Rで学ぶプログラミングの基礎の基礎

(4) 種々のベクトル

本日のメニュー

1. ベクトルの型
   - 実数値型ベクトル
   - 文字型ベクトル
   - 論理値型ベクトル
2. ベクトル型のチェックと変換
3. 文字列操作

ベクトルの型

- データの種類としては主に以下のようなものがある
  - 実数値：1.5 や 3
  - 複素数：1+2i（iは虚数）
  - 文字列："ABC"
  - 因　子："M" を「男性」，"F" を「女性」として扱う
    （"M" や "F" はカテゴリー）
  - 論理値：TRUE（真）と FALSE（偽）
  - 日付値："2013/10/15"

特殊な値

- 特殊な値についても、ついでにここで紹介
  - NULL：「空っぽ」「何もない」という意味
    （0 と混同しがちだが，NULL は「0 ですらない」）
  - NA ：「Not Available」の略で, 欠測（データが存在しない）
  - NaN ：「Not a Number」の略で，非数（0 / 0 の計算結果など）
  - Inf ： 無限大（∞）

実数値ベクトル

- 略して「数値ベクトル」とも呼ばれる
- 3 章までで扱っていたものは全て実数値型ベクトル

```
> x <- c(1, 2, 3)
```

- R では（良くも悪くも）実数と整数をあまり区別せずに処理することが多いが，データを明示的に整数として扱いたい場合，関数 as.integer() で整数型ベクトルに変換することも出来る

```
> y <- as.integer(x)
```

- 整数型については後で個人的見解を述べる

文字型ベクトル

- 文字列を要素とするベクトルを文字型ベクトルと呼ぶ
- 文字列は "（ダブルクォーテーション）で囲む必要がある

```
> x <- c("ABC", "DEF")
```

論理値型ベクトル

- TRUE（真）と FALSE（偽）を要素とするベクトルを論理値型ベクトル（または「論理型ベクトル」）と呼ぶ
- TRUE（真）や FALSE（偽）はそれぞれ，T と F と略記することも出来る
- 値同士の大小比較を行った結果は論理型ベクトルとなる

```
> 1 < 2        # 1 は 2 よりも小さい？
[1] TRUE
> 3 < 2        # 3 は 2 よりも小さい？
[1] FALSE
```

値を整数として扱いたい場合

- 関数 as.integer() で整数型ベクトルに変換することも出来るが，例えば 1.5 を整数型に変換してしまうと値が 1 に丸められてしまう
- 個人的には実数値型ベクトルは実数値型のまま値をもっておき，「値が整数かどうか」をチェックする場合は以下の関数を定義して使う

```
> is.wholenumber <- function(x, tol=.Machine$double.eps^0.5) abs(x-round(x)) < tol
```

- 関数 is.wholenumber の使用例を挙げる → 結果は論理値で返される

```
> is.wholenumber(1)
[1] TRUE
> is.wholenumber(1.5)
[1] FALSE
```

本日のメニュー

1. ベクトルの型
   - 実数値型ベクトル
   - 文字型ベクトル
   - 論理値型ベクトル
2. ベクトル型のチェックと変換
3. 文字列操作

ベクトル型のチェックと変換

- ベクトル型のチェックを行ったり，ある型から別の型に変換を行う場合は以下の関数を使用する

型	型のチェックを行う関数	型の変換を行う関数
実数値	is.numeric()	as.numeric()
整数値	is.integer()	as.integer()
複素数	is.complex()	as.complex()
文字列	is.character()	as.character()
因子	is.factor()	as.factor()
論理値	is.logical()	as.logical()
日付値	—	as.Date

ベクトル型のチェックと変換

```
> x <- "123"            # x は文字型
> typeof(x)             # x の型を調べる便利な関数
[1] "character"
> x + 1                 # 文字と数値は足し算できない
 以下にエラー x + 1 :  二項演算子の引数が数値ではありません
> y <- as.numeric(x)    # x を数値型に変換
> y + 1                 # 今度はうまく行く
[1] 124
```

演習問題①

1. 変数 x に 1 を代入して下さい
2. 1. で作成した変数 x が整数かどうかをチェックして下さい
3. 変数 x を整数型に変換し，変数 y に代入した後，
   変数 y が整数かどうかをチェックして下さい
4. 8 頁で作成した関数 is.wholenumber() は，整数かどうかを
   チェックする関数です
   関数 is.wholenumber() を使って，1. で作成した変数 x が
   整数かどうかをチェックして下さい

