Chapter1. Basics in R

Yerim Lim

Jan 31, 2018

벡터의 생성

```
a <- 1
x <- 3
a
```

```
## [1] 1
```

```
print(a)
```

```
## [1] 1
```

```
print(x)
```

문자형으로 이루어진 벡터

```
x <- c("ree","fie","fum")
print(x)
```

```
## [1] "ree" "fie" "fum"
```

```
c("eveyone","loves","stats")
```

```
## [1] "eveyone" "loves"   "stats"
```

```
c(1,1,2,3,5,6,13,21)
```

```
## [1]  1  1  2  3  5  6 13 21
```

벡터 내 계산

```
c(1*pi, 2*pi, 3*pi, 4*pi)
```

```
## [1]  3.141593  6.283185  9.424778 12.566371
```

```
a <- 1:4  ;a*pi
```

```
## [1]  3.141593  6.283185  9.424778 12.566371
```

논리형 벡터

```
c(TRUE, FALSE,TRUE)
```

```
## [1]  TRUE FALSE  TRUE
```

변수의 결합

```
v1 <- 1:3
v2 <- 4:5
v3 <- c("A","B", "C")
c(v1,v2,v3)
```

```
## [1] "1" "2" "3" "4" "5" "A" "B" "C"
```

수열

```
1:5
```

```
## [1] 1 2 3 4 5
```

```
b<- 2:10
b
```

```
## [1]  2  3  4  5  6  7  8  9 10
```

```
10:19
```

```
##  [1] 10 11 12 13 14 15 16 17 18 19
```

```
9:0
```

```
##  [1] 9 8 7 6 5 4 3 2 1 0
```

```
e <- 10:2
e
```

```
## [1] 10  9  8  7  6  5  4  3  2
```

seq함수

```
seq(from=0, to=20, by=2)
```

```
##  [1]  0  2  4  6  8 10 12 14 16 18 20
```

```
seq(0,20,2)
```

```
##  [1]  0  2  4  6  8 10 12 14 16 18 20
```

```
seq(from=0, to=20, length.out = 5)
```

```
## [1]  0  5 10 15 20
```

```
seq(1.0, 2.0, length.out = 5)
```

```
## [1] 1.00 1.25 1.50 1.75 2.00
```

```
seq(0,10,by=1)
```

```
##  [1]  0  1  2  3  4  5  6  7  8  9 10
```

```
seq(0,10,length=20)
```

```
##  [1]  0.0000000  0.5263158  1.0526316  1.5789474  2.1052632  2.6315789
##  [7]  3.1578947  3.6842105  4.2105263  4.7368421  5.2631579  5.7894737
## [13]  6.3157895  6.8421053  7.3684211  7.8947368  8.4210526  8.9473684
## [19]  9.4736842 10.0000000
```

```
n <- 0
1:n
```

```
## [1] 1 0
```

rep함수

```
rep(1,times=5)
```

```
## [1] 1 1 1 1 1
```

```
rep(1:2, each=2)
```

```
## [1] 1 1 2 2
```

```
rep(1:2, times=2)
```

```
## [1] 1 2 1 2
```

```
c <- 1:5
c
```

```
## [1] 1 2 3 4 5
```

```
rep(c,5) #  c 5   .
```

```
##  [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
```

```
rep(c, each=5)
```

```
##  [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4 5 5 5 5 5
```

```
rep(c, times=5)
```

```
##  [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
```

데이터의 유형

Numeric

```
a <- 3
a
```

```
## [1] 3
```

Character

```
b <- "Character"
b
```

```
## [1] "Character"
```

paste

```
A <- c("A","B","C")
A
```

```
## [1] "A" "B" "C"
```

```
paste("a","b",sep=" ")
```

```
## [1] "a b"
```

```
paste(A, c("d","e"))
```

```
## [1] "A d" "B e" "C d"
```

```
paste(A, c("d"))
```

```
## [1] "A d" "B d" "C d"
```

```
paste(A, c("d","e","t"))
```

```
## [1] "A d" "B e" "C t"
```

```
f <- paste(A,10)
f
```

```
## [1] "A 10" "B 10" "C 10"
```

```
paste(A,10,sep="")
```

```
## [1] "A10" "B10" "C10"
```

paste 기능은 변수명을 할당할 때 쓰기 유용하다. 아래와 같이 사용하면 좋다.

```
paste(A, 1:10, sep="")
```

```
##  [1] "A1"  "B2"  "C3"  "A4"  "B5"  "C6"  "A7"  "B8"  "C9"  "A10"
```

