

KUDH Basics

第2回 統計ソフトウェア『R』 ワークショップ

① R/RStudioの使い方

藤本花音（京都大学）

2023.03.22.

トラブルがあれば質問してください

□トラブルを受講者で共有し，解決しながらすすめましょう

- 授業中，Rに関するトラブルがあれば遠慮なく教えてください
 - ・口頭での質問，zoomやSlackのチャット機能などでご連絡ください
 - ・Slackの方が丁寧なサポートが可能です
- みんなでトラブルとその解決方法を共有しましょう
 - ・オンサイトの方もオンラインの方も，ご自身のトラブルをzoomで画面共有してください
 - ・授業の一環として，トラブルの解決に努めます
 - ・深刻なトラブルな場合は，メイン講師の授業と同時並行して，サブ講師がトラブル解決に努めます
- もちろん，授業後や休憩時間に質問いただけてもOKです

はじめに

複雑すぎる世界を理解し，活動していくために

□対象から情報（データ）を取得して，分析していくことが重要

現象が生じる「世界」

- 生物現象
- 物理現象
- 社会現象
- 顧客の行動
など

観測
データ収集



データ

1001010		
01	1001010	
00	01	1001010
00	00	0100110
11	00	0000100
	11	0000001
		1101111

世界に関する
推測



研究をするうえで欠かせない「データ」

□データを用いてできること

●データ分析のモチベーション

- 目の前の現象を把握したい
- 現象の背後にあるメカニズムを解釈したい
- データから未来のことを予測したい
- データに基づいて意思決定や問題解決を行いたい

●必要なスキル・知識

- データの要約
- データの可視化
- 数理モデルの活用
- 統計学・機械学習の知識
- 扱おうとする現象についてのドメイン知識



因果関係？



統計ソフトウェア「R」

□Rとは？

- フリーの統計ソフトウェア
- 「R言語」とも呼ばれる
- The R Foundation for Statistical Computation (Vienna, Austria)が開発
- 幅広い統計解析手法をカバー
- 「パッケージ」による拡張



RStudioでRをもっと便利に扱う

□RStudio

- ・ Rのための統合開発環境
 - － 生のR + いろいろ便利機能

Welcome to RStudio

Software, education, and services for
the R community



RStudioの使い方

RStudioを起動する

□ R/Rstudioをインストール済みの前提で進めます

- 未インストールの人は過去動画を参考にインストールしてください
- https://www.youtube.com/watch?v=NMmD_3QvFsY

□ 『RStudio』 を起動してください (『R』 ではなく 『RStudio』)

