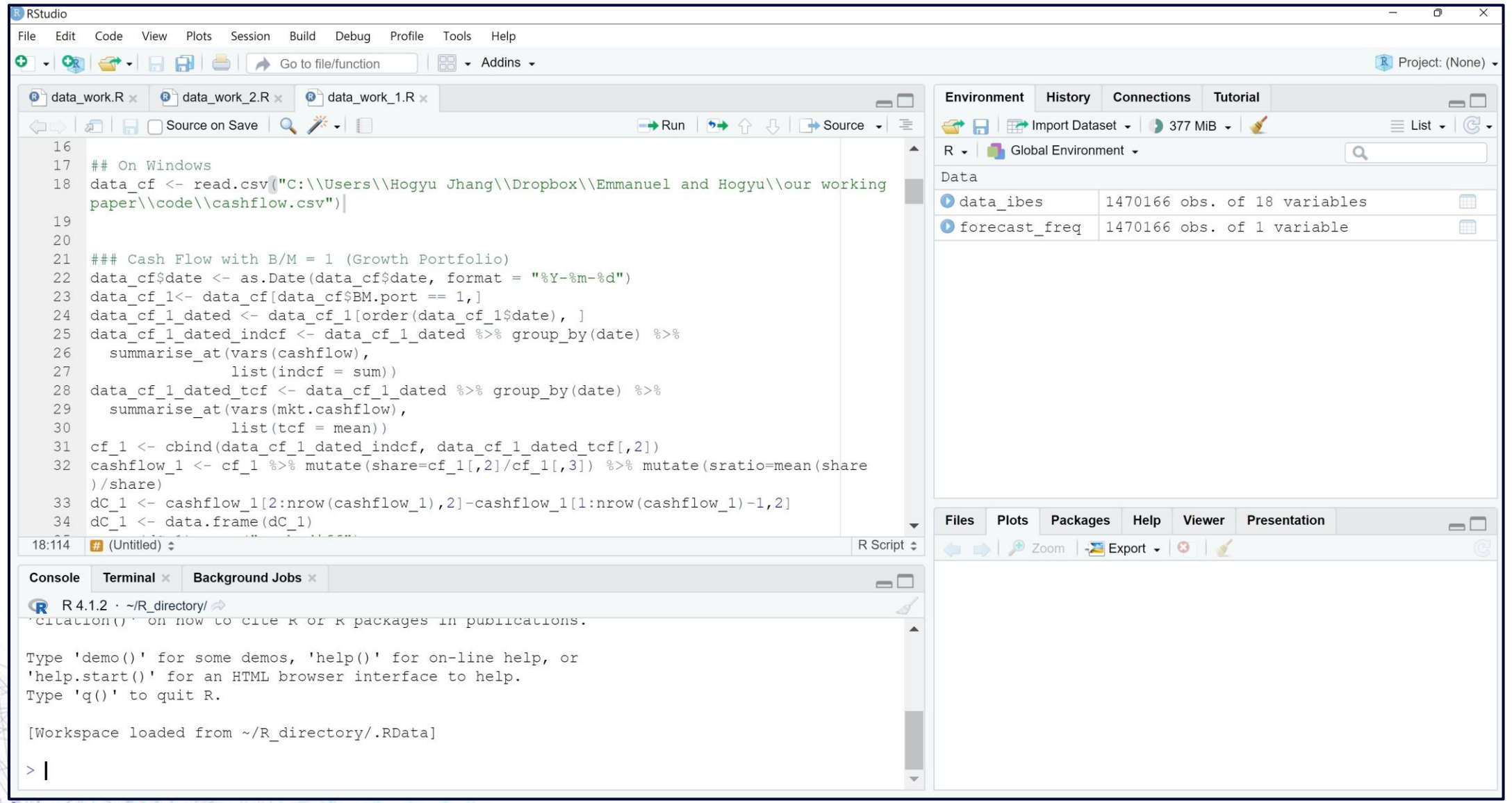
RStudio

- You can work directly in R, but but most users prefer a graphical interface. Most of times, using RStudio, -- an integrated development environment (IDE) -- is highly recommend. RStudio features:
  - A console
  - A powerful code/script editor featuring
  - special tools for plotting, viewing R objects and code history
  - cheatsheets for R programming
  - tab-completion for object names and function arguments (enough reason by itself!)