응용해보자

```
paste("everybody","loves","cats")
```

```
## [1] "everybody loves cats"
```

```
paste("everybody","loves","cats", sep="")
```

```
## [1] "everybodylovescats"
```

```
paste("everybody","loves","cats", sep="-")
```

```
## [1] "everybody-loves-cats"
```

Substr 함수

substr(문자열, 시작, 끝)

```
substr("BigDataAnalysis",1,4)
```

```
## [1] "BigD"
```

벡터에도 사용할 있다.

```
ss <- c("Moe","Larry","Curly")
substr(ss, 1,3)
```

```
## [1] "Moe" "Lar" "Cur"
```

논리값

```
c <- TRUE
c
```

```
## [1] TRUE
```

```
d <- T
d
```

```
## [1] TRUE
```

```
e <- F
e
```

```
## [1] FALSE
```

```
a <- 3
a==pi
```

```
## [1] FALSE
```

```
a!=pi
```

```
## [1] TRUE
```

```
a<pi
```

```
## [1] TRUE
```

```
a>pi
```

```
## [1] FALSE
```

```
a>=pi
```

```
## [1] FALSE
```

```
a<=pi
```

```
## [1] TRUE
```

matrix

```
theDAta <- c(1.1,1.2,2.1,2.2,3.1,3.2)
mat <- matrix(theDAta,2,3)
mat
```

```
##      [,1] [,2] [,3]
## [1,]  1.1  2.1  3.1
## [2,]  1.2  2.2  3.2
```

```
dim(mat)
```

```
## [1] 2 3
```

```
mat
```

```
##      [,1] [,2] [,3]
## [1,]  1.1  2.1  3.1
## [2,]  1.2  2.2  3.2
```

```
t(mat)
```

```
##      [,1] [,2]
## [1,]  1.1  1.2
## [2,]  2.1  2.2
## [3,]  3.1  3.2
```

```
mat %*% t(mat)
```

```
##       [,1]  [,2]
## [1,] 15.23 15.86
## [2,] 15.86 16.52
```

```
diag(mat)
```

```
## [1] 1.1 2.2
```

```
colnames(mat) <- paste("C" ,1:3, sep="")
colnames(mat)
```

```
## [1] "C1" "C2" "C3"
```

```
rownames(mat) <- paste("R", 1:2, sep="")
rownames(mat)
```

```
## [1] "R1" "R2"
```

```
mat
```

```
##     C1  C2  C3
## R1 1.1 2.1 3.1
## R2 1.2 2.2 3.2
```

```
mat[1,]
```

```
##  C1  C2  C3
## 1.1 2.1 3.1
```

```
mat[,3]
```

```
##  R1  R2
## 3.1 3.2
```

```
A <- matrix(0,4,5)
A
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    0    0    0    0
## [2,]    0    0    0    0    0
## [3,]    0    0    0    0    0
## [4,]    0    0    0    0    0
```

```
A <- matrix(1:20, 4,5)
A
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    5    9   13   17
## [2,]    2    6   10   14   18
## [3,]    3    7   11   15   19
## [4,]    4    8   12   16   20
```

```
A[c(1,4), c(2,3)]
```

```
##      [,1] [,2]
## [1,]    5    9
## [2,]    8   12
```

지정한 위치의 원소에 지정한 수를 할당해보자.

```
A[c(1,4), c(2,3)] <- 1
A
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    1    1   13   17
## [2,]    2    6   10   14   18
## [3,]    3    7   11   15   19
## [4,]    4    1    1   16   20
```

```
A+1
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    2    2    2   14   18
## [2,]    3    7   11   15   19
## [3,]    4    8   12   16   20
## [4,]    5    2    2   17   21
```

List

```
lst <- list(3.14, "MOE", c(1,1,2,3), mean)
lst
```

```
## [[1]]
## [1] 3.14
##
## [[2]]
## [1] "MOE"
##
## [[3]]
## [1] 1 1 2 3
##
## [[4]]
## function (x, ...)
## UseMethod("mean")
## <bytecode: 0x000000001a38f3a0>
## <environment: namespace:base>
```