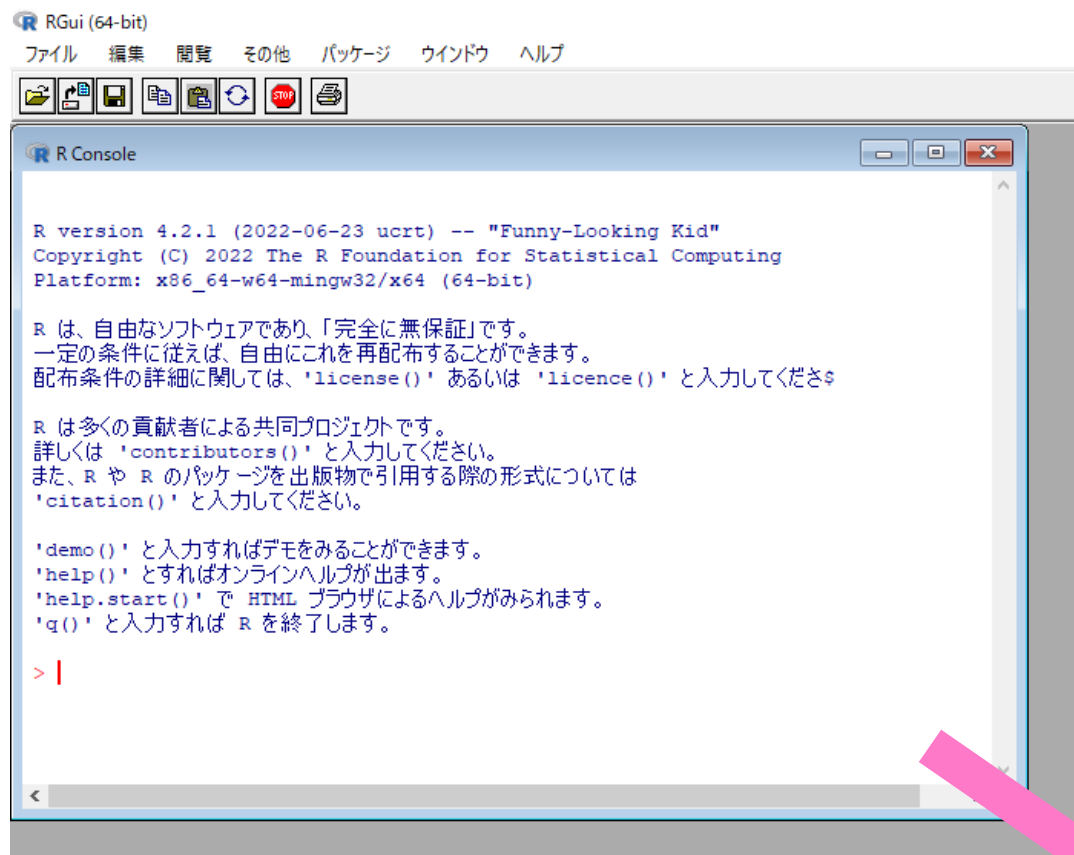
Welcome to RStudio

Software, education, and services for
the R community

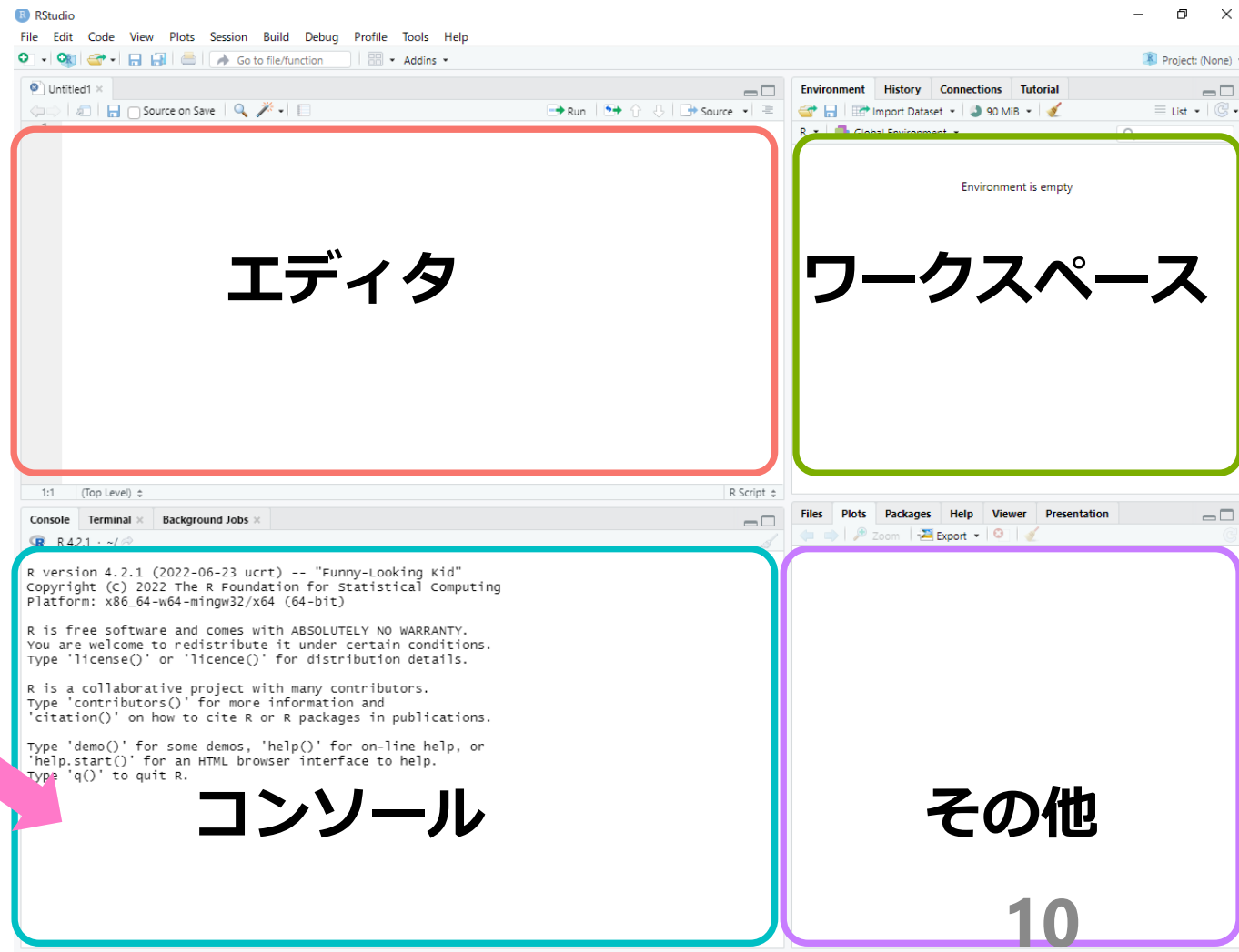


RStudioの起動画面

R

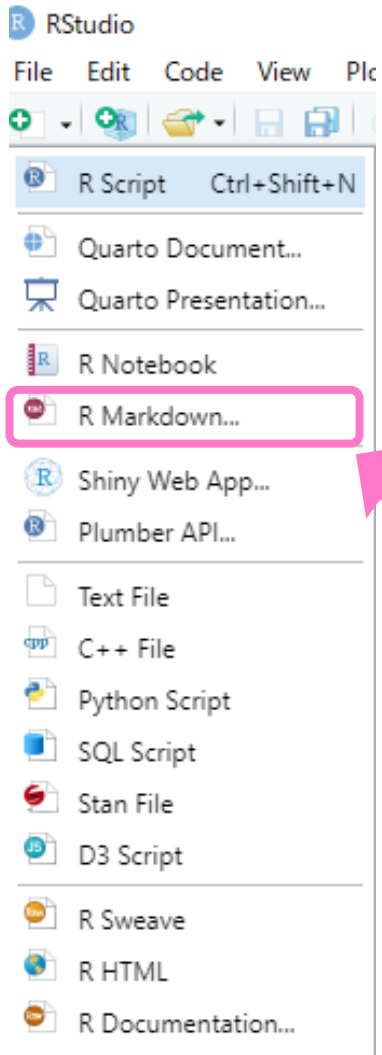


RStudio

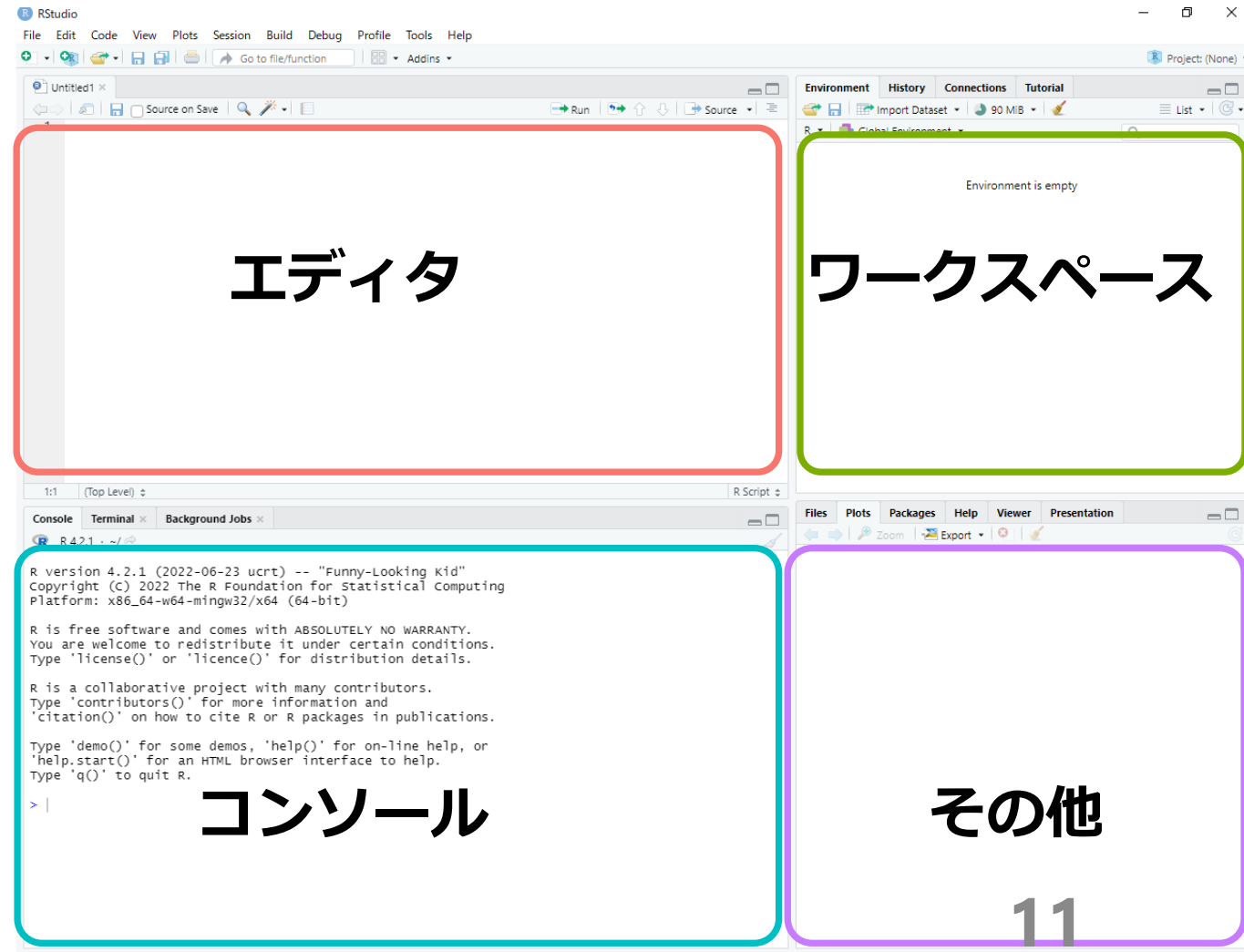


R Markdownを使ってコマンドを打ってみよう

□画面左上の「+」ボタンから「R Markdown...」をクリック



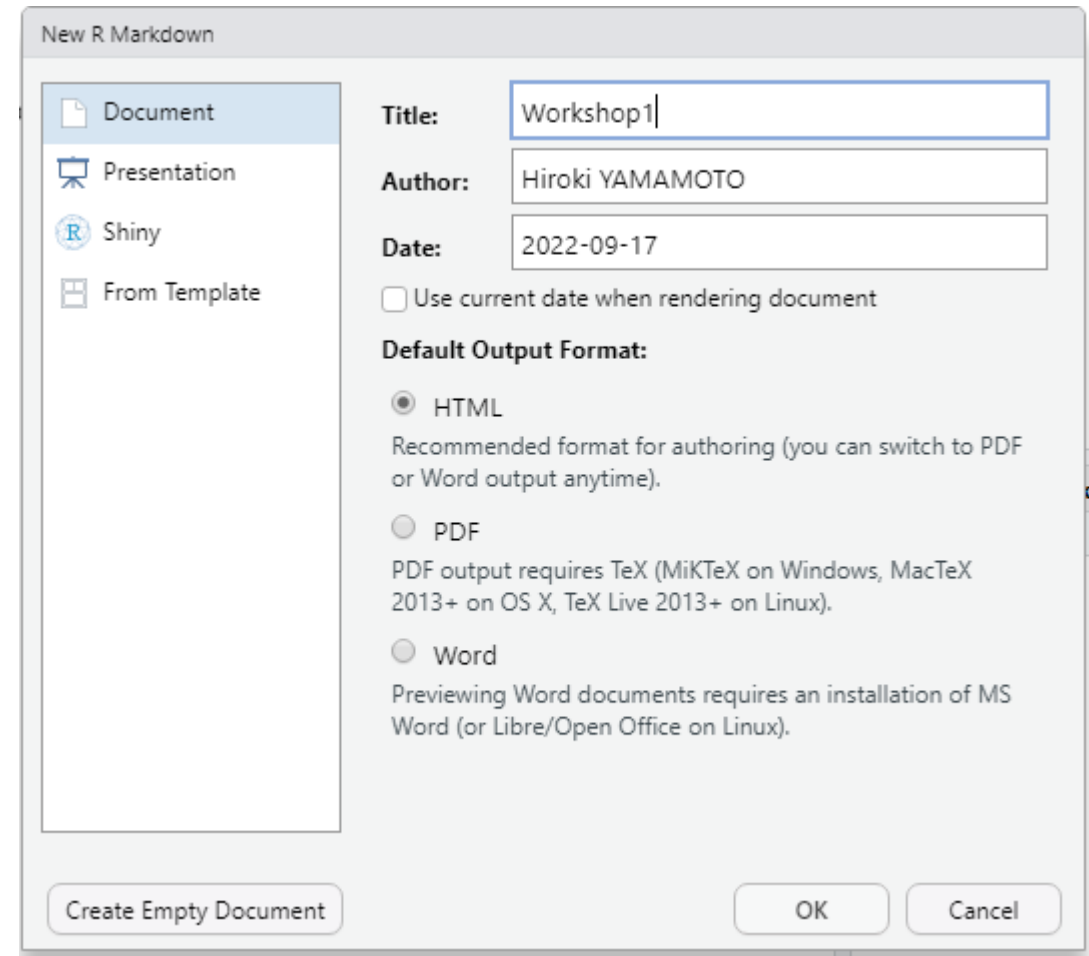
クリック



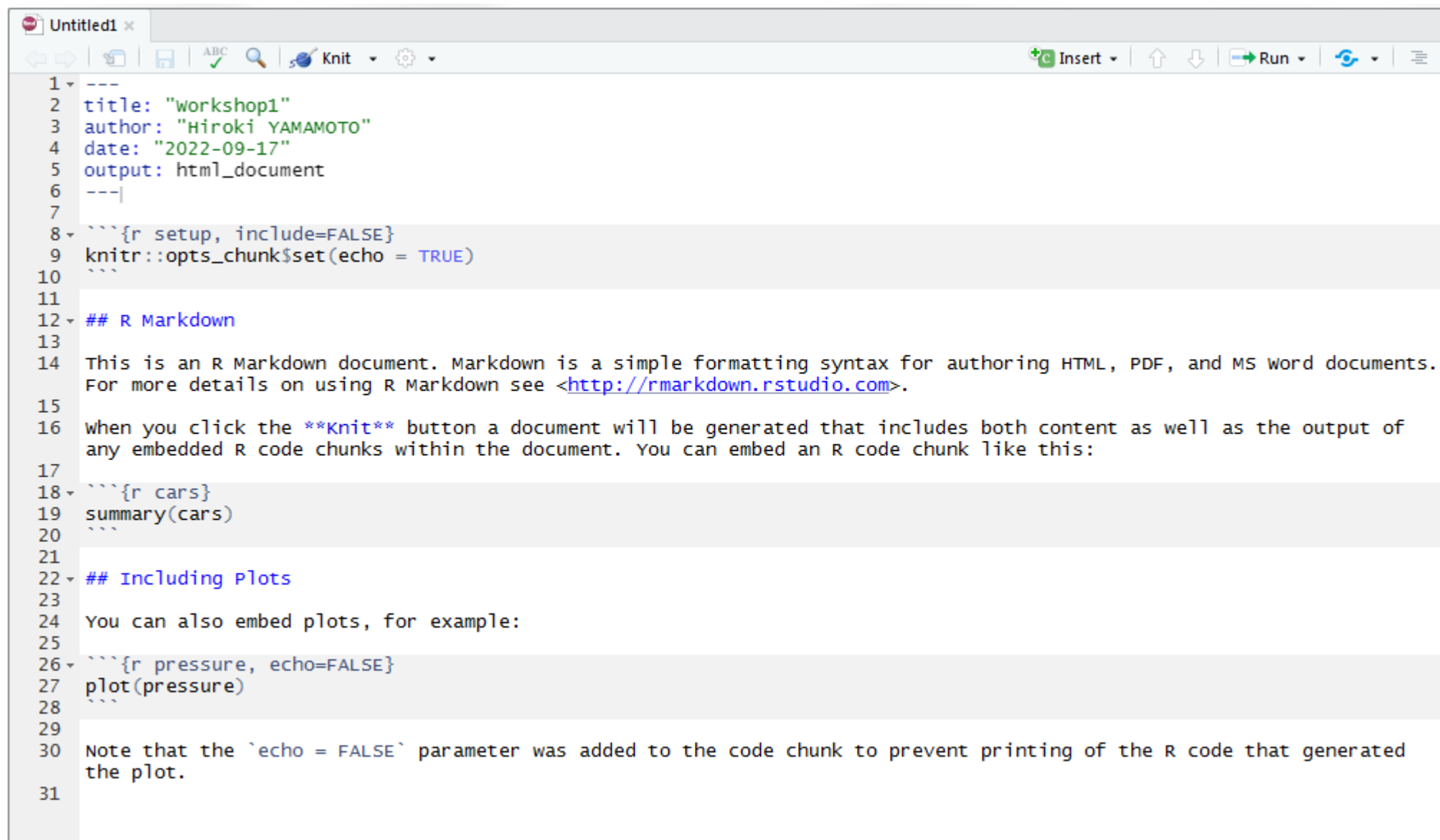
New R Markdownというウィンドウができる

□タイトルや著者欄など基礎情報を記入できる

- Title : 「いまから行う分析の名称」
– “Workshop1”としておきましょう。
- Author : 「分析者の名前」
– ご自身の名前を入力ください。
- Default Output FormatではHTMLを選択
- 「OK」をクリック



こんな画面が自動生成されます

A screenshot of a text editor window titled 'Untitled1'. The editor shows an R Markdown document with a YAML header, a code chunk for R setup, a text block about R Markdown, another code chunk for R code, a text block about including plots, and a final code chunk for an R plot. The document is numbered 1 through 31. The code is as follows:

```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents.
15 For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
16
17 When you click the Knit button a document will be generated that includes both content as well as the output of
18 any embedded R code chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
25
26 You can also embed plots, for example:
27
28 ```{r pressure, echo=FALSE}
29 plot(pressure)
30 ```
31
32 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated
33 the plot.
```

灰色の領域はコードチャンクと呼ばれます

```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents.
15 For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
16
17 When you click the Knit button a document will be generated that includes both content as well as the output of
18 any embedded R code chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
25
26 You can also embed plots, for example:
27
28 ```{r pressure, echo=FALSE}
29 plot(pressure)
30 ```
31
32 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated
33 the plot.
```

コードチャンク

コードチャンク

コードチャンク

先頭6行はyamlヘッダと呼ばれます

```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents.
15 For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
16
17 When you click the Knit button a document will be generated that includes both content as well as the output of
18 any embedded R code chunks within the document. You can embed an R code chunk like this:
19
20 ```{r cars}
21 summary(cars)
22 ```
23
24 ## Including Plots
25
26 You can also embed plots, for example:
27
28 ```{r pressure, echo=FALSE}
29 plot(pressure)
30 ```
31
32 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated
33 the plot.
```

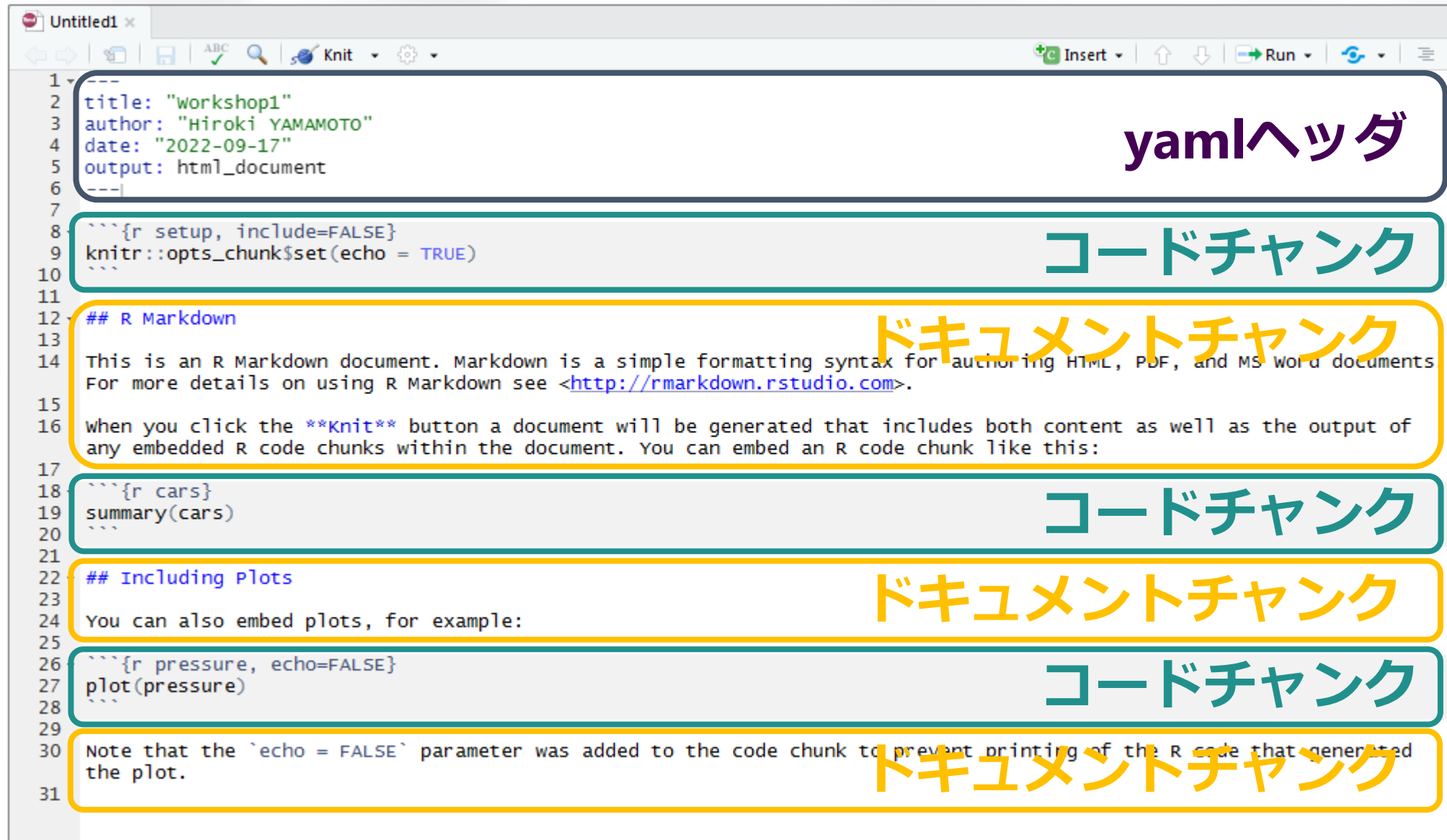
yamlヘッダ

コードチャンク

コードチャンク

コードチャンク

コードチャンクの間はドキュメントチャンク



The image shows a screenshot of an R Markdown document in a code editor. The document is titled "Untitled1" and contains several code chunks and document sections. Annotations are provided for each section:

- YAML Header:** Lines 1-6 contain a YAML header for a workshop. A purple box labeled "yamlヘッダ" highlights this section.
- Code Chunk:** Lines 8-10 contain an R code chunk for setting up the environment. A teal box labeled "コードチャンク" highlights this section.
- Document Chunk:** Lines 12-16 contain a document section titled "## R Markdown" with introductory text. A yellow box labeled "ドキュメントチャンク" highlights this section.
- Code Chunk:** Lines 18-20 contain an R code chunk for summarizing cars. A teal box labeled "コードチャンク" highlights this section.
- Document Chunk:** Lines 22-25 contain a document section titled "## Including Plots" with text about embedding plots. A yellow box labeled "ドキュメントチャンク" highlights this section.
- Code Chunk:** Lines 26-28 contain an R code chunk for plotting pressure. A teal box labeled "コードチャンク" highlights this section.
- Document Chunk:** Lines 30-31 contain a document section with a note about the 'echo' parameter. A yellow box labeled "ドキュメントチャンク" highlights this section.

```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
15
16 When you click the Knit button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
17
18 ```{r cars}
19 summary(cars)
20 ```
21
22 ## Including Plots
23
24 You can also embed plots, for example:
25
26 ```{r pressure, echo=FALSE}
27 plot(pressure)
28 ```
29
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
31
```


Rmarkdownを使ってコードをうってみよう

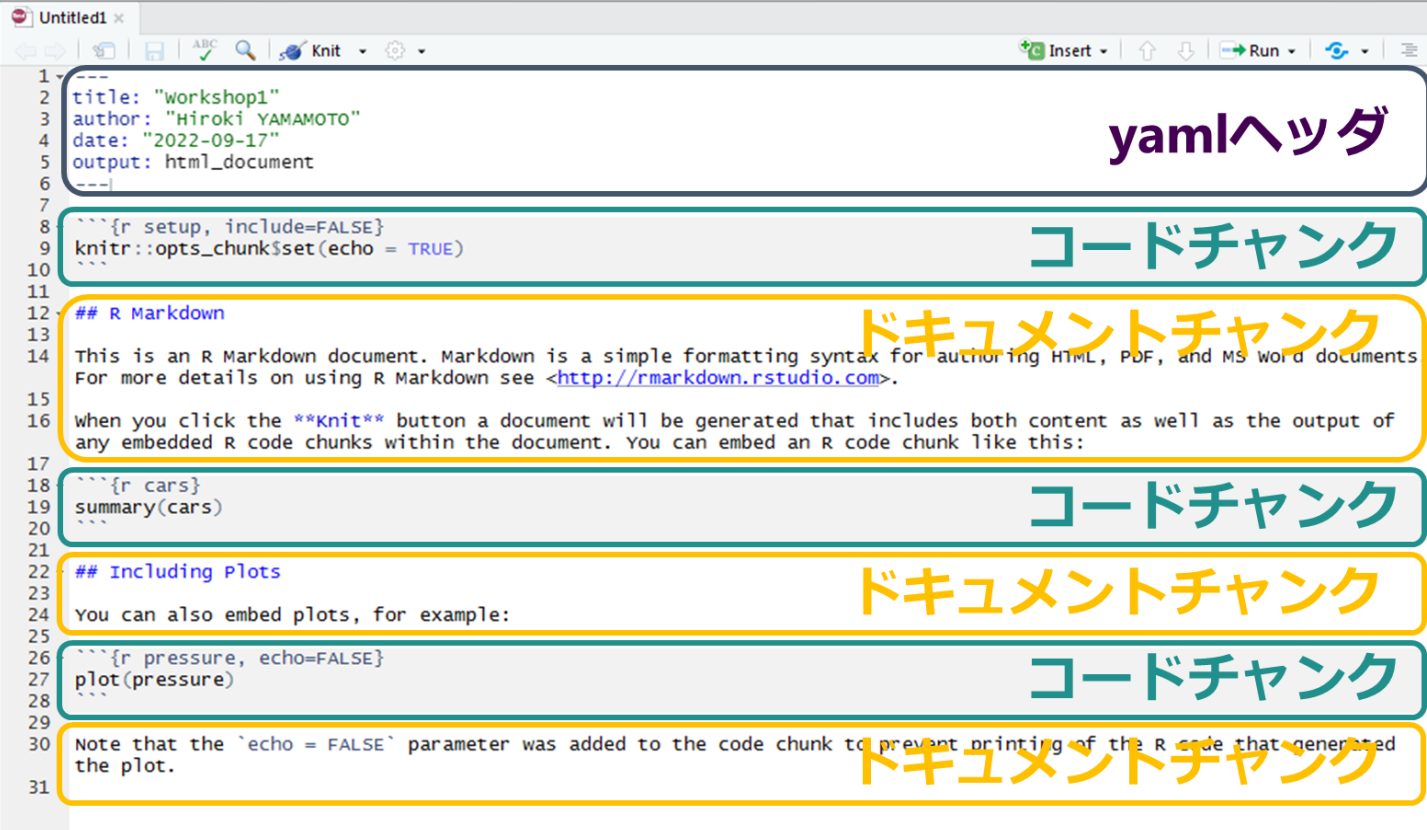
□コードチャンクとyamlヘッダ

・コードチャンク

- Rへのコマンドを書く箇所
- 「``{r}``」から「``」まで
- 「`」が3つ並んでないと機能しないので注意

・yamlヘッダ

- 「---」から「---」まで
- 全体的な設定などを扱う箇所
- この授業ではいじりません



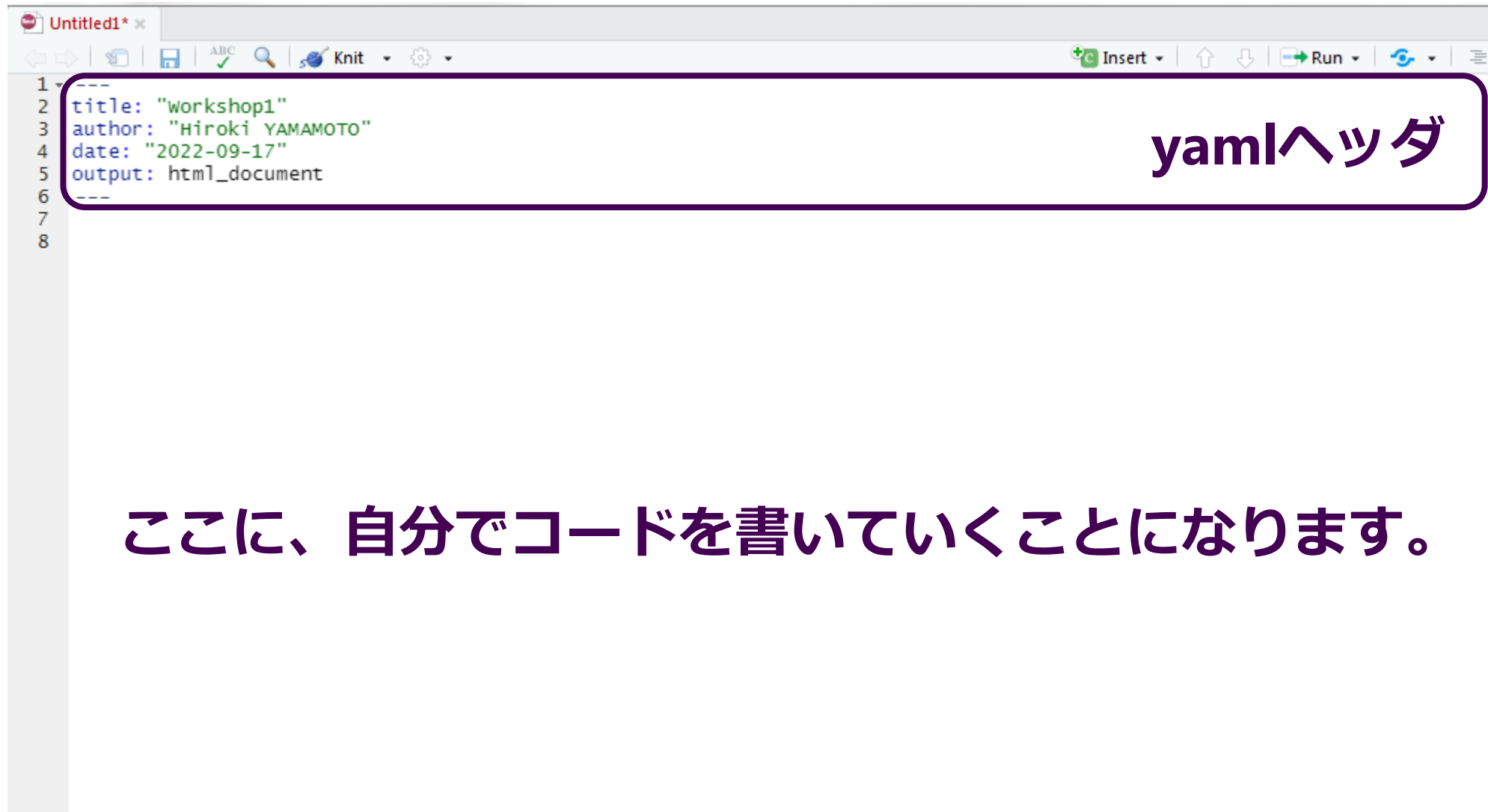
The screenshot shows an R Markdown document titled 'Untitled1' in RStudio. The document content is as follows:

```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
15
16 When you click the **knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
17
18 ```{r cars}
19 summary(cars)
20 ```
21
22 ## Including Plots
23
24 You can also embed plots, for example:
25
26 ```{r pressure, echo=FALSE}
27 plot(pressure)
28 ```
29
30 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
31
```

