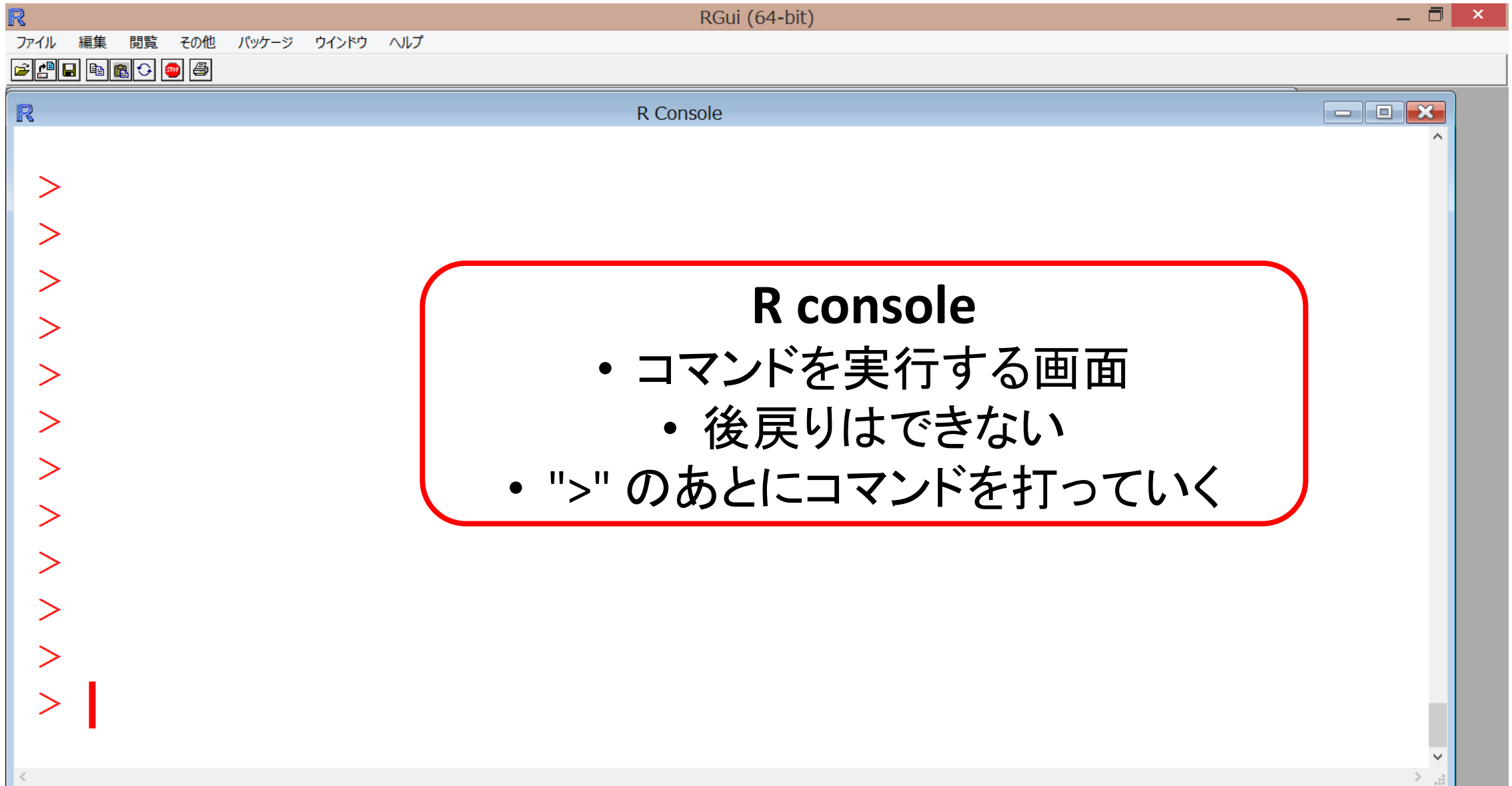