각 리스트의 원소들에 태그를 부여하자.

```
a <- 1:10
b <- matrix(1:10,2,5)
c <- c("name1","name2")

alst <- list(a=a, b=b,c=c)
alst
```

```
## $a
##  [1]  1  2  3  4  5  6  7  8  9 10
##
## $b
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    3    5    7    9
## [2,]    2    4    6    8   10
##
## $c
## [1] "name1" "name2"
```

```
blst <- list(d=2:10*10)
blst
```

```
## $d
## [1]  20  30  40  50  60  70  80  90 100
```

```
alst$a
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
alst[2]
```

```
## $b
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    3    5    7    9
## [2,]    2    4    6    8   10
```

```
alst[[2]]
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    3    5    7    9
## [2,]    2    4    6    8   10
```

리스트를 벡터함수를 통해 합칠 수있다.

```
ablst <- c(alst,blst)
ablst
```

```
## $a
##  [1]  1  2  3  4  5  6  7  8  9 10
##
## $b
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    3    5    7    9
## [2,]    2    4    6    8   10
##
## $c
## [1] "name1" "name2"
##
## $d
## [1]  20  30  40  50  60  70  80  90 100
```

```
str(alst)
```

```
## List of 3
##  $ a: int [1:10] 1 2 3 4 5 6 7 8 9 10
##  $ b: int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
##  $ c: chr [1:2] "name1" "name2"
```

```
str(blst)
```

```
## List of 1
##  $ d: num [1:9] 20 30 40 50 60 70 80 90 100
```

```
str(ablst)
```

```
## List of 4
##  $ a: int [1:10] 1 2 3 4 5 6 7 8 9 10
##  $ b: int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
##  $ c: chr [1:2] "name1" "name2"
```

```
##  $ d: num [1:9] 20 30 40 50 60 70 80 90 100
```

```
score1 <- list(10,20,30,40,50)
score2 <- list(c("a","b"))
```

```
score1[score1>40]
```

```
## [[1]]
## [1] 50
```

리스트에 대한 이해를 높이기 위해 구조를 바꾸어보자. 값 5개를 5개의 리스트로 쪼개어 할당하지 않고, 5개 값아ㅡㄹ 하나의 벡터로 묶어 하나의 리스트로 입력한다면 코딩이 어떻게 변할까?

```
score1a <- list(seq(10,50, by=10))
# score1a[score1a>40] #      .          .
```

```
score1a[[1]][score1a[[1]]>40]
```

```
## [1] 50
```

리스트 형식을 합쳐보자.

```
score12 <- list(score1, score2)
score12
```

```
## [[1]]
## [[1]][[1]]
## [1] 10
##
## [[1]][[2]]
## [1] 20
##
## [[1]][[3]]
## [1] 30
##
## [[1]][[4]]
## [1] 40
##
## [[1]][[5]]
## [1] 50
##
##
## [[2]]
## [[2]][[1]]
## [1] "a" "b"
```

리스트의 하위항목이 늘어난 것을 볼 있다.

합쳐린 리스트를 여러경우로 조회해보자.

```
score12[1]
```

```
## [[1]]
## [[1]][[1]]
## [1] 10
##
## [[1]][[2]]
## [1] 20
##
## [[1]][[3]]
```

```
## [1] 30
##
## [[1]][[4]]
## [1] 40
##
## [[1]][[5]]
## [1] 50
```

```
score12[[1]]
```

```
## [[1]]
## [1] 10
##
## [[2]]
## [1] 20
##
## [[3]]
## [1] 30
##
## [[4]]
## [1] 40
##
## [[5]]
## [1] 50
```

```
score12[[1]][1]
```

```
## [[1]]
## [1] 10
```

```
# score12[[[1]]]
```

위 방법으로 실행시 에러가 난다.