Annotations on the right side of the screenshot:

- yamlヘッダ** (purple text) points to the YAML header (lines 2-5).
- コードチャンク** (teal text) points to the first R code chunk (lines 8-10).
- ドキュメントチャンク** (orange text) points to the R Markdown text block (lines 12-16).
- コードチャンク** (teal text) points to the second R code chunk (lines 18-20).
- ドキュメントチャンク** (orange text) points to the text block about including plots (lines 22-24).
- コードチャンク** (teal text) points to the third R code chunk (lines 26-28).
- ドキュメントチャンク** (orange text) points to the final note (lines 30-31).

yamlヘッダより下の部分をDeleteしましょう



Ctrl+Alt+I でコードチャンクを挿入

□コードチャンクの挿入

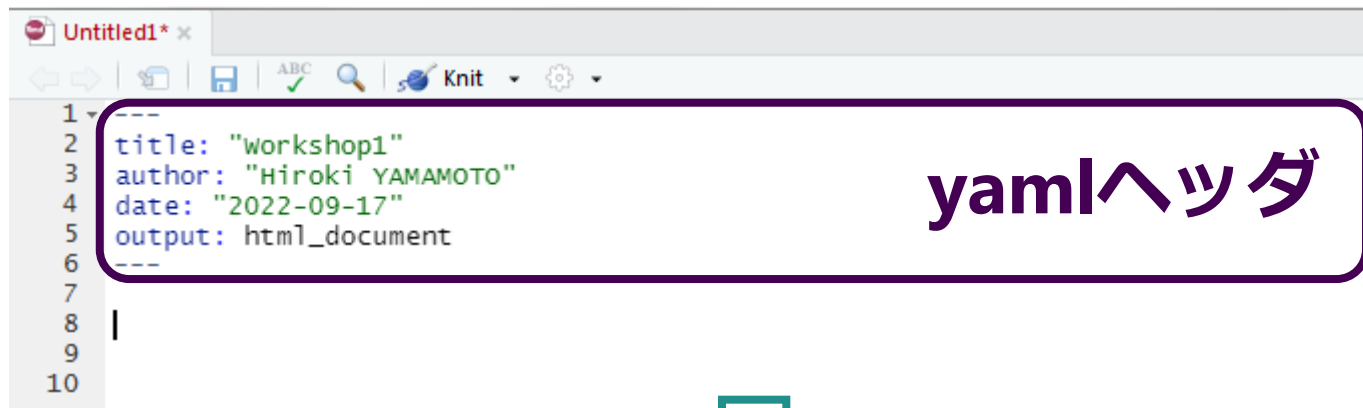
① yamlヘッダの下部に
カーソルを置いて

② Ctrl + Alt + I

※macOSではCommand + Option + I

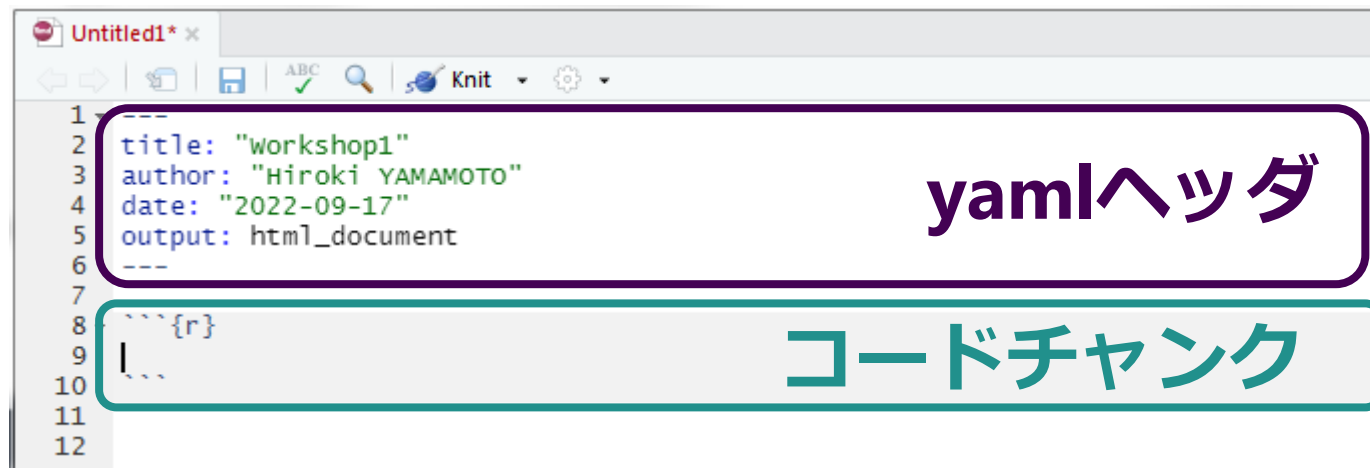
③ チャンクがでてきた！

ここにコードをうっていきます



```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 |
9
10
```

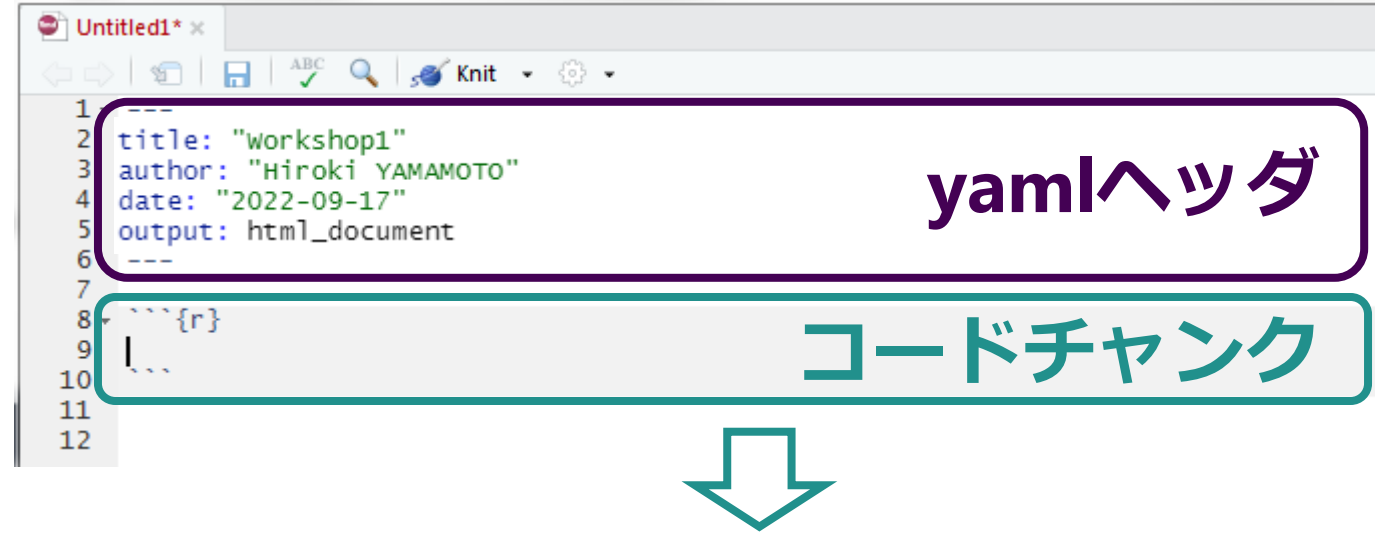
Ctrl+Alt+I



```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 {r}
9 |
10
11
12
```

コードチャンクにコードをうちましよう

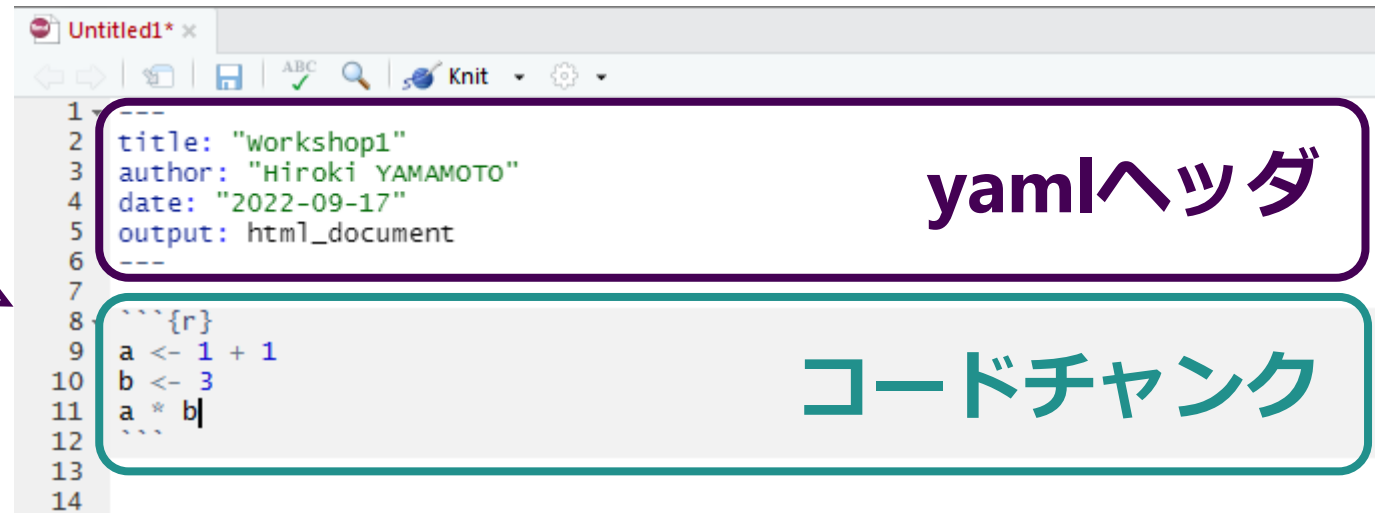
```
```\r\na <- 1 + 1\r\nb <- 3\r\na*b\r\n```
```



The screenshot shows a Knit editor window titled 'Untitled1\*'. The code is as follows:

```
1 ---\r\n2 title: "workshop1"\r\n3 author: "Hiroki YAMAMOTO"\r\n4 date: "2022-09-17"\r\n5 output: html_document\r\n6 ---\r\n7 \r\n8 ```{r}\r\n9 |\r\n10 \r\n11 \r\n12
```

A purple box highlights the first six lines, labeled 'yamlヘッダ'. A teal box highlights lines 8-10, labeled 'コードチャンク'. A large teal arrow points down to the next screenshot.

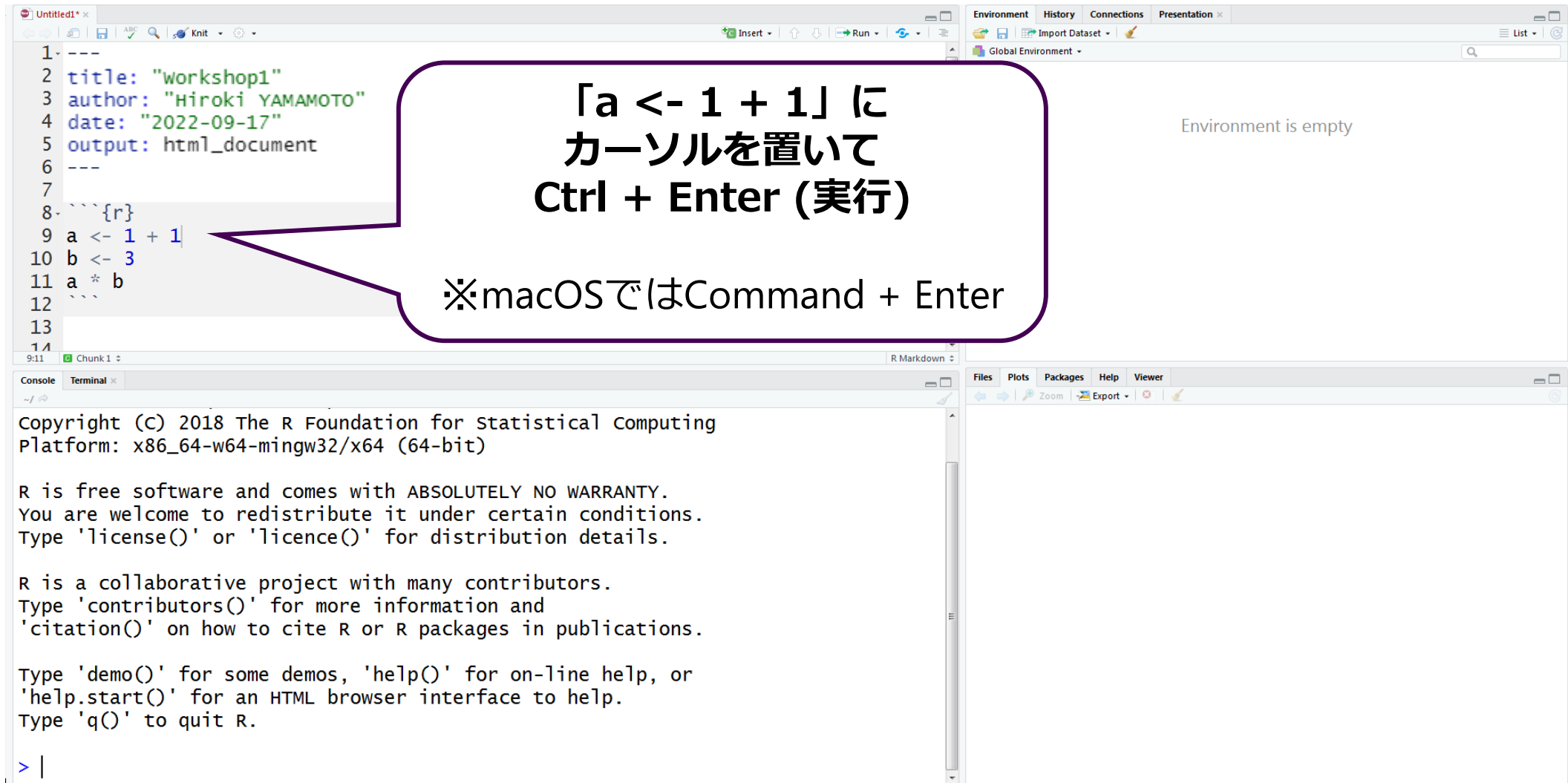


The screenshot shows the same Knit editor window after the code chunk has been filled with R code:

```
1 ---\r\n2 title: "workshop1"\r\n3 author: "Hiroki YAMAMOTO"\r\n4 date: "2022-09-17"\r\n5 output: html_document\r\n6 ---\r\n7 \r\n8 ```{r}\r\n9 a <- 1 + 1\r\n10 b <- 3\r\n11 a * b|\r\n12 \r\n13 \r\n14
```

The purple box for the 'yamlヘッダ' remains the same. The teal box for the 'コードチャンク' now contains the R code from the callout box, with a cursor at the end of the last line.

# Ctrl+Enterでカーソル行のコードを実行



The screenshot shows the RStudio interface. The editor window contains R code with a cursor on line 9: `a <- 1 + 1`. A purple callout box points to this line with the text: 「a <- 1 + 1」にカーソルを置いて Ctrl + Enter (実行). Below this, it says ※macOSではCommand + Enter. The environment pane on the right shows 'Environment is empty'. The console at the bottom displays the R startup message.