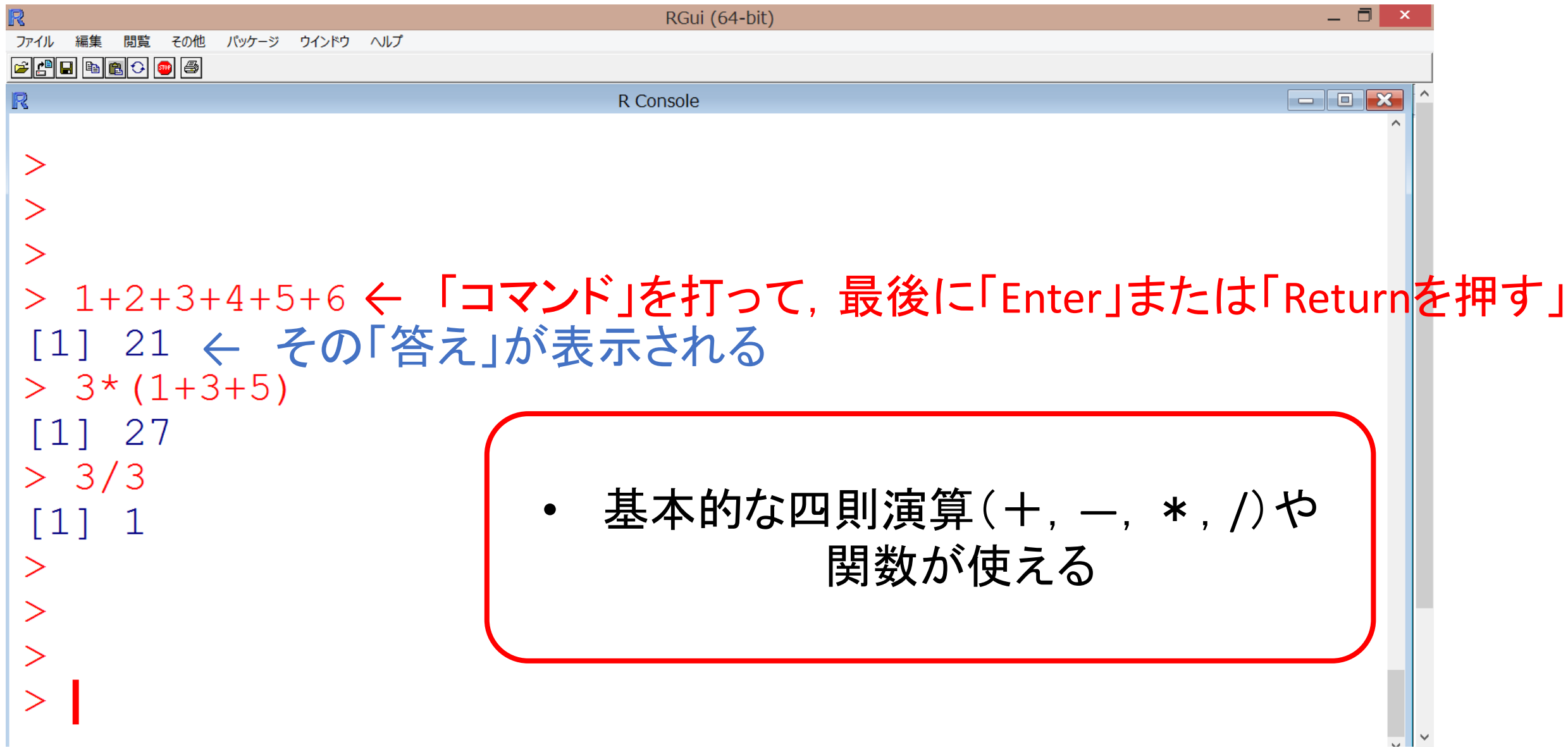