데이터 프레임

```
a=c(1,2,4,6,3,4)
b=c(6,4,2,4,3.2,4)
c=c(7,6,4,2,5,6)
d=c(2,4,3,1,5,6)
e=data.frame(a,b,c,d)
e
```

```
##   a   b c d
## 1 1 6.0 7 2
## 2 2 4.0 6 4
## 3 4 2.0 4 3
## 4 6 4.0 2 1
## 5 3 3.2 5 5
## 6 4 4.0 6 6
```

```
data(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
```

```
data(iris)
newRow <- data.frame(Sepal.Length=3.0, Sepal.Width=3.2, Petal.Length=1.6, Petal.Width=0.3, Species="news
newRow
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 1            3         3.2          1.6         0.3 newsetosa
```

```
iris1 <- rbind(iris, newRow)
iris1
```

```
##     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
## 17           5.4         3.9          1.3         0.4     setosa
## 18           5.1         3.5          1.4         0.3     setosa
## 19           5.7         3.8          1.7         0.3     setosa
## 20           5.1         3.8          1.5         0.3     setosa
## 21           5.4         3.4          1.7         0.2     setosa
## 22           5.1         3.7          1.5         0.4     setosa
## 23           4.6         3.6          1.0         0.2     setosa
## 24           5.1         3.3          1.7         0.5     setosa
## 25           4.8         3.4          1.9         0.2     setosa
## 26           5.0         3.0          1.6         0.2     setosa
## 27           5.0         3.4          1.6         0.4     setosa
## 28           5.2         3.5          1.5         0.2     setosa
## 29           5.2         3.4          1.4         0.2     setosa
## 30           4.7         3.2          1.6         0.2     setosa
## 31           4.8         3.1          1.6         0.2     setosa
## 32           5.4         3.4          1.5         0.4     setosa
## 33           5.2         4.1          1.5         0.1     setosa
## 34           5.5         4.2          1.4         0.2     setosa
## 35           4.9         3.1          1.5         0.2     setosa
## 36           5.0         3.2          1.2         0.2     setosa
## 37           5.5         3.5          1.3         0.2     setosa
## 38           4.9         3.6          1.4         0.1     setosa
## 39           4.4         3.0          1.3         0.2     setosa
## 40           5.1         3.4          1.5         0.2     setosa
```

```
## 41          5.0         3.5          1.3         0.3     setosa
## 42          4.5         2.3          1.3         0.3     setosa
## 43          4.4         3.2          1.3         0.2     setosa
## 44          5.0         3.5          1.6         0.6     setosa
## 45          5.1         3.8          1.9         0.4     setosa
## 46          4.8         3.0          1.4         0.3     setosa
## 47          5.1         3.8          1.6         0.2     setosa
## 48          4.6         3.2          1.4         0.2     setosa
## 49          5.3         3.7          1.5         0.2     setosa
## 50          5.0         3.3          1.4         0.2     setosa
## 51          7.0         3.2          4.7         1.4 versicolor
## 52          6.4         3.2          4.5         1.5 versicolor
## 53          6.9         3.1          4.9         1.5 versicolor
## 54          5.5         2.3          4.0         1.3 versicolor
## 55          6.5         2.8          4.6         1.5 versicolor
## 56          5.7         2.8          4.5         1.3 versicolor
## 57          6.3         3.3          4.7         1.6 versicolor
## 58          4.9         2.4          3.3         1.0 versicolor
## 59          6.6         2.9          4.6         1.3 versicolor
## 60          5.2         2.7          3.9         1.4 versicolor
## 61          5.0         2.0          3.5         1.0 versicolor
## 62          5.9         3.0          4.2         1.5 versicolor
## 63          6.0         2.2          4.0         1.0 versicolor
## 64          6.1         2.9          4.7         1.4 versicolor
## 65          5.6         2.9          3.6         1.3 versicolor
## 66          6.7         3.1          4.4         1.4 versicolor
## 67          5.6         3.0          4.5         1.5 versicolor
## 68          5.8         2.7          4.1         1.0 versicolor
## 69          6.2         2.2          4.5         1.5 versicolor
## 70          5.6         2.5          3.9         1.1 versicolor
## 71          5.9         3.2          4.8         1.8 versicolor
## 72          6.1         2.8          4.0         1.3 versicolor
## 73          6.3         2.5          4.9         1.5 versicolor
## 74          6.1         2.8          4.7         1.2 versicolor
## 75          6.4         2.9          4.3         1.3 versicolor
## 76          6.6         3.0          4.4         1.4 versicolor
## 77          6.8         2.8          4.8         1.4 versicolor
## 78          6.7         3.0          5.0         1.7 versicolor
## 79          6.0         2.9          4.5         1.5 versicolor
## 80          