```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 ```{r}
9 a <- 1 + 1
10 b <- 3
11 a * b
12 ```
13
14
```

Environment is empty

Copyright (C) 2018 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)

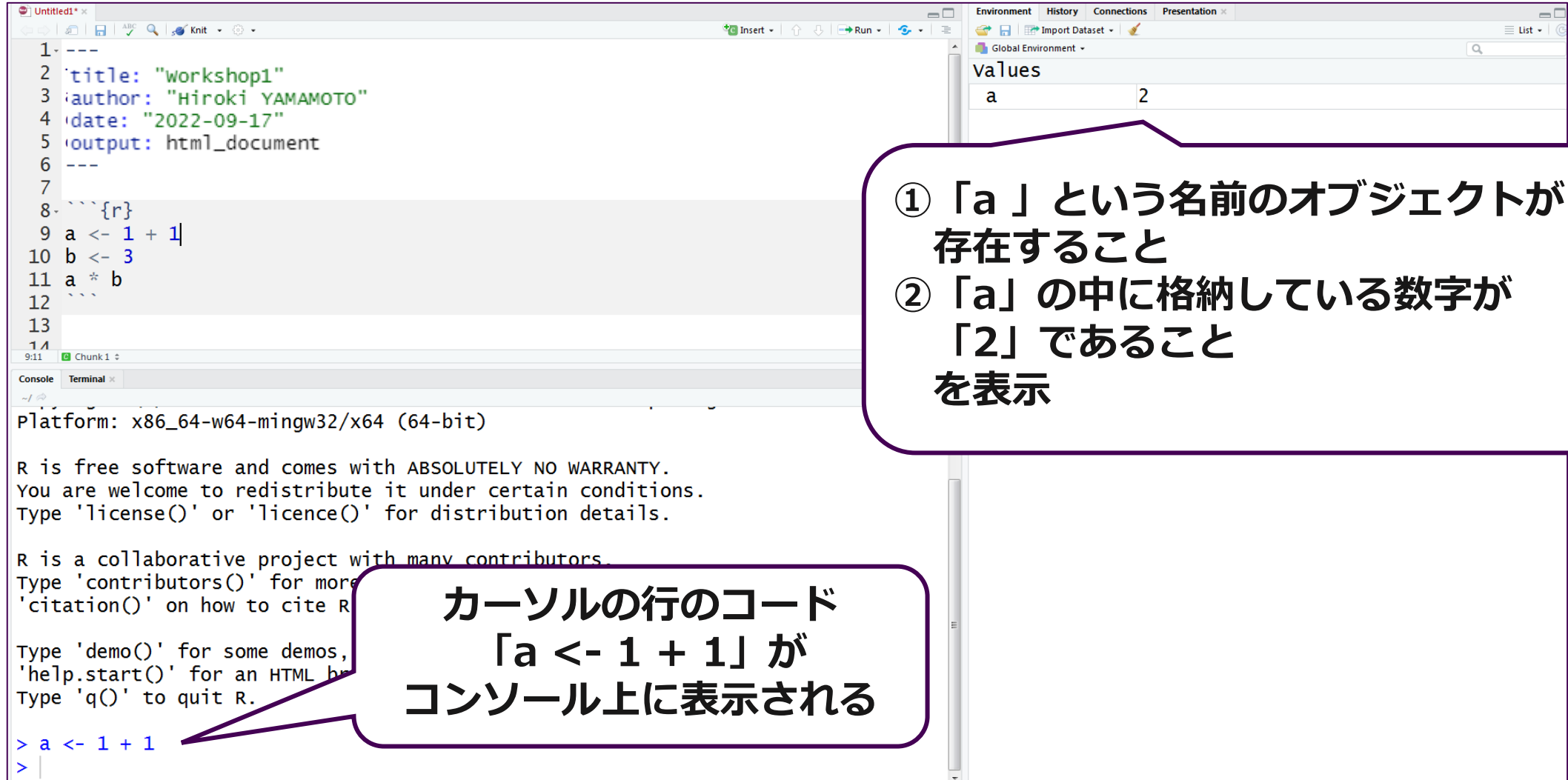
R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.

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.

> |

# Ctrl+Enterでカーソル行のコードを実行



The screenshot displays the RStudio interface with the following components:

- Script Editor:** Contains R code for a workshop document and a calculation. The code is as follows:

```
1 ----
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ----
7
8 {r}
9 a <- 1 + 1
10 b <- 3
11 a * b
12
13
14
```
- Console:** Shows the R startup message and the execution of the code. The output is as follows:

```
Platform: x86_64-w64-mingw32/x64 (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.

R is a collaborative project with many contributors.
Type 'contributors()' for more
'citation()' on how to cite R.

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

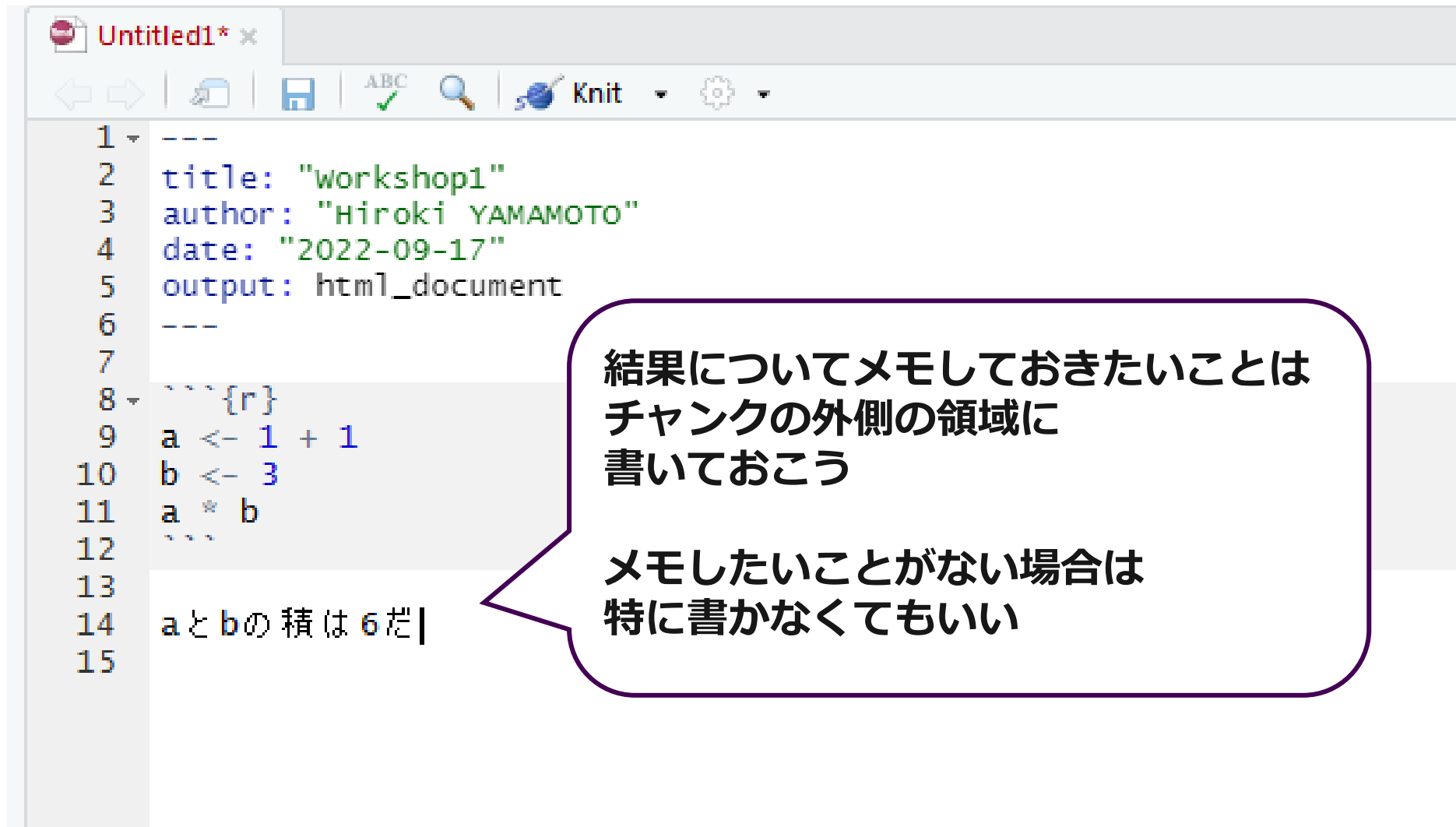
> a <- 1 + 1
>
```
- Global Environment:** Shows the variable 'a' with the value 2.

Two callouts highlight key information:

- ① 「a」という名前のオブジェクトが存在すること
- ② 「a」の中に格納している数字が「2」であることを表示

A callout points to the line `a <- 1 + 1` in the script editor, stating: カースルの行のコード「a <- 1 + 1」がコンソール上に表示される

# 結果をドキュメントチャンクに記述



The screenshot shows a Knit editor window titled "Untitled1\* x". The editor contains R code for a document chunk. The code is as follows:

```
1 ---
2 title: "workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7
8 ```{r}
9 a <- 1 + 1
10 b <- 3
11 a * b
12 ```
13
14 aとbの積は6だ|
15
```

A callout box with a purple border contains the following text:

結果についてメモしておきたいことは  
チャンクの外側の領域に  
書いておこう

メモしたいことがない場合は  
特に書かなくてもいい

# RStudio+Rmarkdownの基本的な使い方

## □RStudioとRmarkdownを用いたコーディングのワークフロー

- ・ 前準備

- (a) エディタにRmarkdown画面を生成

- (b) yamlヘッダを残して下部をDelete

- ・ コードをかく（以下の①～⑤を繰り返す）

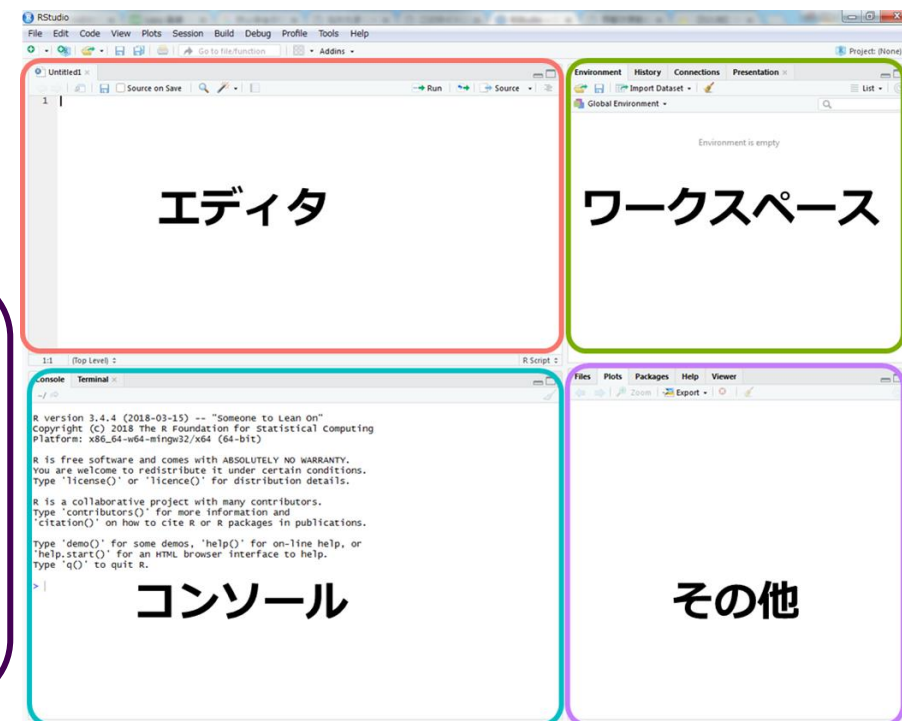
- ①コードチャンクを挿入(Ctrl+Alt+I)

- ②コードチャンクにコードをかく

- ③カーソルの行のコードを実行(Ctrl+Enter)

- ④コンソール・ワークスペースで出力を確認

- ⑤メモをドキュメントチャンクに記述





# 覚えておく と便利なショートカット (Windows)

## ●Ctrl + Enter

- ・カーソルのある行を実行する

## ●Ctrl + Alt + I

- ・コードチャンクを挿入する

## ●Ctrl + Alt + C

- ・カーソルのあるコードチャンクのコードをすべて実行する。

## ●Ctrl + Alt + R

- ・Rmarkdownファイル内にあるコードチャンクをすべて実行する。

# 覚えておく と便利なショートカット (macOS)

## ●Command+ Enter

- ・カーソルのある行を実行する

## ●Command + Option + I

- ・コードチャンクを挿入する

## ●Command + Option + C

- ・カーソルのあるコードチャンクのコードをすべて実行する。

## ●Command + Option + R

- ・Rmarkdownファイル内にあるコードチャンクをすべて実行する。

# データフレームの操作

# Rの基本操作

## □基本的な計算

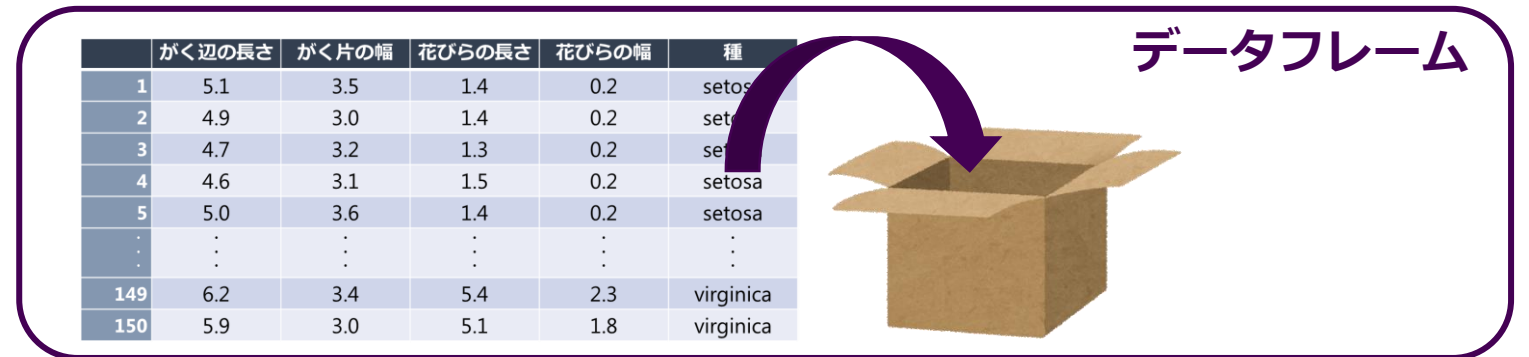
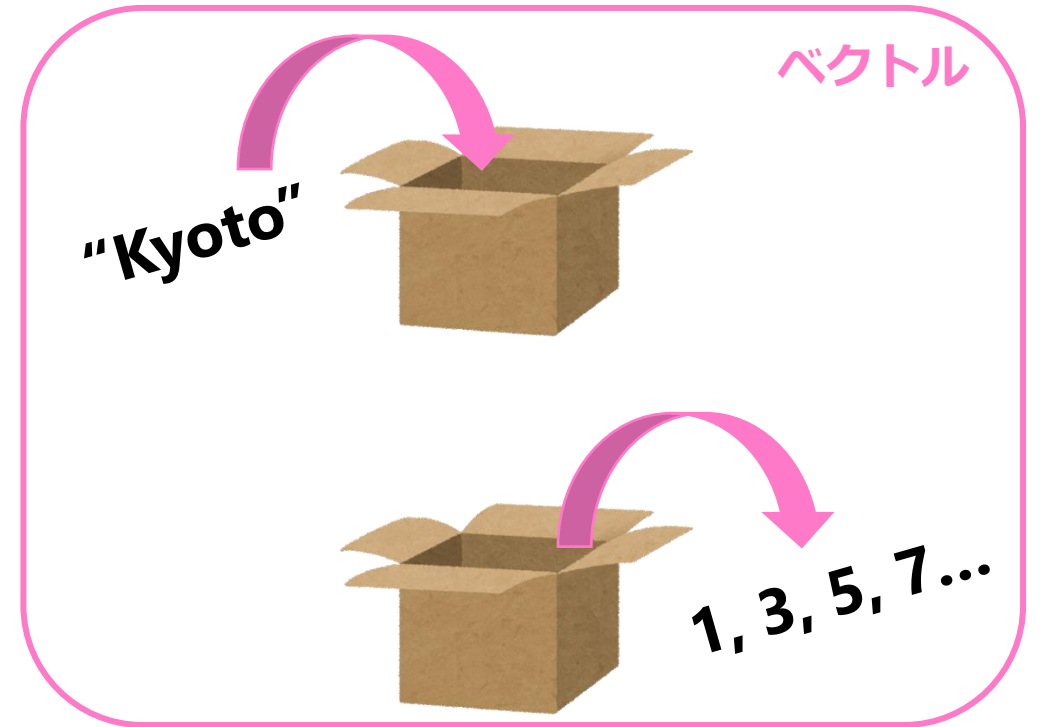
記号	意味
+	足し算
-	引き算
*	掛け算
/	割り算
^	べき乗

```
> 10+10
[1] 20
>
> 10-10
[1] 0
>
> 10*10
[1] 100
>
> 10/10
[1] 1
>
> 10^2
[1] 100
>
> |
```

# オブジェクト：データを格納する「箱」

## □Rのオブジェクト

- 数字・文字列などを格納したり、一部を取り出し・変更したりできる「箱」
- 格納するデータの種類によって、「箱」の種類も変わる。
  - ・ベクトル
  - ・データフレーム
  - ・マトリックス
  - ・リスト



# オブジェクト：データを格納する「箱」

## □オブジェクトへの代入

- ・「<-」でオブジェクトへの代入を表す。

```
> a<-1+1
```

```
>
```

```
> a
```

```
[1] 2
```

```
>
```

```
> b<-3
```

```
>
```

```
> a*b
```

```
[1] 6
```

```
> |
```

➤ 「1 + 1」という計算に「a」という名前をつける

➤ 名前をつけた後で「a」を呼び出すと、「1+1」という計算の結果（つまり2）が返される

➤ 「3」という数値に「b」という名前をつける

➤ 「a\*b」という計算を行うと、「(1+1)×3」という計算の結果（つまり6）が返される

# データフレーム

## □ サンプルデータ iris

iris

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 1	5.1	3.5	1.4	0.2	setosa
## 2	4.9	3.0	1.4	0.2	setosa
## 3	4.7	3.2	1.3	0.2	setosa
## 4	4.6	3.1	1.5	0.2	setosa
## 5	5.0	3.6	1.4	0.2	setosa
## 6	5.4	3.9	1.7	0.4	setosa
## 7	4.6	3.4	1.4	0.3	setosa
## 8	5.0	3.4	1.5	0.2	setosa
## 9	4.4	2.9	1.4	0.2	setosa
## 10	4.9	3.1	1.5	0.1	setosa
## 11	5.4	3.7	1.5	0.2	setosa
## 12	4.8	3.4	1.6	0.2	setosa

列名

列

行

それぞれの升目をセルという

# iris : フィッシャーのアヤメのデータ

## □3種のアヤメについて、がく片・花びらの長さ・幅を計測したデータ

・ setosa ・ versicolor ・ virginica 各50個体ずつ（計150個体）

	がく辺の長さ	がく片の幅	花びらの長さ	花びらの幅	種
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
⋮	⋮	⋮	⋮	⋮	⋮
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica

量的変数

量的変数

量的変数

量的変数

質的変数





# データフレームの先頭6行を表示する

## □head(df)