使い方の簡単な練習

- R console上でコマンドを打っていく



- 計算機がわりに使う (1)



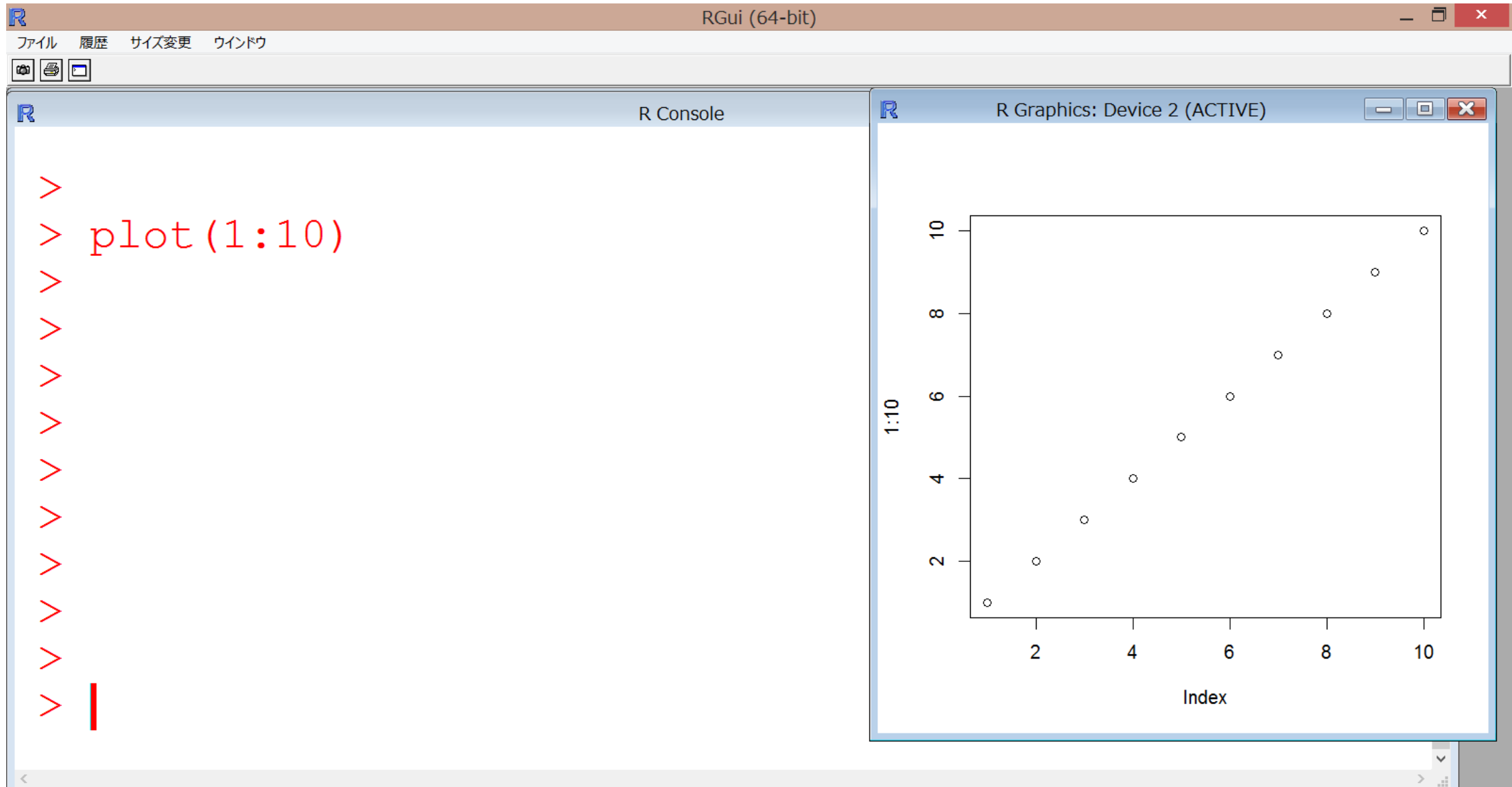
```
RGui (64-bit)
ファイル 編集 閲覧 その他 パッケージ ウィンドウ ヘルプ
R Console
>
>
>
> 1+2+3+4+5+6 ← 「コマンド」を打って、最後に「Enter」または「Returnを押す」
[1] 21 ← その「答え」が表示される
> 3*(1+3+5)
[1] 27
> 3/3
[1] 1
>
>
>
> |
```