本日のメニュー

1. ベクトルの型
   - 実数値型ベクトル
   - 文字型ベクトル
   - 論理値型ベクトル
2. ベクトル型のチェックと変換
3. 文字列操作

文字列操作

- 数値以外のデータ処理の例として，文字列に関する処理の方法をいくつか紹介する

関数	意味
chartr(文字ベクトル1, 文字ベクトル2, 文字ベクトル3)	文字ベクトル3について，文字ベクトル1を文字ベクトル2に置換
grep("文字列", 文字ベクトル)	文字ベクトルの各要素について，"文字列"が含まれているかどうかをチェック（結果は 0 または 1 ）
nchar(文字ベクトル)	文字ベクトルの文字数を数える
paste(文字ベクトル1, 文字ベクトル2, sep="区切り文字")	文字ベクトル 1 と文字ベクトル 2 を結合する（区切り文字は引数 sep で指定する）
substring(文字ベクトル, a, b)	文字ベクトルについて a 文字目から b 文字目までを抽出
toupper(文字ベクトル)	文字ベクトルを大文字に変換
tolower(文字ベクトル)	文字ベクトルを小文字に変換
unlist(strsplit(文字ベクトル, ","))	文字ベクトルをカンマごとに区切った後，ベクトルに変換

文字列操作

```
> ( x <- "You are fool." )
[1] "You are fool."
> chartr("f", "c", x)        # x の中の f を c に置換
[1] "You are cool."
> nchar("12組456789番")      # 半角文字も全角文字も
                             # 1 文字としてカウント
[1] 10
```

文字列操作

```
> paste("ABC", "DEF")  # デフォルトでは文字列間にスペースが入る
[1] "ABC DEF"
> paste("ABC", "DEF", sep=",")   # 区切り文字は引数 sep で指定
[1] "ABC,DEF"
> paste("ABC", "DEF", sep="")    # 区切り文字を入れない場合
[1] "ABCDEF"


> substring("ABCDEF", 2, 5)      # "ABCDEF"の2～5文字目を抽出
[1] "BCDE"
```

演習問題②

1. 文字列 "1,2,3,4,5" を変数 x に代入して下さい.
2. 変数 x の文字数を調べてください.
3. 変数 x について，5 文字目を抽出して下さい.
4. 以下の命令を実行した後，変数 y の型を調べて下さい

```
> y <- unlist(strsplit(x, ",”))
```

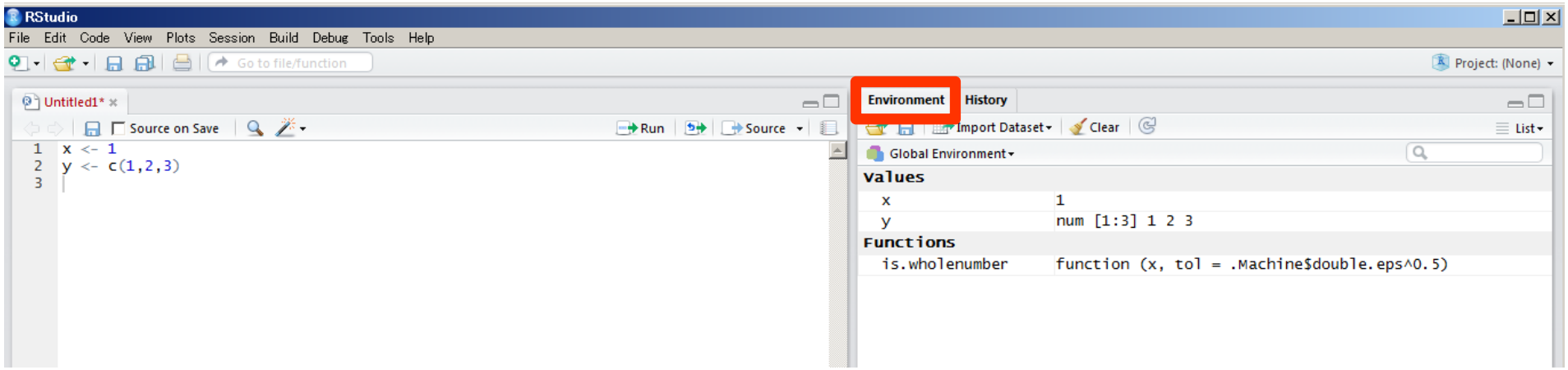
5. 変数 y を数値ベクトルに変換し，変数 z に代入して下さい

本日のメニュー

1. ベクトルの型
   - 実数値型ベクトル
   - 文字型ベクトル
   - 論理値型ベクトル
2. ベクトル型のチェックと変換
3. 文字列操作

おまけ

- 様々な変数や関数を作成したが，作成した変数や関数は右上の [Environment] のタブをクリックすることで一覧として見ることが出来る



このスライドの内容が書籍になっています

- 舟尾 暢男 『 R で学ぶプログラミングの基礎の基礎（カットシステム）』

「R と RStudio のインストール」「R の基礎」「変数とベクトル」
「種々のベクトル」「関数の作成」「条件分岐」「くり返し」
「複雑なくり返し」「グラフ作成の概要」「問題集」「簡単なゲーム作成」

※ 他のベクトル型の解説や本資料の演習問題の解答も載っています

Rで学ぶプログラミングの基礎の基礎

終