5.7         2.6          3.5         1.0 versicolor
## 81          5.5         2.4          3.8         1.1 versicolor
## 82          5.5         2.4          3.7         1.0 versicolor
## 83          5.8         2.7          3.9         1.2 versicolor
## 84          6.0         2.7          5.1         1.6 versicolor
## 85          5.4         3.0          4.5         1.5 versicolor
## 86          6.0         3.4          4.5         1.6 versicolor
## 87          6.7         3.1          4.7         1.5 versicolor
## 88          6.3         2.3          4.4         1.3 versicolor
## 89          5.6         3.0          4.1         1.3 versicolor
## 90          5.5         2.5          4.0         1.3 versicolor
## 91          5.5         2.6          4.4         1.2 versicolor
## 92          6.1         3.0          4.6         1.4 versicolor
## 93          5.8         2.6          4.0         1.2 versicolor
## 94          5.0         2.3          3.3         1.0 versicolor
```

```
## 95          5.6         2.7          4.2         1.3 versicolor
## 96          5.7         3.0          4.2         1.2 versicolor
## 97          5.7         2.9          4.2         1.3 versicolor
## 98          6.2         2.9          4.3         1.3 versicolor
## 99          5.1         2.5          3.0         1.1 versicolor
## 100         5.7         2.8          4.1         1.3 versicolor
## 101         6.3         3.3          6.0         2.5  virginica
## 102         5.8         2.7          5.1         1.9  virginica
## 103         7.1         3.0          5.9         2.1  virginica
## 104         6.3         2.9          5.6         1.8  virginica
## 105         6.5         3.0          5.8         2.2  virginica
## 106         7.6         3.0          6.6         2.1  virginica
## 107         4.9         2.5          4.5         1.7  virginica
## 108         7.3         2.9          6.3         1.8  virginica
## 109         6.7         2.5          5.8         1.8  virginica
## 110         7.2         3.6          6.1         2.5  virginica
## 111         6.5         3.2          5.1         2.0  virginica
## 112         6.4         2.7          5.3         1.9  virginica
## 113         6.8         3.0          5.5         2.1  virginica
## 114         5.7         2.5          5.0         2.0  virginica
## 115         5.8         2.8          5.1         2.4  virginica
## 116         6.4         3.2          5.3         2.3  virginica
## 117         6.5         3.0          5.5         1.8  virginica
## 118         7.7         3.8          6.7         2.2  virginica
## 119         7.7         2.6          6.9         2.3  virginica
## 120         6.0         2.2          5.0         1.5  virginica
## 121         6.9         3.2          5.7         2.3  virginica
## 122         5.6         2.8          4.9         2.0  virginica
## 123         7.7         2.8          6.7         2.0  virginica
## 124         6.3         2.7          4.9         1.8  virginica
## 125         6.7         3.3          5.7         2.1  virginica
## 126         7.2         3.2          6.0         1.8  virginica
## 127         6.2         2.8          4.8         1.8  virginica
## 128         6.1         3.0          4.9         1.8  virginica
## 129         6.4         2.8          5.6         2.1  virginica
## 130         7.2         3.0          5.8         1.6  virginica
## 131         7.4         2.8          6.1         1.9  virginica
## 132         7.9         3.8          6.4         2.0  virginica
## 133         6.4         2.8          5.6         2.2  virginica
## 134         6.3         2.8          5.1         1.5  virginica
## 135         6.1         2.6          5.6         1.4  virginica
## 136         7.7         3.0          6.1         2.3  virginica
## 137         6.3         3.4          5.6         2.4  virginica
## 138         6.4         3.1          5.5         1.8  virginica
## 139         6.0         3.0          4.8         1.8  virginica
## 140         6.9         3.1          5.4         2.1  virginica
## 141         6.7         3.1          5.6         2.4  virginica
## 142         6.9         3.1          5.1         2.3  virginica
## 143         5.8         2.7          5.1         1.9  virginica
## 144         6.8         3.2          5.9         2.3  virginica
## 145         6.7         3.3          5.7         2.5  virginica
## 146         6.7         3.0          5.2         2.3  virginica
## 147         6.3         2.5          5.0         1.9  virginica
## 148         6.5         3.0          5.2         2.0  virginica
```

```
## 149          6.2         3.4          5.4         2.3  virginica
## 150          5.9         3.0          5.1         1.8  virginica
## 151          3.0         3.2          1.6         0.3  newsetosa
```

```
dim(iris); dim(iris1)
```

```
## [1] 150   5
```

```
## [1] 151   5
```