```
head(iris)
```

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 1	5.1	3.5	1.4	0.2	setosa
## 2	4.9	3.0	1.4	0.2	setosa
## 3	4.7	3.2	1.3	0.2	setosa
## 4	4.6	3.1	1.5	0.2	setosa
## 5	5.0	3.6	1.4	0.2	setosa
## 6	5.4	3.9	1.7	0.4	setosa

# データフレームの基礎情報を表示する

## □summary(df)

```
summary(iris)
```

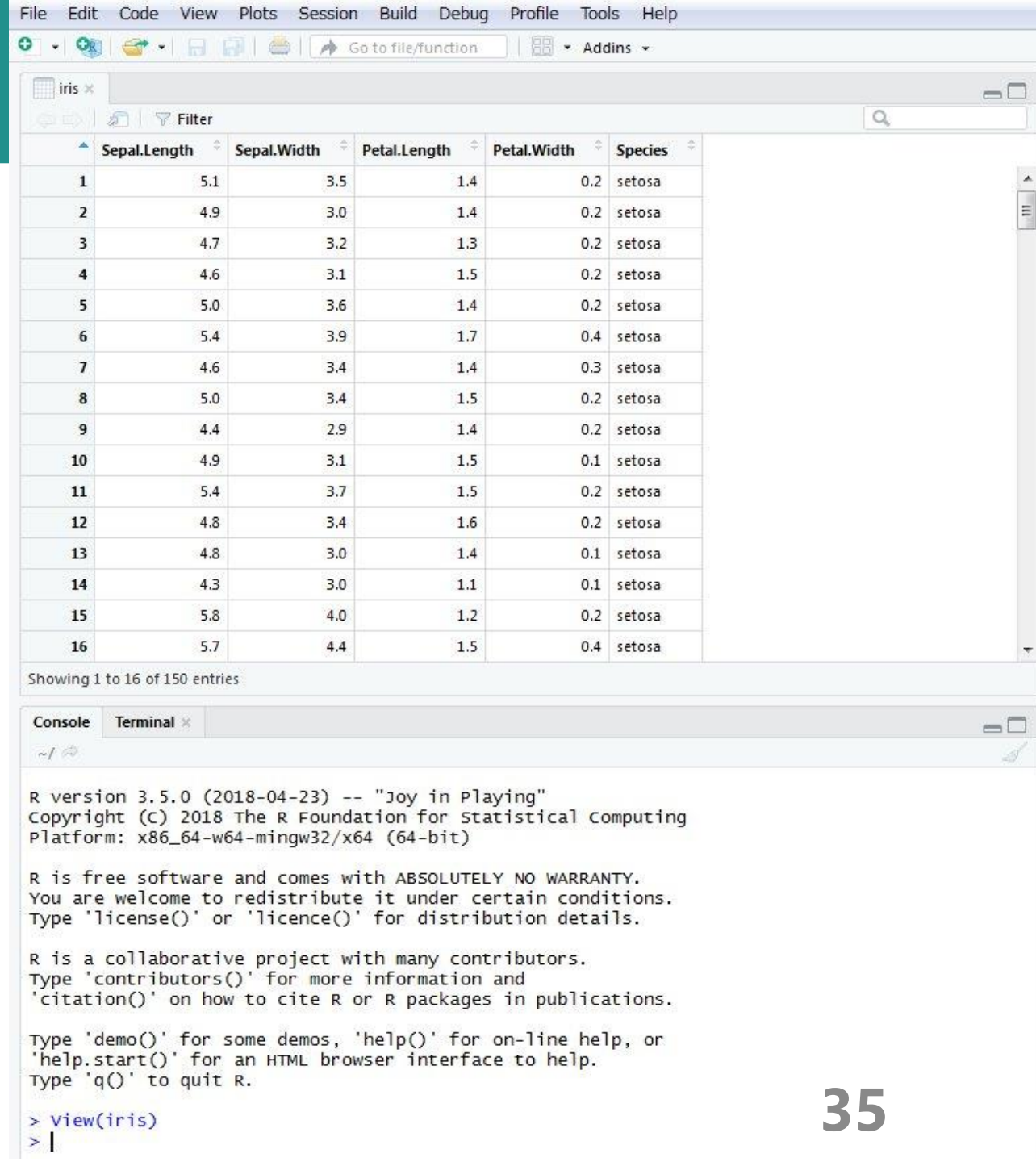
```
Sepal.Length Sepal.Width Petal.Length Petal.Width
Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100
1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300
Median :5.800 Median :3.000 Median :4.350 Median :1.300
Mean :5.843 Mean :3.057 Mean :3.758 Mean :1.199
3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800
Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500
##
Species
setosa :50
versicolor:50
virginica :50
##
```

※数値の列では最小値・平均値・最大値・四分位数などの要約統計量が、  
文字列の列では、格納されている各文字列の観測値の個数が表示される

# 全データを表示する

## □View(df)

- ・エディタに  
データ一覧が表示される。



The screenshot shows the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The toolbar contains icons for adding files, saving, and other functions. The main window displays the 'iris' dataset in a table view. The table has columns for Sepal.Length, Sepal.Width, Petal.Length, Petal.Width, and Species. The data is sorted by Species, with 'setosa' at the top. The bottom pane shows the R console with the output of the 'view(iris)' command, which displays the R version and copyright information.

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa

Showing 1 to 16 of 150 entries

Console Terminal

```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (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.

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.

> view(iris)
> |
```

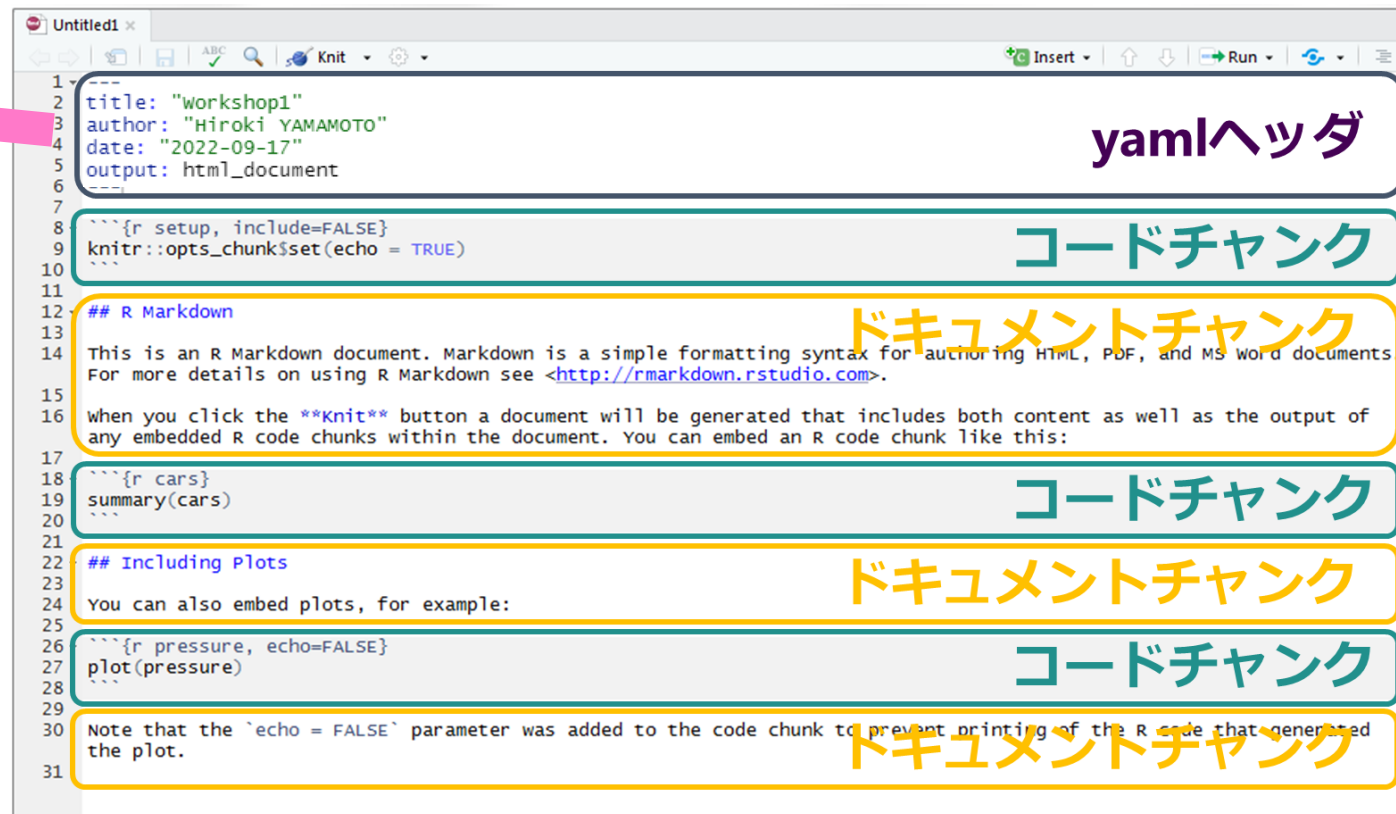
# 分析レポートの生成

# RmarkdownをKnitしよう

## □Windows上部のKnitボタンをクリック



Knitボタン



```
1 title: "workshop1"
2 author: "Hiroki YAMAMOTO"
3 date: "2022-09-17"
4 output: html_document
5 ---
6 ---
7 ---
8 {r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ---
11 ---
12 ## R Markdown
13 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
14 ---
15 when you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
16 ---
17 {r cars}
18 summary(cars)
19 ---
20 ---
21 ## Including Plots
22 You can also embed plots, for example:
23 ---
24 {r pressure, echo=FALSE}
25 plot(pressure)
26 ---
27 ---
28 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
29 ---
30 ---
31 ---
```

yamlヘッダ

コードチャンク

ドキュメントチャンク

コードチャンク

ドキュメントチャンク

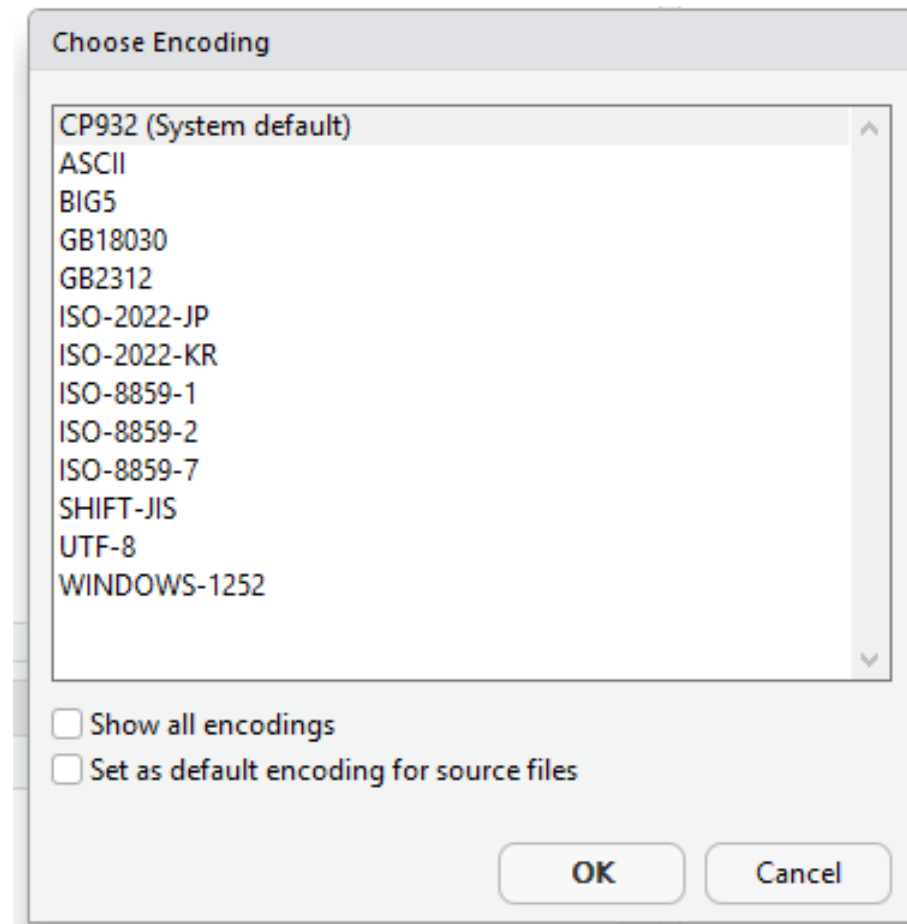
コードチャンク

ドキュメントチャンク

# RmarkdownをKnitしよう

## □Choose Encogingウィンドウがでてくる

- Windowsの場合
  - CP932を選択
- Mac OSの場合
  - UTF-8を選択



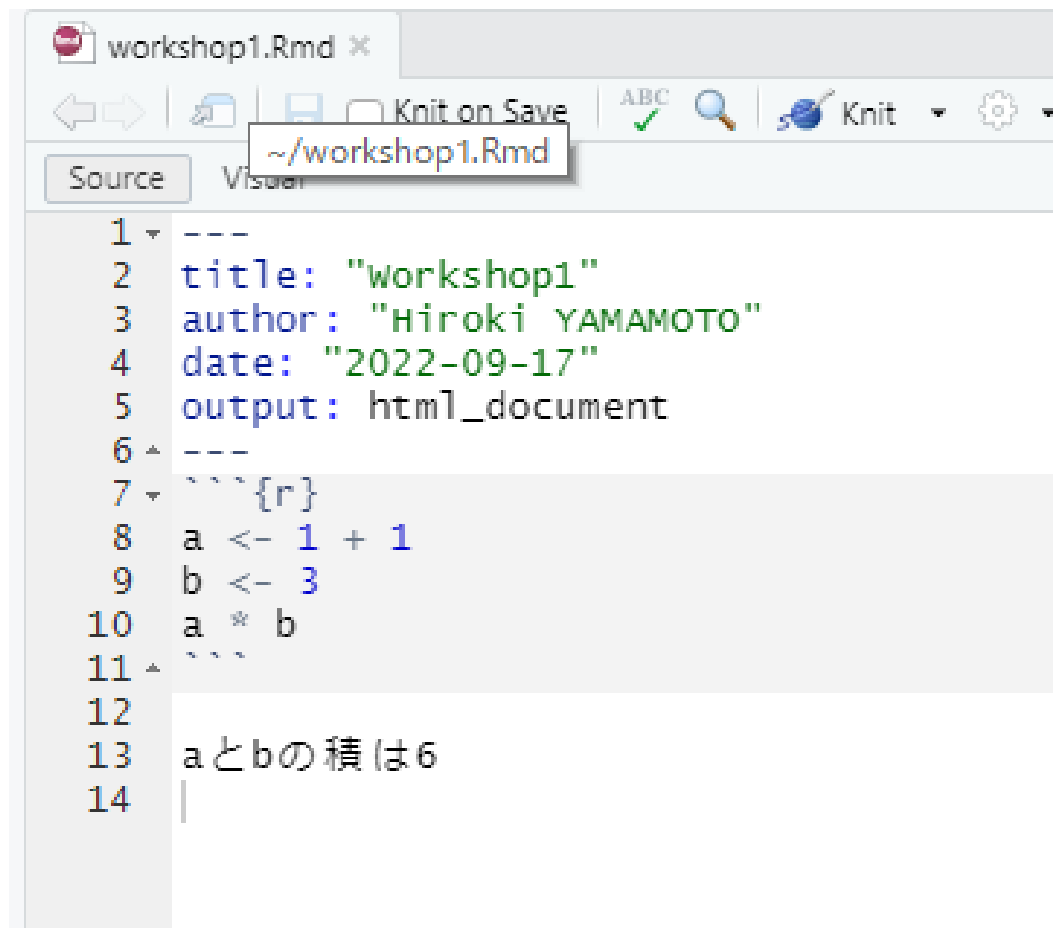
# RmarkdownをKnitしよう

## □“Save File”というウィンドウがでてくる

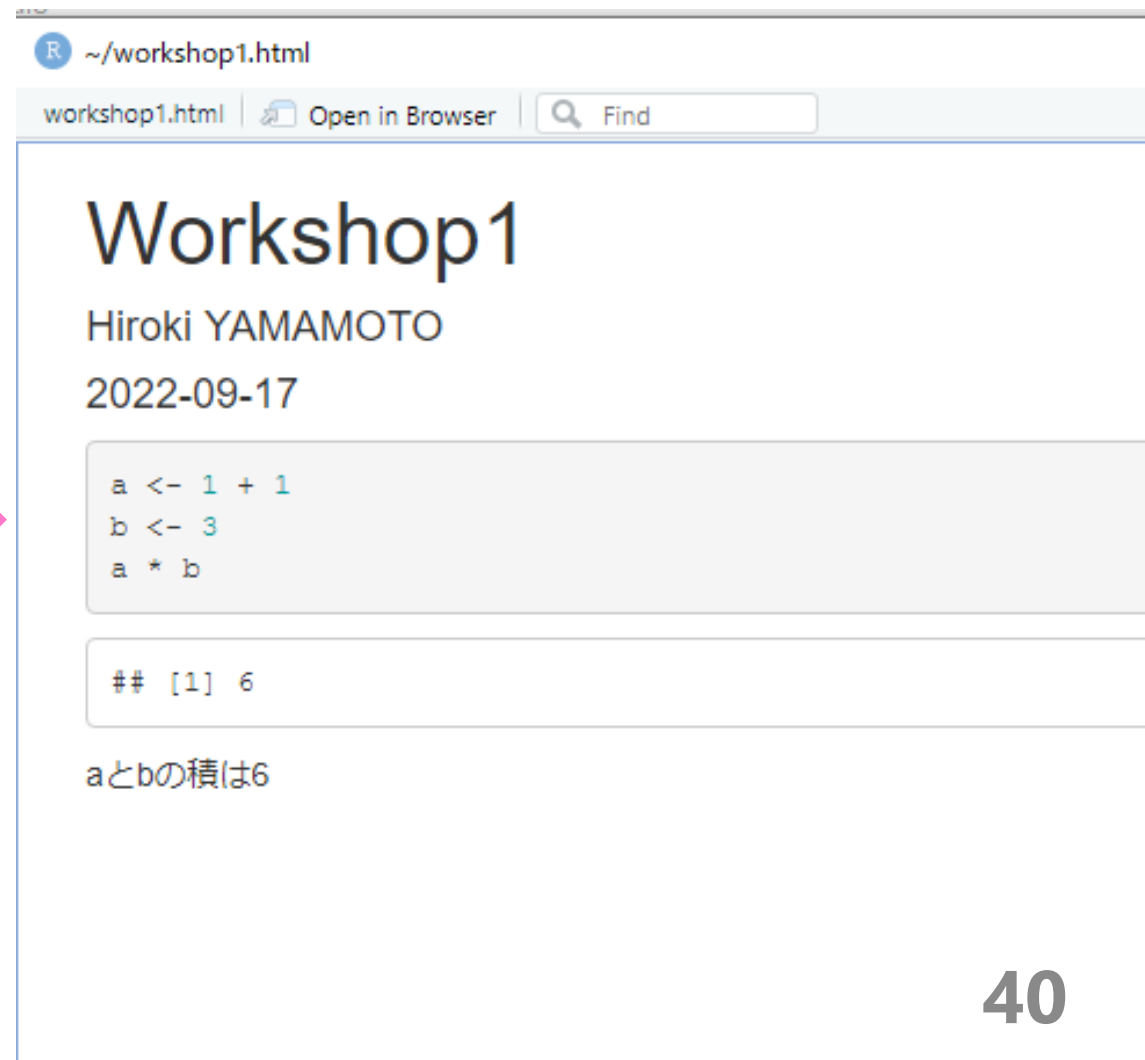
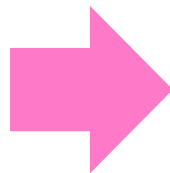
- ・好きな名前をつけて“Save”ボタン
    - －今回は“workshop1”
- という名前をつけましょう。

# RmarkdownをKnitしよう

□htmlファイルが出力され、Viewerに表示される