- 基本的な四則演算 (+, -, *, /) や関数が見える

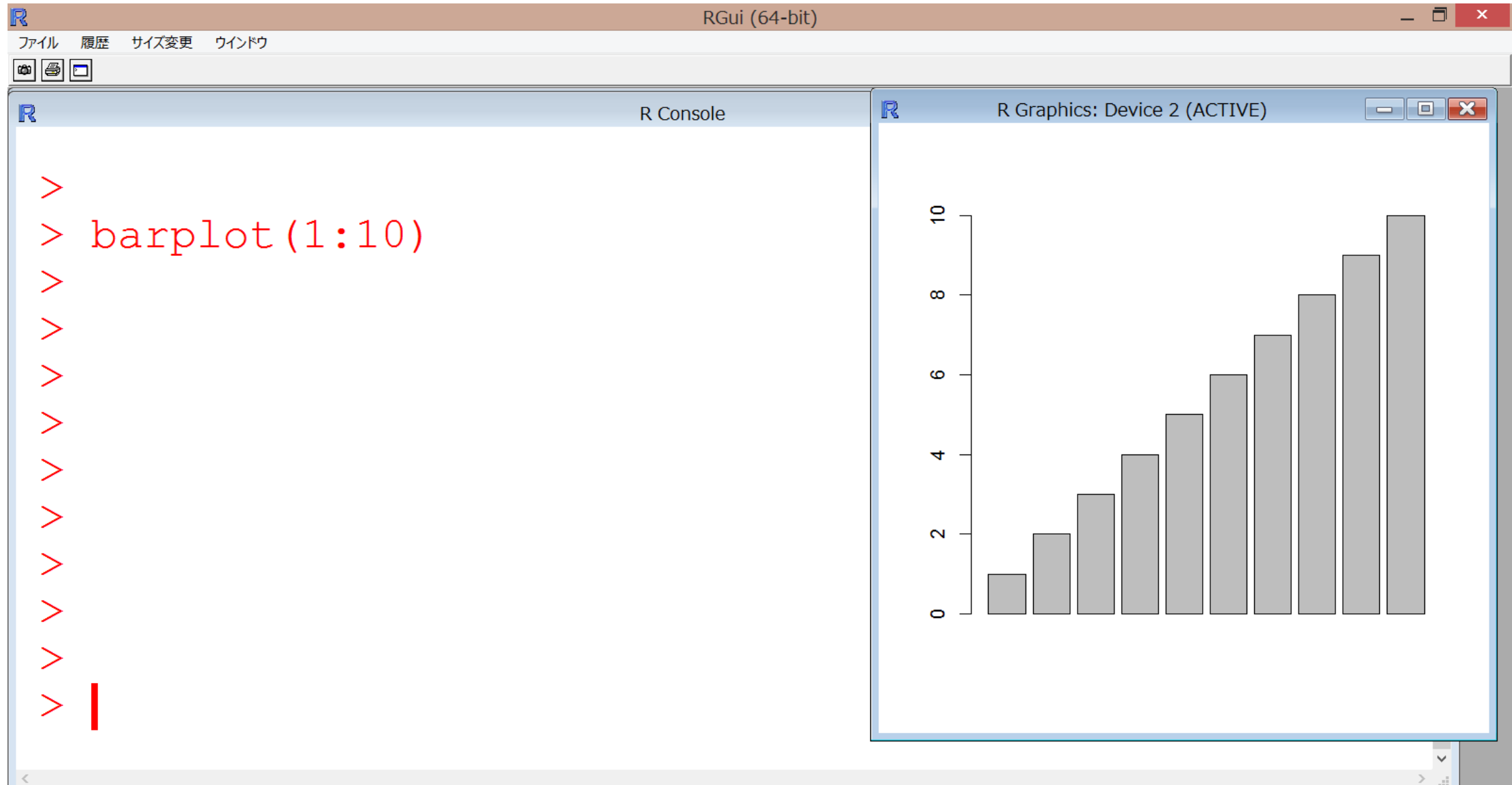
- 計算機がわりに使う (2)

```
RGui (64-bit)
ファイル 編集 閲覧 その他 パッケージ ウィンドウ ヘルプ
R Console
> 1+2*3-4/5 ← 四則演算
[1] 6.2
> 1:5 ← ":" で整数をつなげる. 1:5は, 1から5までの数列を表す
[1] 1 2 3 4 5
> sum(1:5) ← 1から5までの数列の足し算
[1] 15
> mean(1:5) ← 1から5までの数列の平均
[1] 3
>
>
>
>
> |
```

- グラフを書いてみる (1)