RStudio and Installation

- https://www.rstudio.com/products/rstudio/download
- A free version is enough

Rstudio and Installation



RStudio Script Editor

- we use the script editor to save our commands as a record of the steps we took to analyze our data. We can also issue R commands directly from the editor.
- `File > New File > R Script`
- Try 1+1 in the script editor.
- Save your R script with the name `mycode.R`.

Help for R and RStudio

- The `RStudio Help` menu menu contains links to many documents for help with both R (select `R Help`) and RStudio (see RStudio Docs and RStudio Community Forum)
- In RStudio, one can always use cheatsheet by selecting `Help` menu → `Cheatsheets` → `RStudio IDE Cheat Sheet`. You can also download the cheatsheet as a .pdf.

Environment: R packages

- Base R and most R packages are available for download from the Comprehensive R Archive Network (CRAN) https://www.cran.r-project.org.
- Base R comes with a number of basic data management, analysis, and graphical tools.
- However, R's power and flexibility lie in its array of packages (currently more than 15,000 on CRAN!)

Installing Packages

- To use packages, one must first install them using the `install.packages()` function, which typically downloads the package from CRAN and installs it for use.
- Use the argument dependencies=TRUE to load all other packages required by the targeted package.

```
Install.packages("dplyr", dependencies=TRUE)
Install.packages("ggplot2", dependencies=TRUE)
Install.packages("rmarkdown", dependencies=TRUE)
Install.packages("shiny", dependencies=TRUE)
```

Loading Packages

- After installing a package, we can load it into the R environment using the `library()` or `require()` functions, which more or less do the same thing.

- Functions and data structures within the package will then be available for use.

```
library(dplyr)
library(ggplot2)
library(shiny)
```

Package Tutorials

- Many packages include vignettes – longer, tutorial style guides for a package.

- To see a list of available vignettes for the packages that are loaded, use `vignette()` with no arguments.
  Then to view a vignette, place its name inside `vignette()`.

Basic R Coding

- Remember that we assign data to objects with <- or =.
- Character data (i.e. strings) are surrounded by " or '.
- In the script editor, create an object named a and assign it the character string "hello".
- Tip
  - The # character at the beginning of a line signifies a comment, which is not executed.

Functions and Help

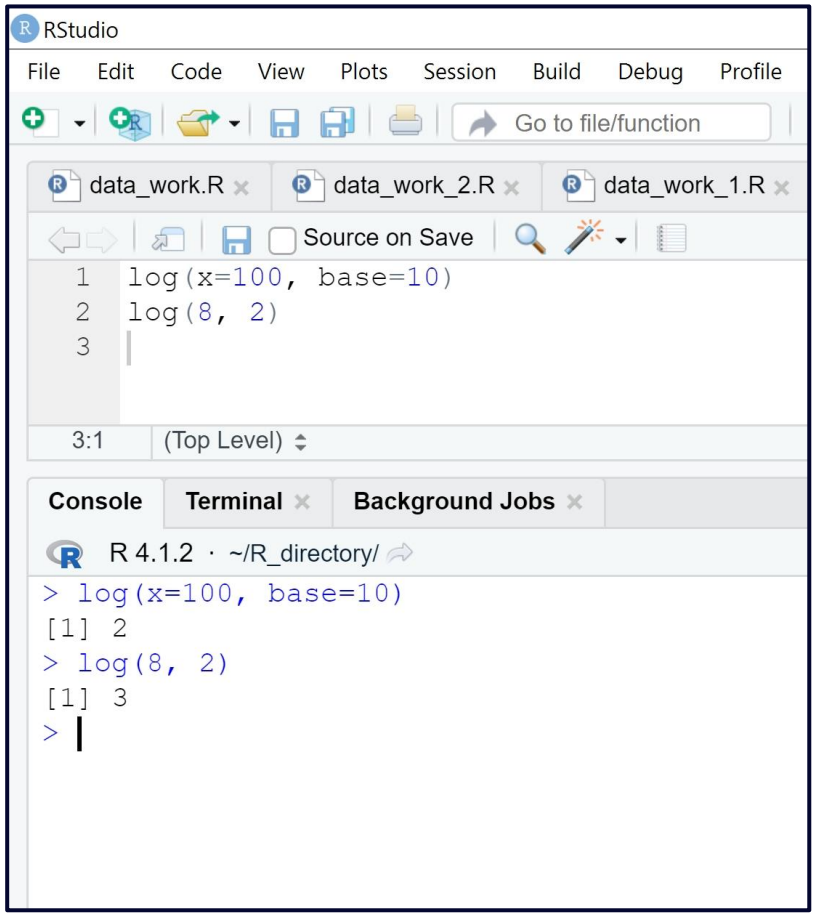
- Functions perform most of the work on data in R.