rbind를 할 때는, 각 열의 이름이 같아야한다. 일치시킬 것.

새로운 데이터 프레임을 만들어서 새로운 예제를 만들어보자.

```
name <- c("john","peter","jennifer")
gender <- factor(c("m","m","f"))
hw1 <- c(60,60,80)
hw2 <- c(40,50,30)
grades <- data.frame(name, gender, hw1, hw2)
grades
```

```
##       name gender hw1 hw2
## 1     john      m  60  40
## 2    peter      m  60  50
## 3 jennifer      f  80  30
```

```
grades[1,2]
```

```
## [1] m
## Levels: f m
```

```
grades[,"name"] #same result
```

```
## [1] john     peter    jennifer
## Levels: jennifer john peter
```

```
grades$name #same result
```

```
## [1] john     peter    jennifer
## Levels: jennifer john peter
```

```
grades[grades$gender=="m",]
```

```
##    name gender hw1 hw2
## 1  john      m  60  40
## 2 peter      m  60  50
```

```
grades[,"hw1"]
```

```
## [1] 60 60 80
```

subset함수 이용하기

```
data(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
```

```r
subset(iris, select = Species, subset=(Petal.Length>1.7)) #select: column you want to showup, suset: lo
```

```
##        Species
## 25      setosa
## 45      setosa
## 51  versicolor
## 52  versicolor
## 53  versicolor
## 54  versicolor
## 55  versicolor
## 56  versicolor
## 57  versicolor
## 58  versicolor
## 59  versicolor
## 60  versicolor
## 61  versicolor
## 62  versicolor
## 63  versicolor
## 64  versicolor
## 65  versicolor
## 66  versicolor
## 67  versicolor
## 68  versicolor
## 69  versicolor
## 70  versicolor
## 71  versicolor
## 72  versicolor
## 73  versicolor
## 74  versicolor
## 75  versicolor
## 76  versicolor
## 77  versicolor
## 78  versicolor
## 79  versicolor
## 80  versicolor
## 81  versicolor
## 82  versicolor
## 83  versicolor
## 84  versicolor
## 85  versicolor
## 86  versicolor
## 87  versicolor
## 88  versicolor
## 89  versicolor
## 90  versicolor
## 91  versicolor
## 92  versicolor
## 93  versicolor
## 94  versicolor
## 95  versicolor
## 96  versicolor
```

```
## 97  versicolor
## 98  versicolor
## 99  versicolor
## 100 versicolor
## 101  virginica
## 102  virginica
## 103  virginica
## 104  virginica
## 105  virginica
## 106  virginica
## 107  virginica
## 108  virginica
## 109  virginica
## 110  virginica
## 111  virginica
## 112  virginica
## 113  virginica
## 114  virginica
## 115  virginica
## 116  virginica
## 117  virginica
## 118  virginica
## 119  virginica
## 120  virginica
## 121  virginica
## 122  virginica
## 123  virginica
## 124  virginica
## 125  virginica
## 126  virginica
## 127  virginica
## 128  virginica
## 129  virginica
## 130  virginica
## 131  virginica
## 132  virginica
## 133  virginica
## 134  virginica
## 135  virginica
## 136  virginica
## 137  virginica
## 138  virginica
## 139  virginica
## 140  virginica
## 141  virginica
## 142  virginica
## 143  virginica
## 144  virginica
## 145  virginica
## 146  virginica
## 147  virginica
## 148  virginica
## 149  virginica
## 150  virginica
```

```r
subset(iris,subset=(Petal.Length>1.7) , select = Species) # same result
```

```
##        Species
## 25      setosa
## 45      setosa
## 51  versicolor
## 52  versicolor
## 53  versicolor
## 54  versicolor
## 55  versicolor
## 56  versicolor
## 57  versicolor
## 58  versicolor
## 59  versicolor
## 60  versicolor
## 61  versicolor
## 62  versicolor
## 63  versicolor
## 64  versicolor
## 65  versicolor
## 66  versicolor
## 67  versicolor
## 68  versicolor
## 69  versicolor
## 70  versicolor
## 71  versicolor
## 72  versicolor
## 73  versicolor
## 74  versicolor
## 75  versicolor
## 76  versicolor
## 77  versicolor
## 78  versicolor
## 79  versicolor
## 80  versicolor
## 81  versicolor
## 82  versicolor
## 83  versicolor
## 84  versicolor
## 85  versicolor
## 86  versicolor
## 87  versicolor
## 88  versicolor
## 89  versicolor
## 90  versicolor
## 91  versicolor
## 92  versicolor
## 93  versicolor
## 94  versicolor
## 95  versicolor
## 96  versicolor
## 97  versicolor
## 98  versicolor
## 99  versicolor
```

```
## 100 versicolor
## 101  virginica
## 102  virginica
## 103  virginica
## 104  virginica
## 105  virginica
## 106  virginica
## 107  virginica
## 108  virginica
## 109  virginica
## 110  virginica
## 111  virginica
## 112  virginica
## 113  virginica
## 114  virginica
## 115  virginica
## 116  virginica
## 117  virginica
## 118  virginica
## 119  virginica
## 120  virginica
## 121  virginica
## 122  virginica
## 123  virginica
## 124  virginica
## 125  virginica
## 126  virginica
## 127  virginica
## 128  virginica
## 129  virginica
## 130  virginica
## 131  virginica
## 132  virginica
## 133  virginica
## 134  virginica
## 135  virginica
## 136  virginica
## 137  virginica
## 138  virginica
## 139  virginica
## 140  virginica
## 141  virginica
## 142  virginica
## 143  virginica
## 144  virginica
## 145  virginica
## 146  virginica
## 147  virginica
## 148  virginica
## 149  virginica
## 150  virginica
```

```
subset(iris, c(Sepal.Length, Petal.Length, Species),
       subset=((Sepal.Width==3.0)&(Petal.Width==0.2)))
```

```
##    Sepal.Length Petal.Length Species
## 2           4.9          1.4  setosa
## 26          5.0          1.6  setosa
## 39          4.4          1.3  setosa
```