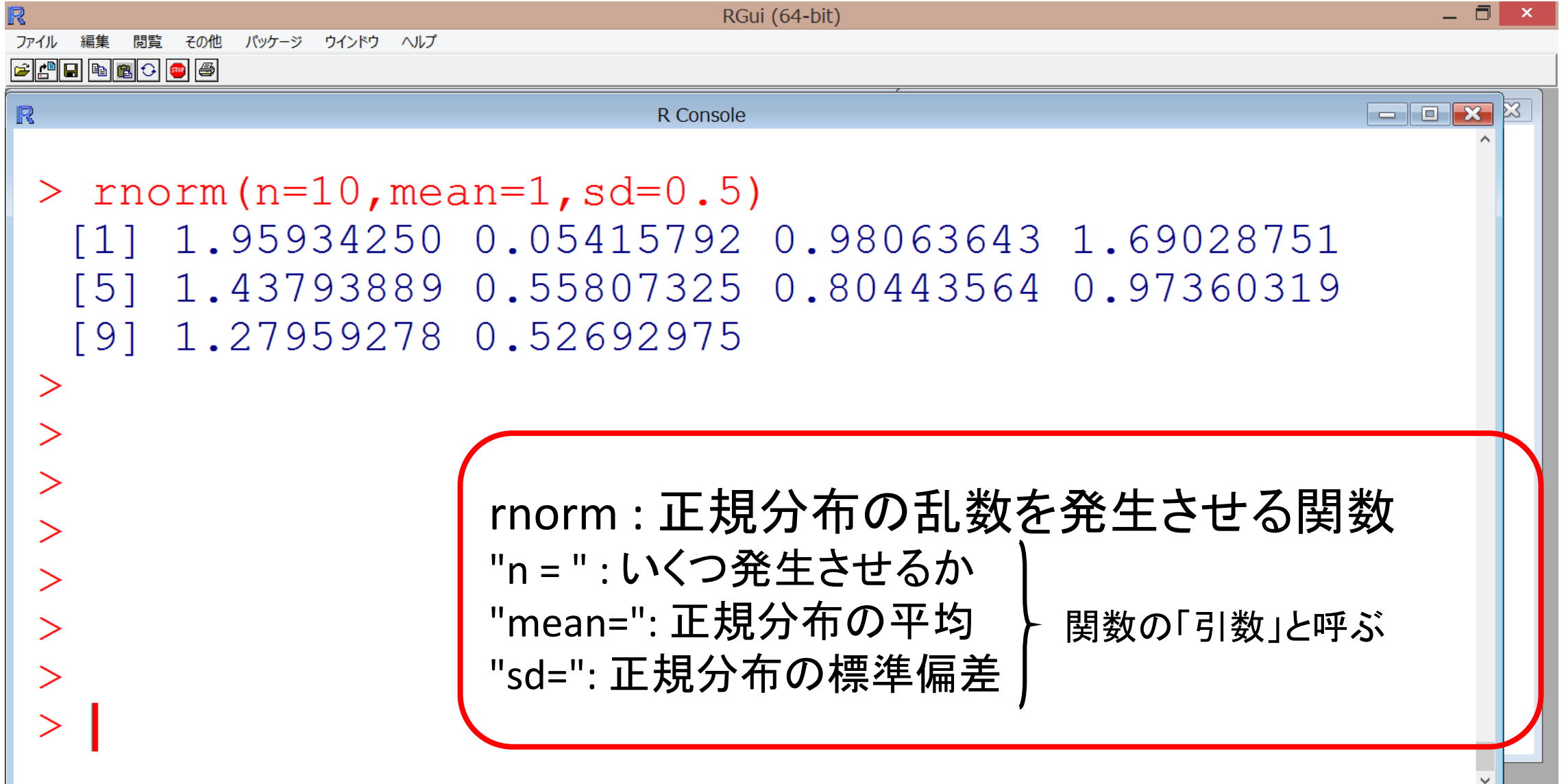
• グラフを書いてみる (2)



- グラフを書いてみる (3)

The screenshot displays the RGui (64-bit) interface. The main window is titled "RGui (64-bit)" and contains two panes. The left pane, titled "R Console", shows the command `pie(1:10)` entered in red text, with several red prompt characters (`>`) above and below it. The right pane, titled "R Graphics: Device 2 (ACTIVE)", displays a pie chart with 10 slices, numbered 1 through 10. The slices are colored in a sequence: 1 (light blue), 2 (light red), 3 (light cyan), 4 (light purple), 5 (yellow), 6 (light blue), 7 (light cyan), 8 (light red), 9 (light purple), and 10 (yellow). The chart is centered in the right pane.