- Functions in R are much the same as they are in math - they perform some operation on an input and return some output. For example, the mathematical function $f(x) = x^2$, takes an input x, and returns its square. Similarly, the `mean()` function in R takes a vector of numbers and returns its mean.
- The inputs to functions are often referred to as arguments.
- Help files for R functions are accessed by preceding the name of the function with `?`.
- Try opening the help file for `log()` with the code `?log`.

Function Arguments

- Values for arguments to functions can be specified either by name or position.

- For `log()`, we see the first argument is x, the number whose log we want to take, and the second is `base`, the base of the logarithm.



Vectors

- Vectors, the fundamental data structure in R, are one-dimensional and homogeneous.

- A single variable can usually be represented by one of the following vector data types:
  - logical: TRUE or FALSE (1 or 0)
  - integer: integers only
    (represented by a number followed by L; e.g. 10L is the integer 10)
  - double: real numbers, also known as numeric
  - character: strings

Example

3	5	-2	24	1	0	2	1

apple	Orange	pear

4.17

TURE	FALSE	TURE	TURE

One-dimensional

Homogeneous

Integer, character, double, and logical vector

Creating Vectors

- The c() function combines values of common type together to form a vector.

```
1  first_vec <- c(1,3,5)
2  first_vec
3
```

3:1 (Top Level)

Console Terminal Background Jo

R 4.1.2 · ~/R_directory/

```
> first_vec <- c(1,3,5)
> first_vec
[1] 1 3 5
>
```

```
1  char_vec<-c("these", "are", "some", "words")
2  length(char_vec)
```

2:17 (Top Level)

Console Terminal Background Jobs

R 4.1.2 · ~/R_directory/

```
> char_vec<-c("these", "are", "some", "words")
> length(char_vec)
[1] 4
>
```

```
1  first_vec <- c(1,3,5)
2  first_vec > c(2,2,2)
```

2:21 (Top Level)

Console Terminal Background Jo

R 4.1.2 · ~/R_directory/

```
> first_vec <- c(1,3,5)
> first_vec > c(2,2,2)
[1] FALSE  TRUE  TRUE
>
```

Creating Vectors

- The c() function combines values of common type together to form a vector.

```
1  rep(0, times=3)
2  rep("abc", 4)
3
```

```
> rep(0, times=3)
[1] 0 0 0
> rep("abc", 4)
[1] "abc" "abc" "abc" "abc"
>
```

```
1  seq(from=1, to=5, by=2)
2  seq(10,0,-5)
3
```

```
> seq(from=1, to=5, by=2)
[1] 1 3 5
> seq(10,0,-5)
[1] 10  5  0
>
```

```
1  3:7
2  rep(seq(1,3,1), times=2)
3
```

```
> 3:7
[1] 3 4 5 6 7
> rep(seq(1,3,1), times=2)
[1] 1 2 3 1 2 3
>
```

Subsetting Vectors with []

```
1  a<-seq(10,1,-1)
2  a[2]
3  a[seq(1,5)]
4  a[c(1,3,4)]
5
```

5:1 (Top Level)

Console Terminal Background J

R 4.1.2 · ~/R_directory/

```
> a<-seq(10,1,-1)
> a[2]
[1] 9
> a[seq(1,5)]
[1] 10  9  8  7  6
> a[c(1,3,4)]
[1] 10  8  7
>
```

Conditional Selection

```
1  scores <- c(55, 24, 43, 10)
2  scores[c(FALSE, TRUE, TRUE, FALSE)]
3  scores < 30
4  scores[scores<30]
5
```

5:1 (Top Level)

Console | Terminal | Background Jobs

R 4.1.2 · ~/R_directory/

```
> scores <- c(55, 24, 43, 10)
> scores[c(FALSE, TRUE, TRUE, FALSE)]
[1] 24 43
> scores < 30
[1] FALSE  TRUE FALSE  TRUE
> scores[scores<30]
[1] 24 10
>
```