With 함수

```
#with( dataframe, columnname)
head(with(iris, Species))
```

```
## [1] setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica
```

merge 함수

```
name <- c("Moe","Larry","Curly","Harry")
year.born <- c(1887, 1902, 1903, 1964)
place.born <- c("BensonHurst","Philadelphia","Brooklyn","Moscow")
born <- data.frame(name, year.born, place.born)
born
```

```
##    name year.born   place.born
## 1   Moe      1887  BensonHurst
## 2 Larry      1902 Philadelphia
## 3 Curly      1903     Brooklyn
## 4 Harry      1964       Moscow
```

```
name <- c("Moe","Curly","Larry")
year.died <- c(1952, 1975, 1975)
died <- data.frame(name, year.died)
died
```

```
##    name year.died
## 1   Moe      1952
## 2 Curly      1975
## 3 Larry      1975
```

```
merge(born,died, by="name")
```

```
##    name year.born   place.born year.died
## 1 Curly      1903     Brooklyn      1975
## 2 Larry      1902 Philadelphia      1975
## 3   Moe      1887  BensonHurst      1952
```

다른 예제를 풀어보자.

```
data(mtcars)
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
```

```
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
colnames(mtcars)
```

```
##  [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear"
## [11] "carb"
```

```
mtcars[1:5, c("mpg", "cyl")]
```

```
##                   mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Datsun 710        22.8   4
## Hornet 4 Drive    21.4   6
## Hornet Sportabout 18.7   8
```

```
subset(mtcars, select=c("mpg","cyl"), cyl>=6 )[1:6,]
```

```
##                   mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Hornet 4 Drive    21.4   6
## Hornet Sportabout 18.7   8
## Valiant           18.1   6
## Duster 360        14.3   8
```

```
head(mtcars[ mtcars$cyl>=6,c("mpg","cyl") ], n=6)
```

```
##                   mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Hornet 4 Drive    21.4   6
## Hornet Sportabout 18.7   8
## Valiant           18.1   6
## Duster 360        14.3   8
```

```
subset(mtcars, c("mpg","cyl")  , subset=c(cyl>=6 & mpg>=15))
```

```
##                   mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Hornet 4 Drive    21.4   6
## Hornet Sportabout 18.7   8
## Valiant           18.1   6
## Merc 280          19.2   6
## Merc 280C         17.8   6
## Merc 450SE        16.4   8
## Merc 450SL        17.3   8
## Merc 450SLC       15.2   8
## Dodge Challenger  15.5   8
## AMC Javelin       15.2   8
## Pontiac Firebird  19.2   8
## Ford Pantera L    15.8   8
## Ferrari Dino      19.7   6
## Maserati Bora     15.0   8
```

```
mtcars[ mtcars$cyl>=6 & mtcars$mpg>=15, c("mpg","cyl")]
```

```
##                    mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Hornet 4 Drive    21.4   6
## Hornet Sportabout 18.7   8
## Valiant           18.1   6
## Merc 280          19.2   6
## Merc 280C         17.8   6
## Merc 450SE        16.4   8
## Merc 450SL        17.3   8
## Merc 450SLC       15.2   8
## Dodge Challenger  15.5   8
## AMC Javelin       15.2   8
## Pontiac Firebird  19.2   8
## Ford Pantera L    15.8   8
## Ferrari Dino      19.7   6
## Maserati Bora     15.0   8
```

```
mtcars[ c(mtcars$cyl>=6 & mtcars$mpg>=15), c("mpg","cyl")]
```

```
##                    mpg cyl
## Mazda RX4         21.0   6
## Mazda RX4 Wag     21.0   6
## Hornet 4 Drive    21.4   6
## Hornet Sportabout 18.7   8
## Valiant           18.1   6
## Merc 280          19.2   6
## Merc 280C         17.8   6
## Merc 450SE        16.4   8
## Merc 450SL        17.3   8
## Merc 450SLC       15.2   8
## Dodge Challenger  15.5   8
## AMC Javelin       15.2   8
## Pontiac Firebird  19.2   8
## Ford Pantera L    15.8   8
## Ferrari Dino      19.7   6
## Maserati Bora     15.0   8
```