• 乱数の発生 (1)



```
RGui (64-bit)
ファイル 編集 閲覧 その他 パッケージ ウィンドウ ヘルプ
R Console
> rnorm(n=10,mean=1,sd=0.5)
[1] 1.95934250 0.05415792 0.98063643 1.69028751
[5] 1.43793889 0.55807325 0.80443564 0.97360319
[9] 1.27959278 0.52692975
>
>
>
>
>
>
>
>
>
```

rnorm : 正規分布の乱数を発生させる関数
"n=" : いくつ発生させるか
"mean=" : 正規分布の平均
"sd=" : 正規分布の標準偏差

} 関数の「引数」と呼ぶ

• 乱数の発生 (2)

```
RGui (64-bit)
ファイル 編集 閲覧 その他 パッケージ ウィンドウ ヘルプ
R Console
> rnorm(n=10) ← 引数は省略可能. 省略すると「デフォルトの」値が
[1] -0.015459087 -0.008134011  2.634126584   使われる
[4]  1.445240956 -1.320343126  2.376047415
[7] -0.018060766 -0.671554868 -1.331151241
[10] -0.489313667
> args(rnorm) ← どんな値がデフォルトとして設定されているかは
function (n, mean = 0, sd = 1)      args関数で確認できる.
NULL                                  args(関数名)
>
>
> rnorm関数では, 平均=0, 標準偏差=1がデフォルト
> |
```

- 乱数の発生 (3)

```
RGui (64-bit)
ファイル 編集 閲覧 その他 パッケージ ウィンドウ ヘルプ
R
R Console
> mean(rnorm(10))
[1] 0.2206719
> mean(rnorm(1000))
[1] 0.01781741
> mean(rnorm(100000))
[1] -0.0008039798
>
>
>
>
>
>
>
```

本当に平均が0となっているか？

- 発生させる乱数の数を増やすほど、平均が0に近くなっている

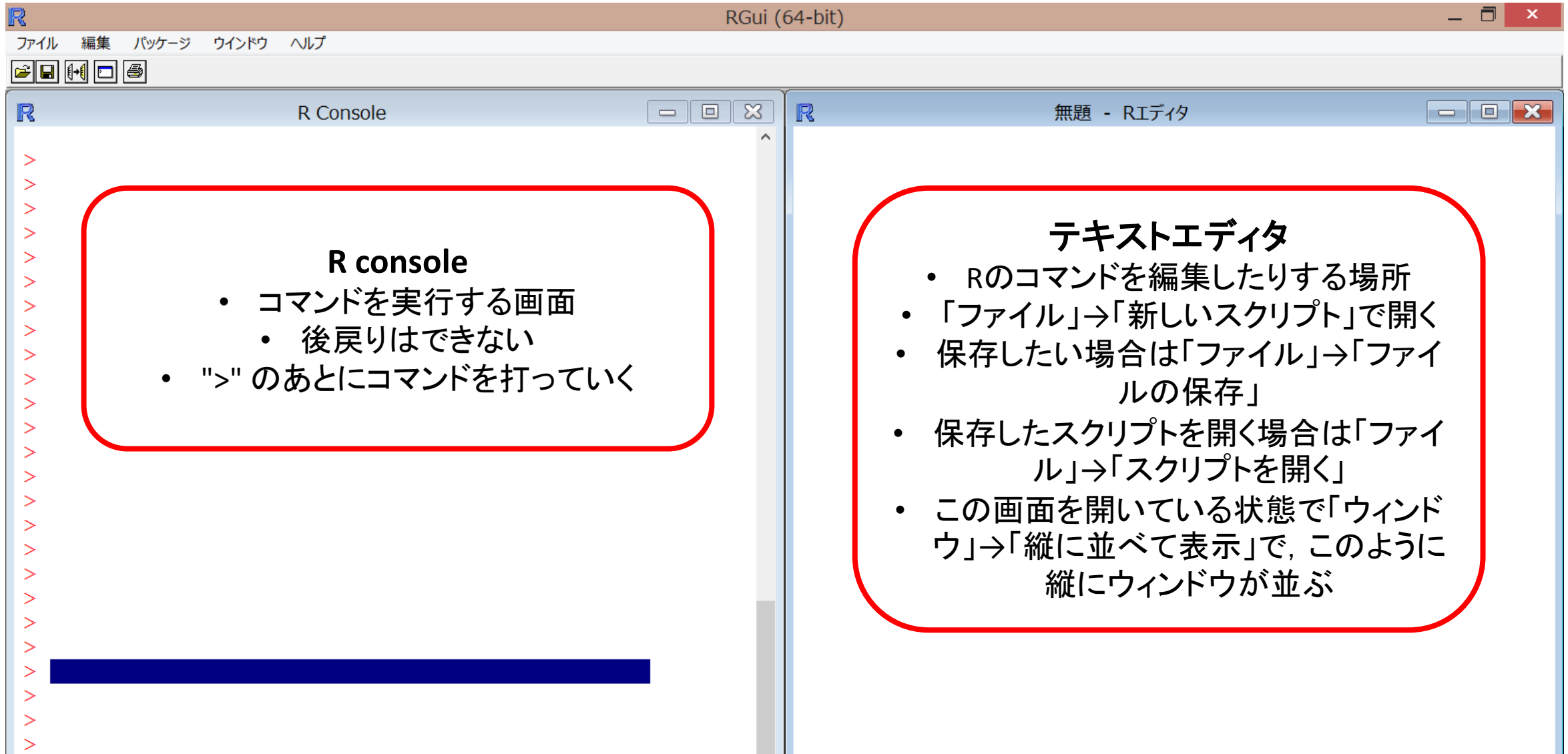
- 値の代入 " \leftarrow "

```
R
ファイル 編集 閲覧 その他 パッケージ ウインドウ ヘルプ
R Console
>
> x1 <- 1:10
> y1 <- rnorm(10)
> x1
[1] 1 2 3 4 5 6 7 8 9 10
> y1
[1] -0.37172657 0.70773649 -1.43890111 1.39473763
[5] -0.07382986 -0.78670341 -0.31269360 -0.17160203
[9] -0.35414872 1.24522649
> plot(x1, y1)
>
> |
```