```
1 ---
2 title: "Workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6 ---
7 ```{r}
8 a <- 1 + 1
9 b <- 3
10 a * b
11 ```
12
13 aとbの積は6
14 |
```



```
~/workshop1.html
workshop1.html Open in Browser Find

Workshop1
Hiroki YAMAMOTO
2022-09-17

a <- 1 + 1
b <- 3
a * b

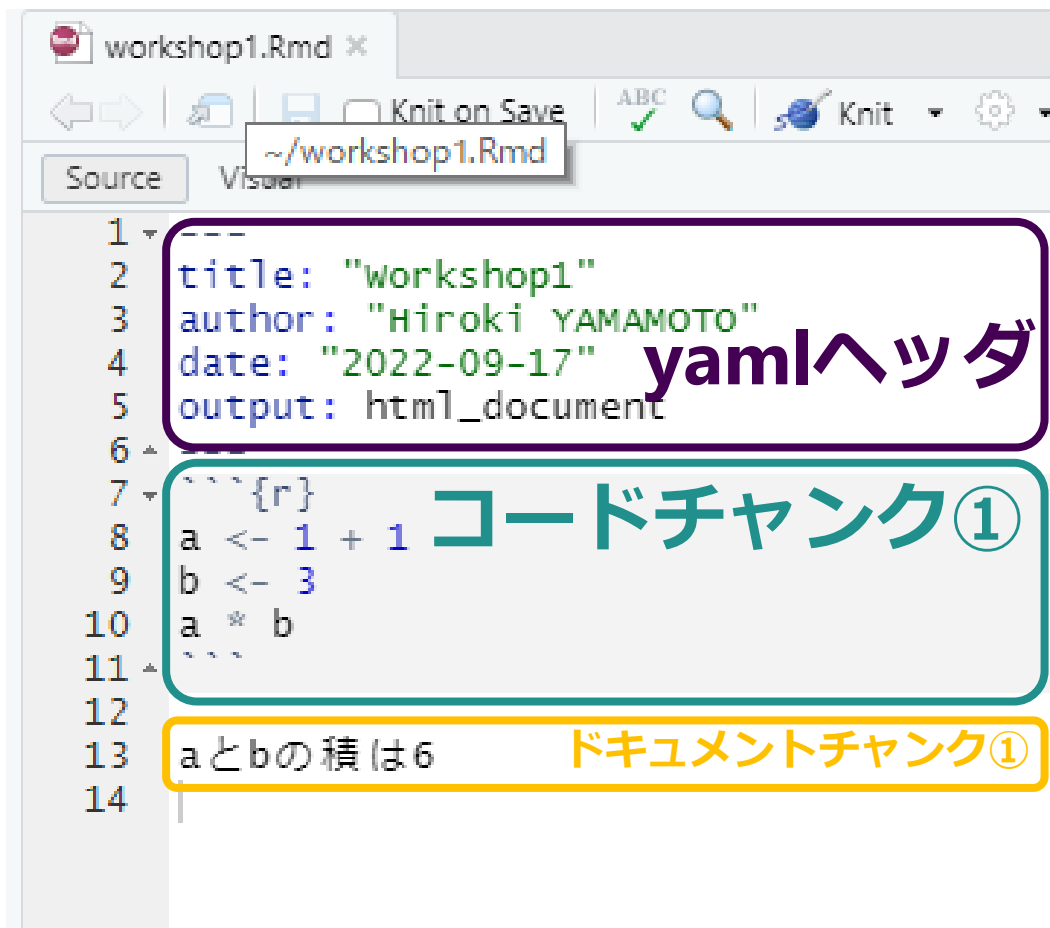
[1] 6

aとbの積は6
```



# RmarkdownをKnitしよう

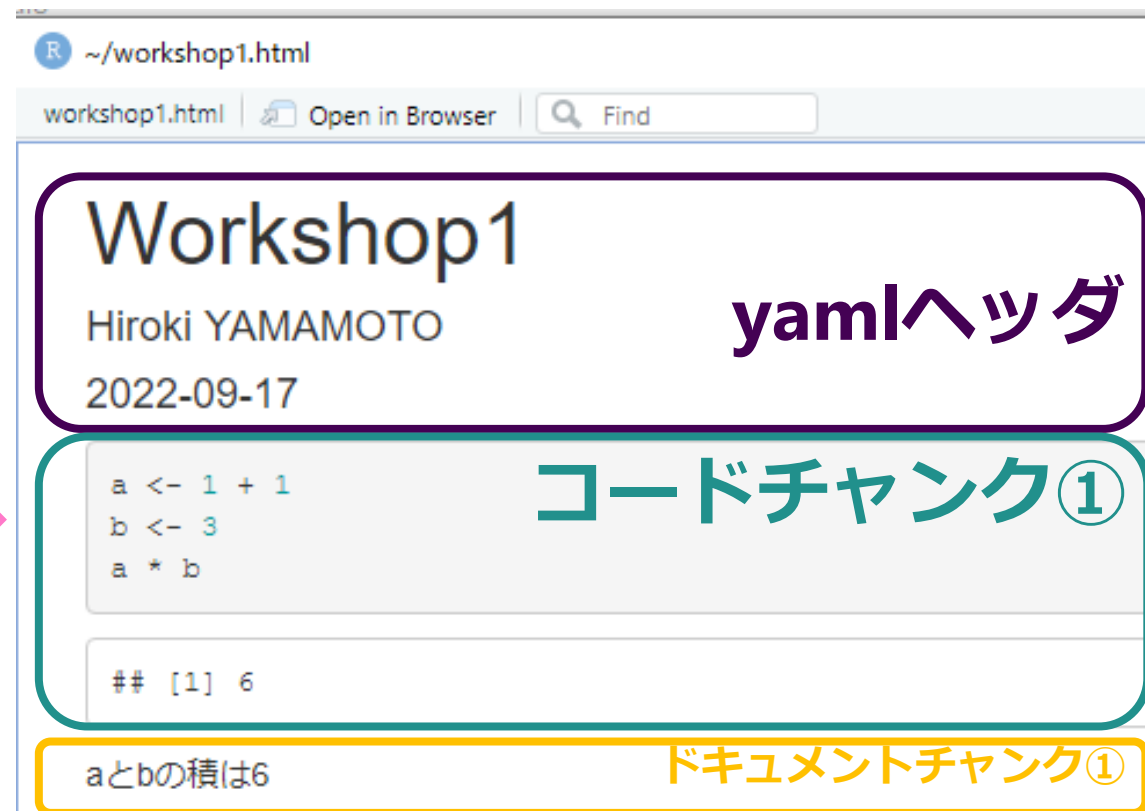
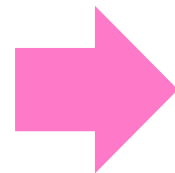
□Rmarkdownに書いたコードとViewerのhtml出力は対応している



```
1 ---
2 title: "Workshop1"
3 author: "Hiroki YAMAMOTO"
4 date: "2022-09-17"
5 output: html_document
6
7 ```{r}
8 a <- 1 + 1
9 b <- 3
10 a * b
11 ```
12
13 aとbの積は6
14
```

Annotations in the image:

- yamlヘッダ** (lines 2-5)
- コードチャンク①** (lines 7-11)
- ドキュメントチャンク①** (line 13)



workshop1.html | Open in Browser | Find

**Workshop1**  
Hiroki YAMAMOTO  
2022-09-17

**コードチャンク①**

```
a <- 1 + 1
b <- 3
a * b
```

```
[1] 6
```

aとbの積は6

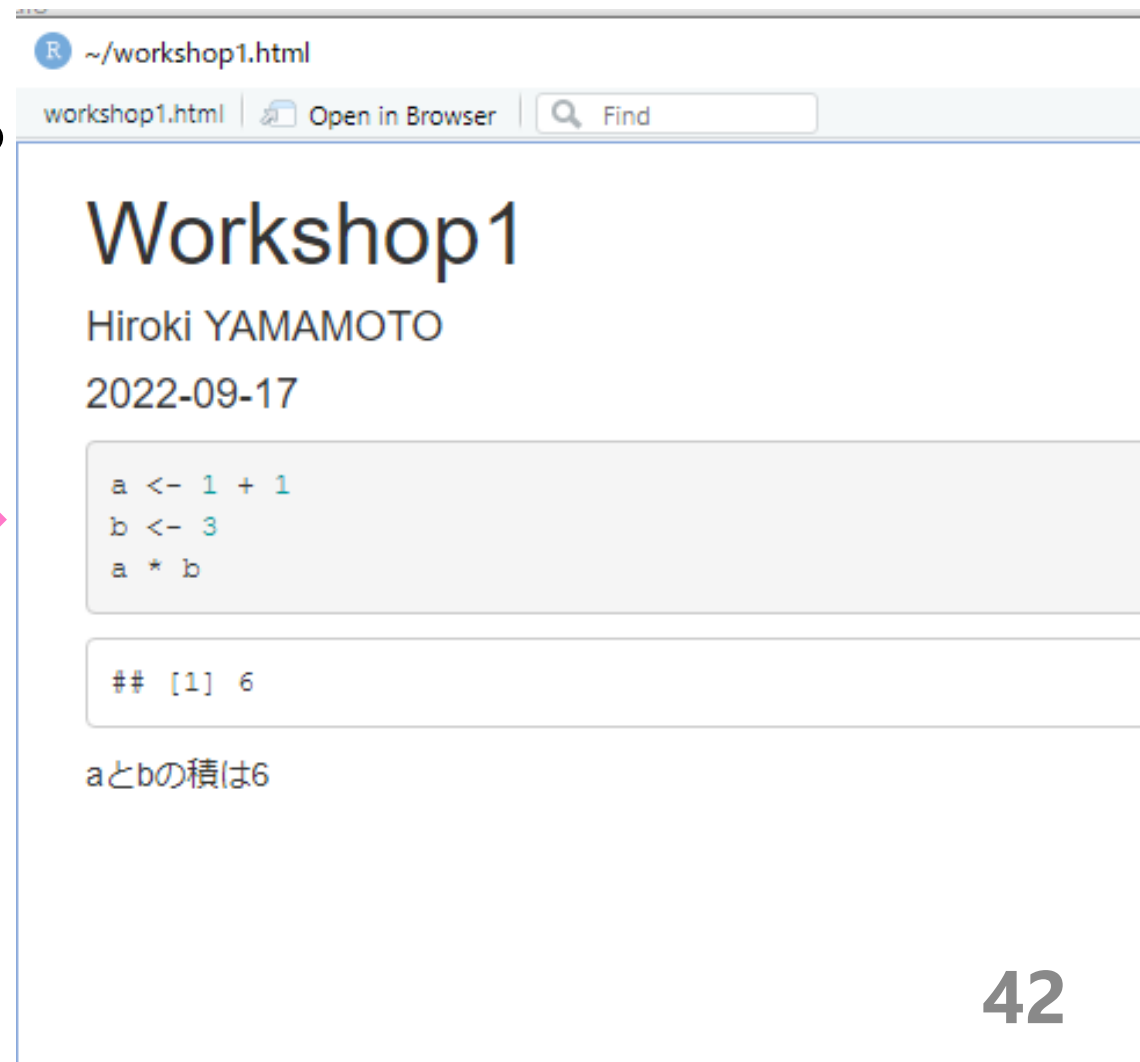
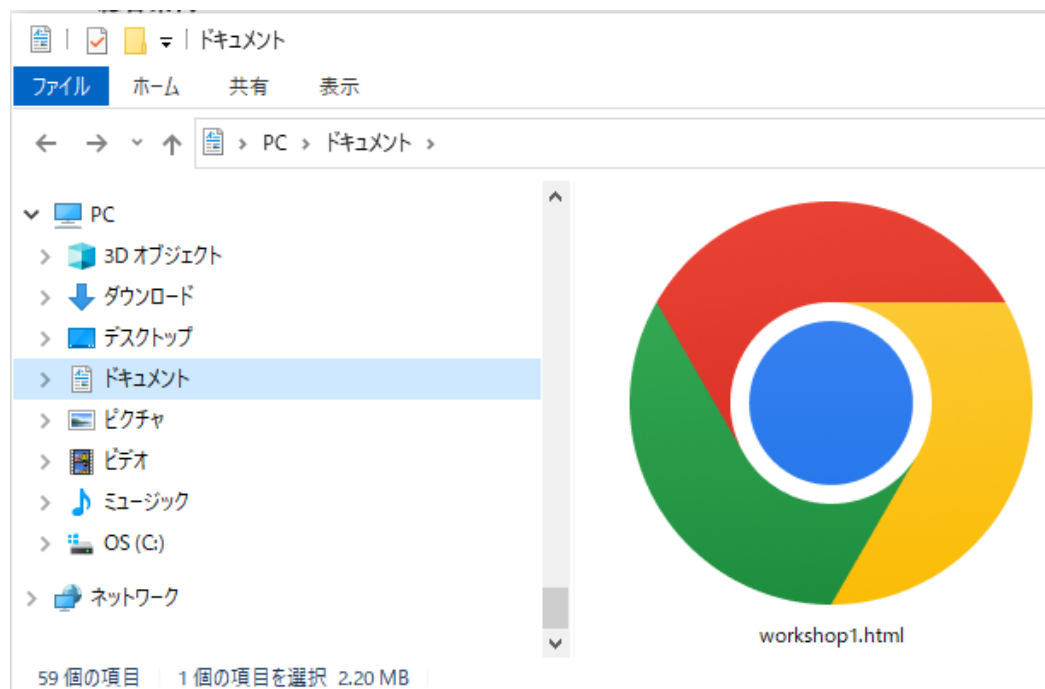
Annotations in the image:

- yamlヘッダ** (title, author, date)
- コードチャンク①** (code and output)
- ドキュメントチャンク①** (text)

# RmarkdownをKnitしよう

## □My documentsにhtmlファイルが出力される

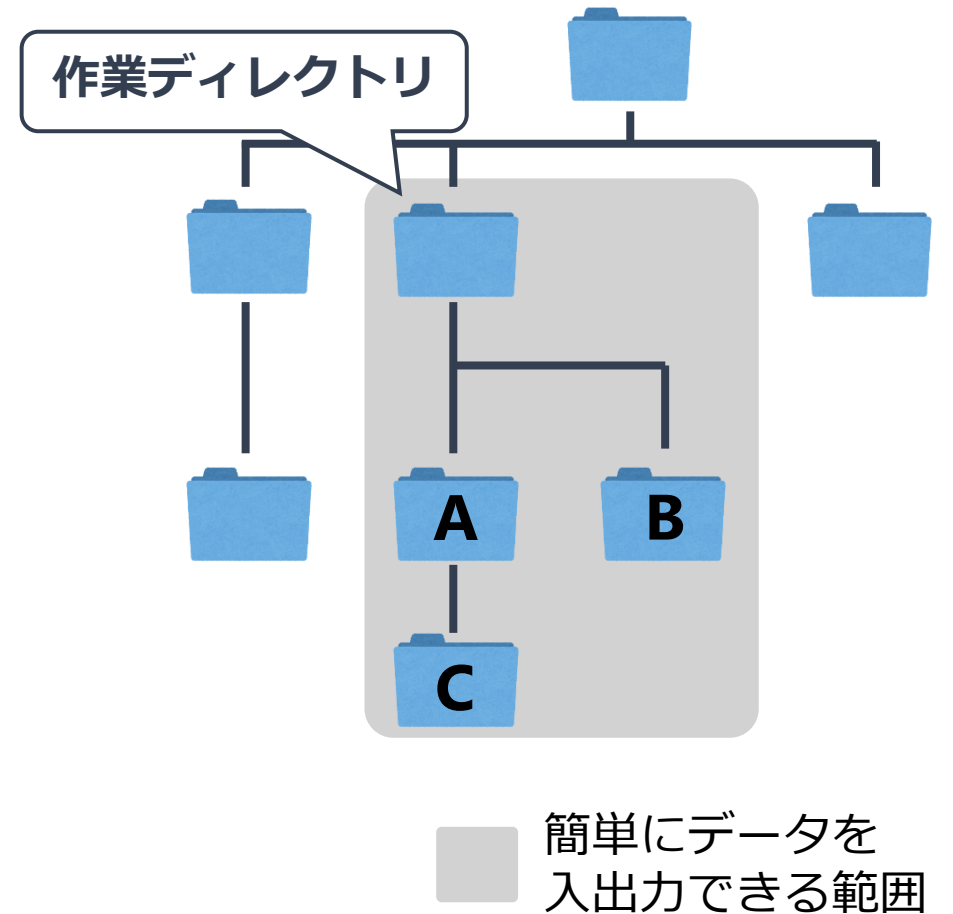
- ・クリックするとWebブラウザに  
さきほどのhtmlファイルが表示される  
— “workshop1.html”



# 作業ディレクトリ

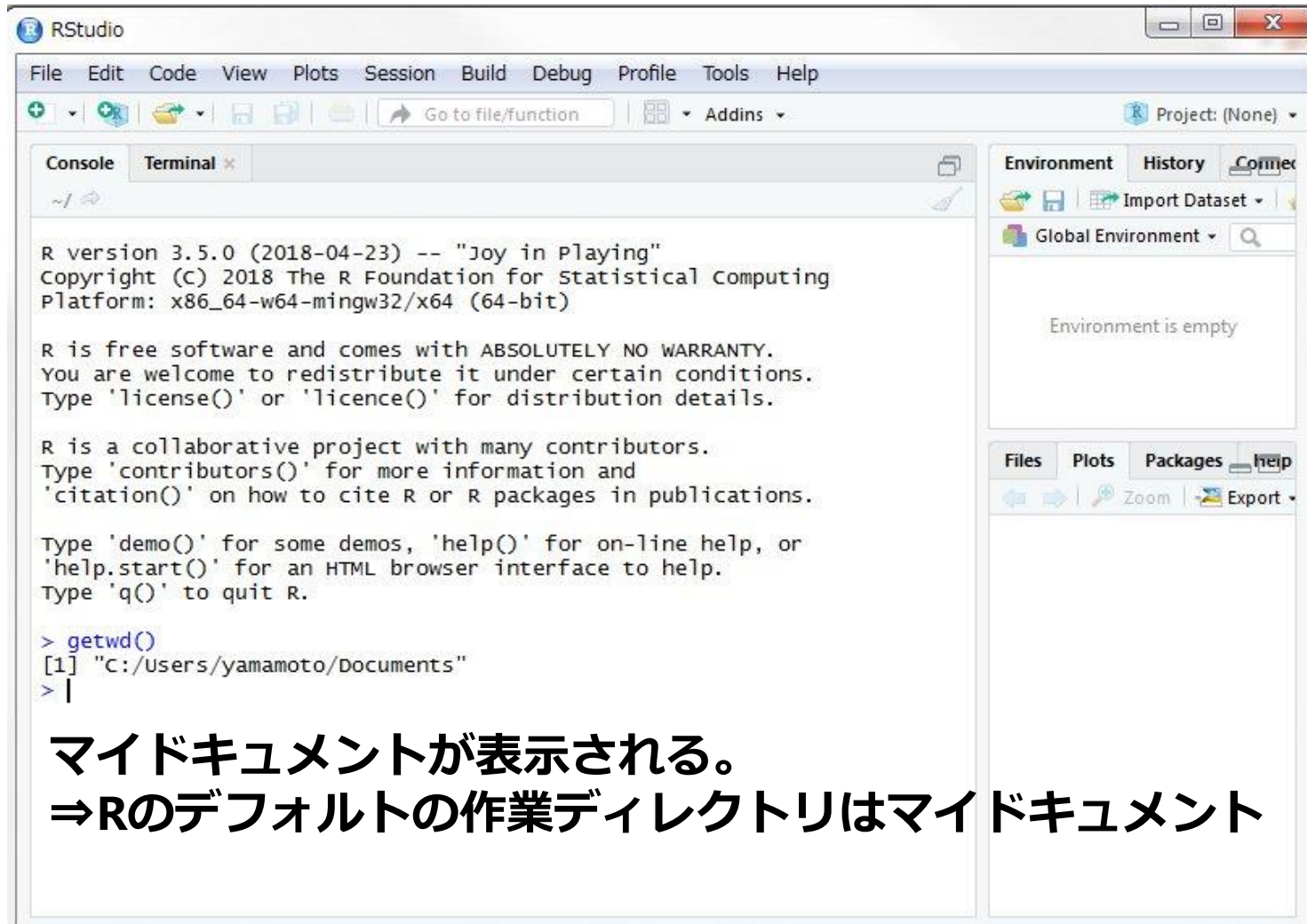
## □作業ディレクトリ

- ・ PCは多くのフォルダが階層構造をつくる
- ・ Rが、ファイルを入出力することができるフォルダを**作業ディレクトリ**という。
  - ー 作業ディレクトリと  
その深層のフォルダにあるデータは簡単にファイルの入出力ができる。
  - ー 作業ディレクトリの外部で  
ファイルを入出力したいときはディレクトリを変更するなど、一手間必要。



# getwd()関数で作業ディレクトリを確認

## □RStudioを起動してコンソールでgetwd()を実行



The screenshot shows the RStudio interface with the console pane active. The console displays the R startup message and the output of the `getwd()` function. The output is `"C:/Users/yamamoto/Documents"`, indicating the current working directory is the user's Documents folder.

```
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (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.

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.

> getwd()
[1] "C:/Users/yamamoto/Documents"
> |
```

マイドキュメントが表示される。

⇒Rのデフォルトの作業ディレクトリはマイドキュメント

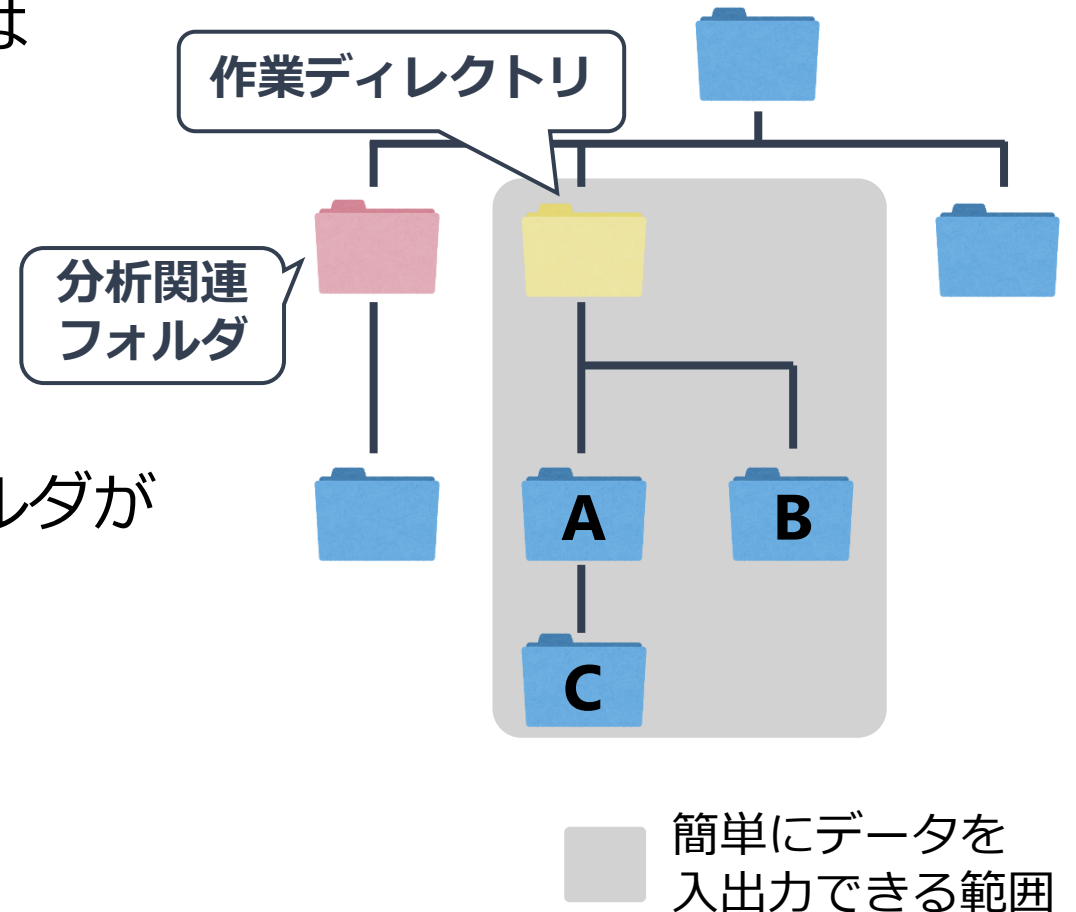
# 作業ディレクトリ

## □Rのデフォルトの作業ディレクトリは「マイドキュメント」

- ・マイドキュメント内のファイルに関しては入出力は簡単。
  - それ以外のフォルダにアクセスしようとする则一手間必要。
- ・分析関連ファイルをまとめておいたフォルダが作業ディレクトリになると便利。



作業ディレクトリを変更しましょう。



# 作業ディレクトリの変更

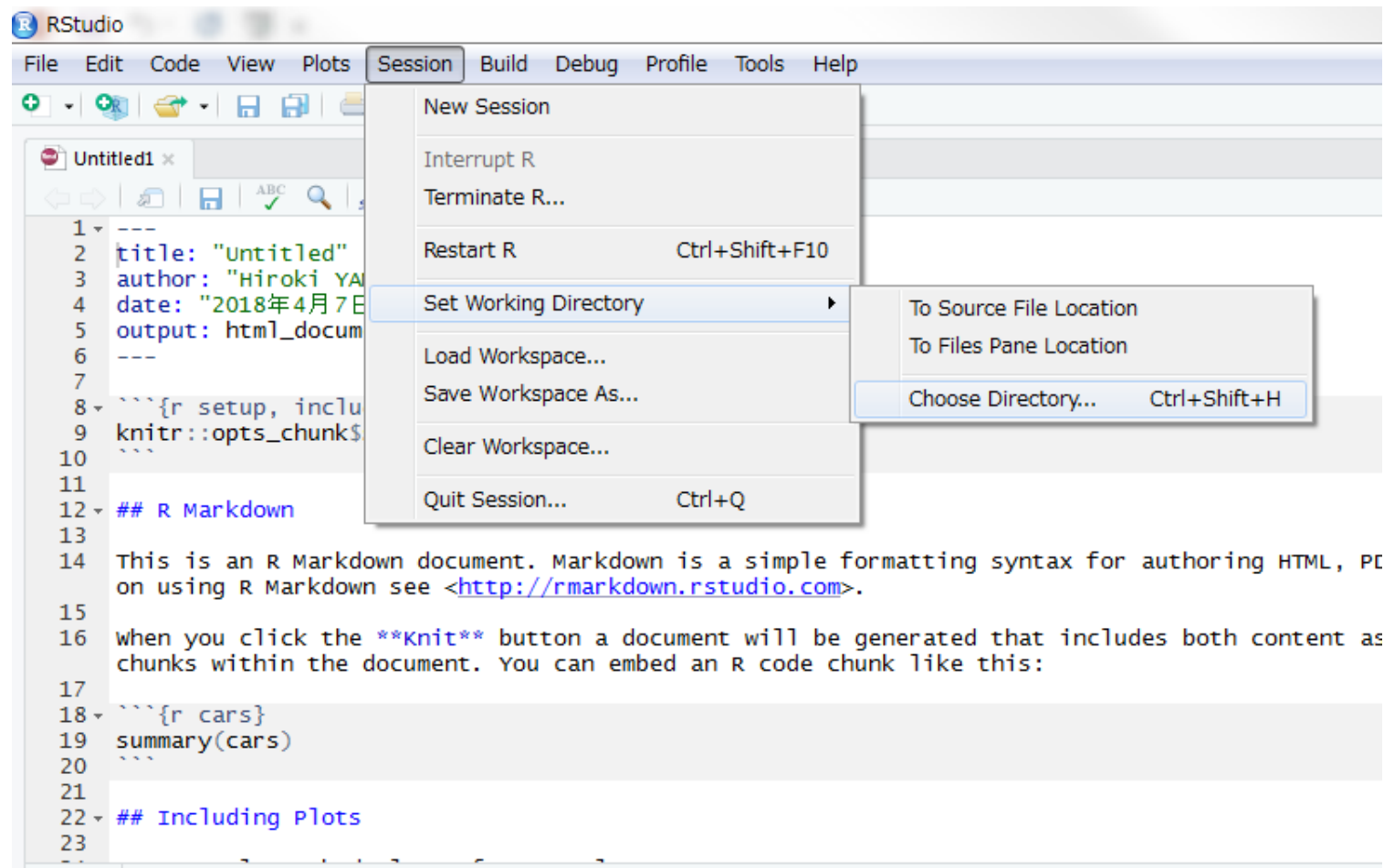
## □ディレクトリを変更する

- “Session”

⇒“Set Working Directory”

⇒“Choose Directory...”

で、ファイルを入出力する  
フォルダを指定



# Rmarkdown+Knitで分析レポート作成

## □分析レポート作成のワークフロー

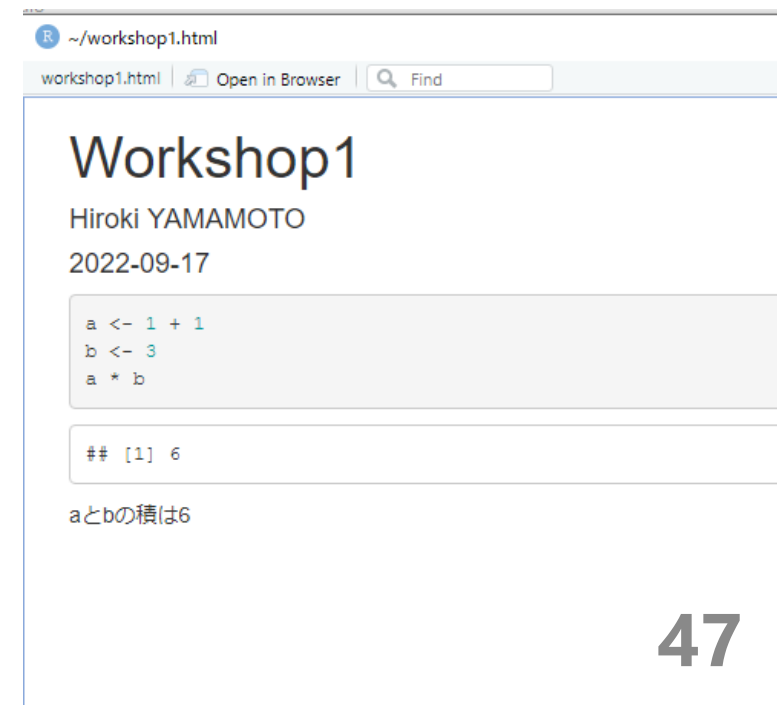
### ・前準備

- (a) エディタにRmarkdown画面を生成
- (b) yamlヘッダを残して下部をDelete

### ・Knit

- (1) Knitボタンを押す
- (2) ファイル名をつけて保存
- (3) htmlファイルが出力される

- ・コードをかく（以下の①～⑤を繰り返す）
  - ①コードチャンクを挿入(Ctrl+Alt+I)
  - ②コードチャンクにコードをかく
  - ③カーソルの行のコードを実行(Ctrl+Enter)
  - ④コンソール・ワークスペースで出力を確認
  - ⑤メモをドキュメントチャンクに記述



# パッケージのインストール



# パッケージをインストールしよう

## □パッケージ

### ●R用の関数のセット

- Rのインストール時から備わっているもの
- ウェブ上からダウンロードして使うもの
- 分析の目的にあわせて、必要なパッケージを用いる

### ●**install.packages()**関数を使用して、パッケージをインストール

- パッケージ名をダブルクォーテーション("")でくくって指定

### ●tidyverseパッケージをインストールしてみよう

- **install.packages("tidyverse")**

# 関数とは？

## □関数 (function)

- 「入力」に対して「出力」を返すもの
  - ・ 入力する値のことを「**引数**（ひきすう）」と呼ぶ
  - ・ 引数が複数あることもあり、  
左から「第1引数・第2引数・第3引数...」と呼ばれる
- Rの関数は全て、関数の名前の後にカッコ（）がついた形式
  - ・ 例：install.packages()関数



# Tidyverseパッケージとは...

## □データ解析のためのRパッケージ群

Components



# パッケージの読み込み

## □library()関数でパッケージを読み込む

- パッケージはRを起動するたびに読み込む必要があるので注意。
  - ・コードの先頭で読み込むことをオススメします。

### ●library(tidyverse) ...tidyverseパッケージを読み込む

```
library(tidyverse)
```

```
-- Attaching packages -----
----- tidyverse 1.2.1 --
```

```
√ ggplot2 2.2.1 √ purrr 0.2.4
√ tibble 1.4.2 √ dplyr 0.7.4
√ tidyr 0.8.0 √ stringr 1.3.0
√ readr 1.1.1 √ forcats 0.3.0
```

 tidyverseパッケージに含まれるパッケージ群