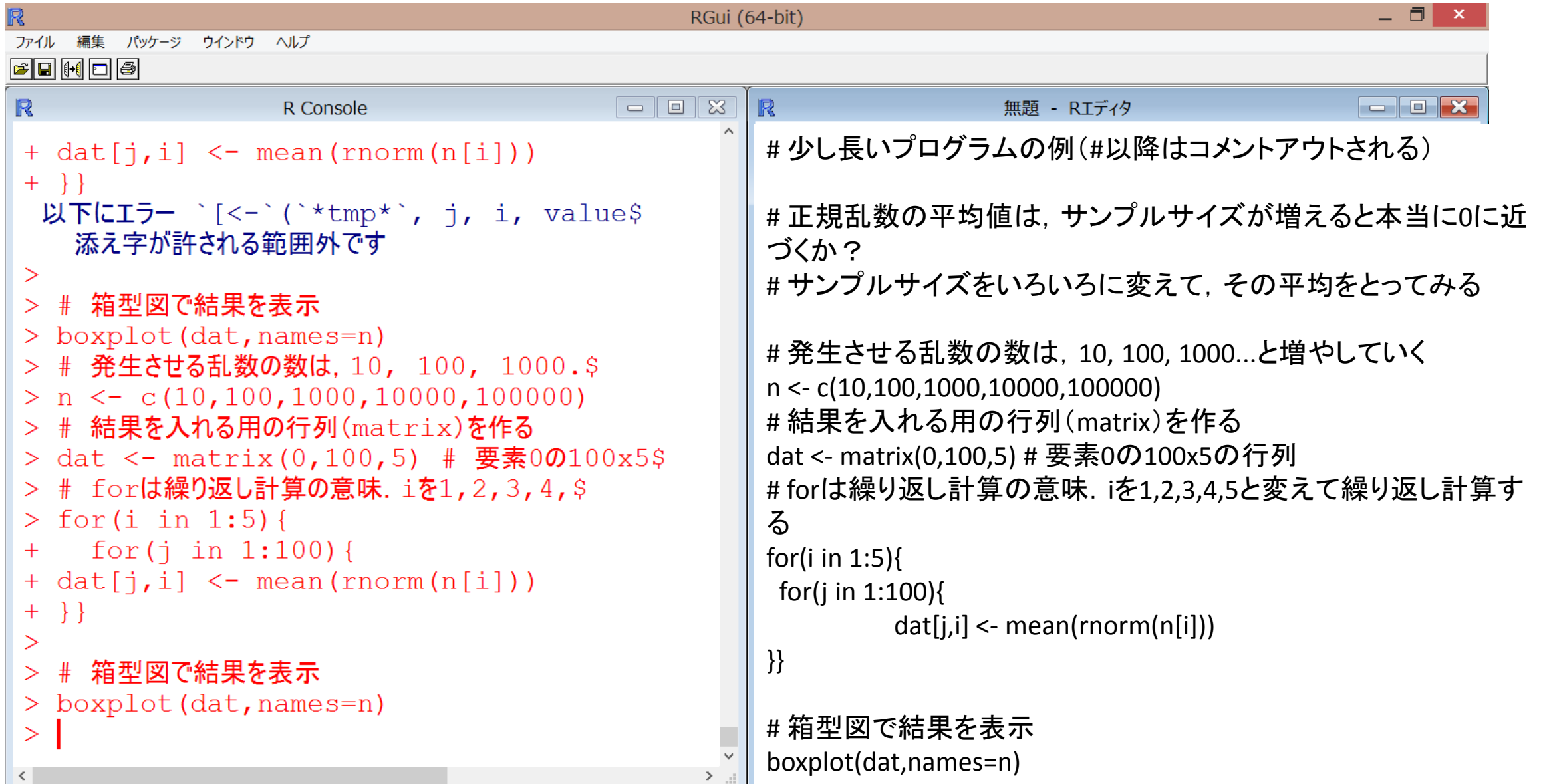
1から10までの数列を, x1という「オブジェクト」に代入
代入記号は " \leftarrow "