```
-- Conflicts -----
----- tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag() masks stats::lag()
```

# データフレームの型：ワイド型とロング型

ワイド型データフレーム

地点	6時	12時	18時
札幌			
東京			
福岡			



ロング型データフレーム

地点	時刻	天気
札幌	6時	
札幌	12時	
札幌	18時	
東京	6時	
東京	12時	
東京	18時	
福岡	6時	
福岡	12時	
福岡	18時	

# データフレームの型：ワイド型とロング型

ワイド型データフレーム

地点	6時	12時	18時
札幌			
東京			
福岡			

1つの観測

ロング型データフレーム

地点	時刻	天気
札幌	6時	
札幌	12時	
札幌	18時	
東京	6時	
東京	12時	
東京	18時	
福岡	6時	
福岡	12時	
福岡	18時	

1つの観測

# Rではロング型が扱いやすい

□ ggplot2やdplyrの関数はロング型に対応

□ 第 1 引数にデータフレームを入れ, 列名で変数を指定

> ggplot(**iris**, aes(x = Sepal.Length, y = Sepal.Width)) + ...

□ 列名を指定してデータを抽出する

> select(iris, **Sepal.Length**, **Species**)

> filter(iris, **Sepal.Length** > 7)

Rにデフォルトで用意されている  
データフレーム「iris」



> head(iris)

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa



## データフレームを操作・加工する関数が詰まったパッケージ

①行の一部を取り出す

`filter()`関数

②列の一部を取り出す

`select()`関数

③新しい列を追加する

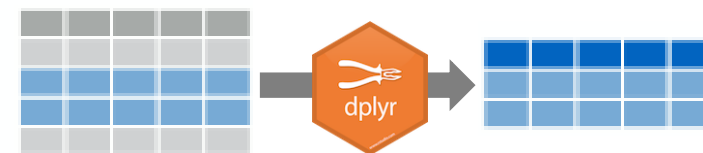
`mutate()`関数

④データをグループ化する

`group_by()`関数

⑤データを要約する

`summarise()`関数



	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

dplyrの関数を使って  
データフレームirisを  
加工してみよう



# ①行の一部を抽出する：filter()関数

## □filter()関数

- 第1引数：データフレームの名前
- 第2引数以降：抽出したい行を表す条件式

データフレームirisからsetosaのアヤメのデータだけ抽出しよう

```
```{r}  
filter(iris, Species == "setosa")  
```
```

※文字列の指定には "" ダブルクォーテーションが必要

# Rの比較演算子

## □条件式を書く際に使用する記号

| 記号   | 意味          |
|------|-------------|
| ==   | 等しい ※「=」は代入 |
| !=   | 等しくない       |
| >=   | 以上          |
| >    | より大きい       |
| <=   | 以下          |
| <    | より小さい       |
| %in% | グループメンバシップ  |

| 記号     | 意味         |
|--------|------------|
| is.na  | NA(欠損値)である |
| !is.na | NA(欠損値)でない |
| &      | かつ         |
|        | または        |

# ①行の一部を抽出する： filter()関数

irisからSepal.Lengthが7より大きいデータだけ抽出しよう

```
filter(iris, Sepal.Length > 7) # Sepal.Lengthが7より大きい行を抽出
```

| ## |   | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species   |
|----|---|--------------|-------------|--------------|-------------|-----------|
| ## | 1 | 7.1          | 3.0         | 5.9          | 2.1         | virginica |
| ## | 2 | 7.6          | 3.0         | 6.6          | 2.1         | virginica |
| ## | 3 | 7.3          | 2.9         | 6.3          | 1.8         | virginica |
| ## | 4 | 7.2          | 3.6         | 6.1          | 2.5         | virginica |
| ## | 5 | 7.7          | 3.8         | 6.7          | 2.2         | virginica |
| ## | 6 | 7.7          | 2.6         | 6.9          | 2.3         | virginica |
| ## | 7 | 7.7          | 2.8         | 6.7          | 2.0         | virginica |

# ①行の一部を抽出する： filter()関数

filter()関数で複数の条件を指定しよう  
「Sepal.Lengthが6.5以上 かつ Speciesがversicolor」の行を抽出

※条件式を & または コンマ で区切る ⇒ 「かつ」を表す



```
Sepal.Lengthが6.5以上かつSpecies "versicolor"である行を抽出
filter(iris, Sepal.Length >= 6.5, Species == "versicolor")
```

| ## |   | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species    |
|----|---|--------------|-------------|--------------|-------------|------------|
| ## | 1 | 7.0          | 3.2         | 4.7          | 1.4         | versicolor |
| ## | 2 | 6.9          | 3.1         | 4.9          | 1.5         | versicolor |
| ## | 3 | 6.5          | 2.8         | 4.6          | 1.5         | versicolor |
| ## | 4 | 6.6          | 2.9         | 4.6          | 1.3         | versicolor |
| ## | 5 | 6.7          | 3.1         | 4.4          | 1.4         | versicolor |
| ## | 6 | 6.6          | 3.0         | 4.4          | 1.4         | versicolor |

# ①行の一部を抽出する： filter()関数

filter()関数で複数の条件を指定しよう  
「Sepal.Lengthが7以上 または Sepal.Lengthが4以上」の行を抽出

※条件式を | (縦線) で区切る ⇒ 「または」を表す



```
Sepal.Lengthが7以上 またはSepal.Lengthが4以上である行を抽出
filter(iris, Sepal.Length >= 7 | Sepal.Width >= 4)
```

| ## |   | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species    |
|----|---|--------------|-------------|--------------|-------------|------------|
| ## | 1 | 5.8          | 4.0         | 1.2          | 0.2         | setosa     |
| ## | 2 | 5.7          | 4.4         | 1.5          | 0.4         | setosa     |
| ## | 3 | 5.2          | 4.1         | 1.5          | 0.1         | setosa     |
| ## | 4 | 5.5          | 4.2         | 1.4          | 0.2         | setosa     |
| ## | 5 | 7.0          | 3.2         | 4.7          | 1.4         | versicolor |

## ②列の一部を抽出する：select()関数

### □select()関数

- 第1引数：データフレームの名前
- 第2引数以降：抽出したい列名

irisからSpeciesという名前の列だけ抽出しよう

```
select(iris, Species)
```

irisからSpecies, Sepal.Length, Sepal.Widthの列を抽出しよう

```
select(iris, Species, Sepal.Length, Sepal.Width)
```

## ②列の一部を抽出する：select()関数

指定した列を除外しよう

```
select(iris, -Species)
```

|   | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width |
|---|--------------|-------------|--------------|-------------|
| 1 | 5.1          | 3.5         | 1.4          | 0.2         |
| 2 | 4.9          | 3.0         | 1.4          | 0.2         |
| 3 | 4.7          | 3.2         | 1.3          | 0.2         |
| 4 | 4.6          | 3.1         | 1.5          | 0.2         |
| 5 | 5.0          | 3.6         | 1.4          | 0.2         |
| 6 | 5.4          | 3.9         | 1.7          | 0.4         |

Sで始まる列名だけ抽出しよう

```
select(iris, starts_with("S"))
```

|   | Sepal.Length | Sepal.Width | Species |
|---|--------------|-------------|---------|
| 1 | 5.1          | 3.5         | setosa  |
| 2 | 4.9          | 3.0         | setosa  |
| 3 | 4.7          | 3.2         | setosa  |
| 4 | 4.6          | 3.1         | setosa  |
| 5 | 5.0          | 3.6         | setosa  |
| 6 | 5.4          | 3.9         | setosa  |

# ③新しい列を追加する：mutate()関数

## □mutate()関数

- 第1引数：データフレーム

- 第2引数以降：新しい列の名前と追加する内容を指定

各アヤメのがく・花びらの面積を計算し、新規な列に追加しよう

```
mutate(iris,
 Area_Sepal = Sepal.Length * Sepal.Width,
 Area_Petal = Petal.Length * Petal.Width)
```

新規な列名  
を指定

それぞれの列に、  
各行の計算結果を代入

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species | Area_Sepal | Area_Petal |      |
|--------------|-------------|--------------|-------------|---------|------------|------------|------|
| 5.1          | ×           | 3.5          | ×           | 0.2     | setosa     | 17.85      | 0.28 |
| 4.9          |             | 3.0          |             | 0.2     | setosa     | 14.70      | 0.28 |
| 4.7          |             | 3.2          |             | 0.2     | setosa     | 15.04      | 0.26 |
| 4.6          |             | 3.1          |             | 0.2     | setosa     | 14.26      | 0.30 |
| 5.0          |             | 3.6          |             | 0.2     | setosa     | 18.00      | 0.28 |
| 5.4          |             | 3.9          |             | 0.4     | setosa     | 21.06      | 0.68 |



### ③条件ごとに異なる値を追加する

#### □mutate()関数 + if\_else()関数

□if\_else(条件式, TRUEのとき, FALSEのとき)

□Speciesがvirginicaならば"virginica", それ以外は"NOT virginica"を入れる列Species2を追加しよう

```
mutate(iris,
 Species2 = if_else(Species == "virginica", "virginica", "NOT virginica"))
```

| ## |   | Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species | Species2      |
|----|---|--------------|-------------|--------------|-------------|---------|---------------|
| ## | 1 | 5.1          | 3.5         | 1.4          | 0.2         | setosa  | NOT virginica |
| ## | 2 | 4.9          | 3.0         | 1.4          | 0.2         | setosa  | NOT virginica |
| ## | 3 | 4.7          | 3.2         | 1.3          | 0.2         | setosa  | NOT virginica |
| ## | 4 | 4.6          | 3.1         | 1.5          | 0.2         | setosa  | NOT virginica |
| ## | 5 | 5.0          | 3.6         | 1.4          | 0.2         | setosa  | NOT virginica |
| ## | 6 | 5.4          | 3.9         | 1.7          | 0.4         | setosa  | NOT virginica |
| ## | 7 | 4.5          | 2.4         | 1.4          | 0.2         | setosa  | NOT virginica |

## ④データを一行に要約： summarise()関数

### □summarise()関数

- ・ 第1引数：データフレーム

第2引数以降：要約して出力するデータフレームの列の情報

irisのSepal.Lengthの平均と不偏標準偏差を計算し、  
その結果を新たなデータフレームに要約しよう

```
summarise(iris,
 Mean = mean(Sepal.Length),
 SD = sd(Sepal.Length))
```

新規な列名 = 関数名(計算したい元の列名)

```
Mean SD
1 5.843333 0.8280661
```

# ⑤データをグループ化する：group\_by()関数

## □group\_by()関数

- ・ 第1引数：データフレーム
- 第2引数以降：グループ化する変数名

irisのデータをSpeciesごとにグループ化しよう

```
group_by(iris, Species)
```

- Speciesでグループ化したirisを、**iris\_g**というオブジェクトに代入
- iris\_gの頭6行だけを表示

```
iris_g <- group_by(iris, Species)
head(iris_g)
```

```
A tibble: 6 × 5
Groups: Species [1]
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
<dbl> <dbl> <dbl> <dbl> <fct>
1 5.1 3.5 1.4 0.2 setosa
2 4.9 3 1.4 0.2 setosa
3 4.7 3.2 1.3 0.2 setosa
4 4.6 3.1 1.5 0.2 setosa
5 5 3.6 1.4 0.2 setosa
6 5.4 3.9 1.7 0.4 setosa
```

変数Speciesでグループ化  
されていることを表す

# group\_by()関数+summarise()関数

グループごとにデータを要約しよう

- group\_by()関数でグループ化したデータフレームiris\_gを, summarise()関数の第1引数に渡す

```
iris_g <- group_by(iris, Species)

summarise(iris_g,
 Mean = mean(Sepal.Length),
 SD = sd(Sepal.Length))
```

SpeciesごとにMeanとSD  
が計算されている

```
A tibble: 3 × 3
 Species Mean SD
 <fct> <dbl> <dbl>
1 setosa 5.01 0.352
2 versicolor 5.94 0.516
3 virginica 6.59 0.636
```

# データフレームToothGrowth

モルモットにビタミンCかジュースを投与し，歯の長さ(**len**)を記録

□投与物(**supp**)：ビタミンC (VC) / オレンジジュース (OJ) の2水準

□投与量(**dose**)：0.5 / 1.0 / 2.0 mgの3水準

> head(ToothGrowth)

|                                                                                     |   | len  | supp | dose |
|-------------------------------------------------------------------------------------|---|------|------|------|
|    | 1 | 4.2  | VC   | 0.5  |
|   | 2 | 11.5 | VC   | 0.5  |
|  | 3 | 7.3  | VC   | 0.5  |
|  | 4 | 5.8  | VC   | 0.5  |
|  | 5 | 6.4  | VC   | 0.5  |
|  | 6 | 10.0 | VC   | 0.5  |



## ToothGrowthをdoseとsuppの2変数でグループ化しよう

### □group\_by()関数

- 第1引数：データフレーム
- 第2引数以降：グループ化する変数名

```
TG_g <- group_by(ToothGrowth, dose, supp)
head(TG_g)
```

一旦別のデータフレームに格納  
頭6行だけを表示

```
A tibble: 6 × 3
Groups: dose, supp [1]
 len supp dose
<dbl> <fct> <dbl>
1 4.2 VC 0.5
2 11.5 VC 0.5
3 7.3 VC 0.5
4 5.8 VC 0.5
5 6.4 VC 0.5
6 10 VC 0.5
```

← 変数doseと変数suppの2変数で  
グループ化された

# group\_by()関数+summarise()関数

□2変数でグループ化したデータフレームTG\_gをsummarise()関数に渡し、条件ごとに要約統計量を計算する

```
TG_g <- group_by(ToothGrowth, dose, supp)

summarise(TG_g,
 Mean = mean(len),
 Median = median(len),
 Variance = var(len),
 SD = sd(len))
```

```
A tibble: 6 × 6
Groups: dose [3]
 dose supp Mean Median Variance SD
 <dbl> <fct> <dbl> <dbl> <dbl> <dbl>
1 0.5 OJ 13.2 12.2 19.9 4.46
2 0.5 VC 7.98 7.15 7.54 2.75
3 1 OJ 22.7 23.5 15.3 3.91
4 1 VC 16.8 16.5 6.33 2.52
5 2 OJ 26.1 26.0 7.05 2.66
6 2 VC 26.1 26.0 23.0 4.80
```

dose (3水準) × supp (2水準) ごとに  
平均値, 中央値, 不偏分散, 不偏標準偏差  
が計算されている

# group\_by()関数+mutate()関数

□グループごとに新しい列の値を追加する

□group\_by()関数で出力したデータフレームをmutate()関数の第1引数へ

```
iris_g <- group_by(iris, Species)
mutate(iris_g,
 Mean = mean(Sepal.Length)) # 平均値
```

```
A tibble: 150 x 6
Groups: Species [3]
Sepal.Length Sepal.Width Petal.Length Petal.Width Species Mean
<dbl> <dbl> <dbl> <dbl> <fct> <dbl>
1 5.10 3.50 1.40 0.200 setosa 5.01
2 4.90 3.00 1.40 0.200 setosa 5.01
3 4.70 3.20 1.30 0.200 setosa 5.01
4 4.60 3.10 1.50 0.200 setosa 5.01
5 5.00 3.60 1.40 0.200 setosa 5.01
6 5.40 3.90 1.70 0.400 setosa 5.01
```



# ungroup()関数：グループ化を解除

## □ungroup()関数

- ・ 第1引数：グループ化されたデータフレーム

```
iris_g <- group_by(iris, Species)
ungroup(iris_g)
```

グループ化されていない

```
A tibble: 150 x 5
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
<dbl> <dbl> <dbl> <dbl> <fct>
1 5.10 3.50 1.40 0.200 setosa
2 4.90 3.00 1.40 0.200 setosa
3 4.70 3.20 1.30 0.200 setosa
4 4.60 3.10 1.50 0.200 setosa
5 5.00 3.60 1.40 0.200 setosa
6 5.40 3.90 1.70 0.400 setosa
```

# パイプ演算子を使ったパイプ処理

**パイプ演算子** (Win: Ctrl+Shift+**M**, Mac: ⌘+Shift+**M**)

- tidyverse内のmagrittrパッケージで扱える
- 前の値を, 後ろの関数の第一引数として渡す



%>%

- 上下のコマンドはそれぞれ同じ意味になる

```
head(iris)
```

```
iris %>% head()
```

```
filter(iris, Species == "setosa")
```

```
iris %>% filter(Species == "setosa")
```

# パイプ演算子を使ったパイプ処理

## パイプ演算子

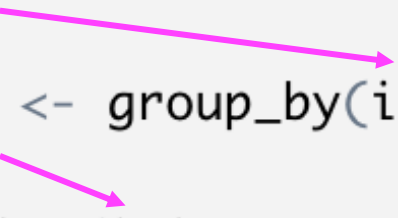
- tidyverseの関数は第1引数がデータフレーム
- 関数間のデータフレームの受け渡しがスムーズ
- いちいちデータフレームを作る必要がない
- 複数の処理をパイプライン化してひと繋がりにできる



## 以下のコマンドをパイプ演算子を使って書いてみよう

- ① irisのうちSepal.Lengthが5より大きい行を抽出する
- ② Speciesごとにグループ化する
- ③ グループごとにSepal.LengthのMeanとSDを計算する

```
iris_2 <- filter(iris, Sepal.Length > 5)
iris_g <- group_by(iris_2, Species)
summarise(iris_g,
 Mean = mean(Sepal.Length),
 SD = sd(Sepal.Length))
```



```
iris %>% |
 filter(Sepal.Length > 5) %>%
 group_by(Species) %>%
 summarise(Mean = mean(Sepal.Length),
 SD = sd(Sepal.Length))
```

# dplyrでデータフレームを操作



## 紹介したdplyrの関数

|            |                             |
|------------|-----------------------------|
| □行の一部を取り出す | <code>filter()</code> 関数    |
| □列の一部を取り出す | <code>select()</code> 関数    |
| □新しい列を追加する | <code>mutate()</code> 関数    |
| □データをグループ化 | <code>group_by()</code> 関数  |
| □データを要約する  | <code>summarise()</code> 関数 |

## その他Tips

|           |                                     |
|-----------|-------------------------------------|
| • 行を並び替える | <code>arrange()</code> 関数           |
| • 列名を変える  | <code>rename()</code> 関数            |
| • 欠損値を除外  | <code>filter(df, !is.na(列名))</code> |
| • パイプ演算子  | <code>%&gt;%</code>                 |

## 関数の合わせ技

- `group_by() + summarise()`
- `group_by() + mutate()`
- `mutate() + if_else()`



R Markdownをknitして  
htmlを出力しましょう