x1 (Enter)で, x1の中身
が表示される

- 長いプログラムを書く場合は？ → 例えば、このようにウィンドウを配置する（左；R console, 右；R エディタ）



- 右のテキストエディタでプログラムを編集→実行したい部分を選んで Control + R



```
RGui (64-bit)
ファイル 編集 パッケージ ウィンドウ ヘルプ
R Console
+ dat[j,i] <- mean(rnorm(n[i]))
+ }}
以下にエラー `[<-`(`*tmp*`, j, i, value$
添え字が許される範囲外です
>
> # 箱型図で結果を表示
> boxplot(dat, names=n)
> # 発生させる乱数の数は, 10, 100, 1000.$
> n <- c(10,100,1000,10000,100000)
> # 結果を入れる用の行列(matrix)を作る
> dat <- matrix(0,100,5) # 要素0の100x5$
> # forは繰り返し計算の意味. iを1,2,3,4,$
> for(i in 1:5){
+   for(j in 1:100){
+     dat[j,i] <- mean(rnorm(n[i]))
+   }}
>
> # 箱型図で結果を表示
> boxplot(dat, names=n)
> |

無題 - RIデータ
# 少し長いプログラムの例(#以降はコメントアウトされる)
# 正規乱数の平均値は, サンプルサイズが増えると本当に0に近づくか?
# サンプルサイズをいろいろに変えて, その平均をとってみる
# 発生させる乱数の数は, 10, 100, 1000...と増やしていく
n <- c(10,100,1000,10000,100000)
# 結果を入れる用の行列(matrix)を作る
dat <- matrix(0,100,5) # 要素0の100x5の行列
# forは繰り返し計算の意味. iを1,2,3,4,5と変えて繰り返し計算する
for(i in 1:5){
  for(j in 1:100){
    dat[j,i] <- mean(rnorm(n[i]))
  }}
# 箱型図で結果を表示
boxplot(dat, names=n)
```

結果のグラフ(barplot)

RGui (64-bit)

ファイル 履歴 サイズ変更 ウィンドウ

R Console

```
+ dat[j,i] <- mean(rnorm(n[i]))
+ }}
以下にエラー `<-` (`*tmp*`
添え字が許される範囲外です
>
> # 箱型図で結果を表示
> boxplot(dat, names=n)
> # 発生させる乱数の数は, 10
> n <- c(10, 100, 1000, 10000, 100000)
> # 結果を入れる用の行列(ma
> dat <- matrix(0, 100,
> # forは繰り返し計算の意味
> for(i in 1:5){
+   for(j in 1:100){
+   dat[j,i] <- mean(rno
+   }}
>
> # 箱型図で結果を表示
> boxplot(dat, names=n)
```

無題 - R1データ

```
# 少し長いプログラムの例(#以降はコメントアウトされる)
```

R Graphics: Device 2 (ACTIVE)

当に0に近づくか？
みる
と増やしていく
x5の行列
変えて繰り返し計算する

実習では..

- データの読み込み
- 様々なデータの集計、ソート、整形
- プロット
- 時空間データのプロット
- 簡単な統計解析

等を扱う予定です