```
mtcars[(mtcars$gear>=3 & mtcars$cyl>=7)|(mtcars$gear>=3 & mtcars$mpg>=21), c("mpg","cyl","gear") ]
```

```
##                     mpg cyl gear
## Mazda RX4          21.0   6    4
## Mazda RX4 Wag      21.0   6    4
## Datsun 710         22.8   4    4
## Hornet 4 Drive     21.4   6    3
## Hornet Sportabout  18.7   8    3
## Duster 360         14.3   8    3
## Merc 240D          24.4   4    4
## Merc 230           22.8   4    4
## Merc 450SE         16.4   8    3
## Merc 450SL         17.3   8    3
## Merc 450SLC        15.2   8    3
## Cadillac Fleetwood 10.4   8    3
## Lincoln Continental 10.4   8    3
## Chrysler Imperial  14.7   8    3
```

```
## Fiat 128            32.4   4    4
## Honda Civic         30.4   4    4
## Toyota Corolla      33.9   4    4
## Toyota Corona       21.5   4    3
## Dodge Challenger    15.5   8    3
## AMC Javelin         15.2   8    3
## Camaro Z28          13.3   8    3
## Pontiac Firebird    19.2   8    3
## Fiat X1-9           27.3   4    4
## Porsche 914-2       26.0   4    5
## Lotus Europa        30.4   4    5
## Ford Pantera L      15.8   8    5
## Maserati Bora       15.0   8    5
## Volvo 142E          21.4   4    4
```

```
mtcars[(mtcars$gear>3 & mtcars$cyl>7)|(mtcars$gear>3 & mtcars$mpg>21), c("mpg","cyl","gear") ]
```

```
##                 mpg cyl gear
## Datsun 710     22.8   4    4
## Merc 240D      24.4   4    4
## Merc 230       22.8   4    4
## Fiat 128       32.4   4    4
## Honda Civic    30.4   4    4
## Toyota Corolla 33.9   4    4
## Fiat X1-9      27.3   4    4
## Porsche 914-2  26.0   4    5
## Lotus Europa   30.4   4    5
## Ford Pantera L 15.8   8    5
## Maserati Bora  15.0   8    5
## Volvo 142E     21.4   4    4
```

```
mtcars[(mtcars$gear>3) &( mtcars$cyl>7 |mtcars$mpg>21), c("mpg","cyl","gear") ]
```

```
##                 mpg cyl gear
## Datsun 710     22.8   4    4
## Merc 240D      24.4   4    4
## Merc 230       22.8   4    4
## Fiat 128       32.4   4    4
## Honda Civic    30.4   4    4
## Toyota Corolla 33.9   4    4
## Fiat X1-9      27.3   4    4
## Porsche 914-2  26.0   4    5
## Lotus Europa   30.4   4    5
## Ford Pantera L 15.8   8    5
## Maserati Bora  15.0   8    5
## Volvo 142E     21.4   4    4
```

```
mtcars[rownames(mtcars)=='Volvo 142E',]
```

```
##             mpg cyl disp  hp drat   wt qsec vs am gear carb
## Volvo 142E 21.4   4  121 109 4.11 2.78 18.6  1  1    4    2
```

```
library(ggplot2movies)
data(movies)
head(movies)
```

```
##                      title year length budget rating votes   r1   r2  r3
```

```
## 1                       $ 1971    121     NA    6.4   348  4.5  4.5 4.5
## 2        $1000 a Touchdown 1939     71     NA    6.0    20  0.0 14.5 4.5
## 3   $21 a Day Once a Month 1941      7     NA    8.2     5  0.0  0.0 0.0
## 4                 $40,000 1996     70     NA    8.2     6 14.5  0.0 0.0
## 5 $50,000 Climax Show, The 1975     71     NA    3.4    17 24.5  4.5 0.0
## 6                   $pent 2000     91     NA    4.3    45  4.5  4.5 4.5
##     r4   r5   r6   r7   r8   r9  r10 mpaa Action Animation Comedy Drama
## 1  4.5 14.5 24.5 24.5 14.5  4.5  4.5           0         0      1     1
## 2 24.5 14.5 14.5 14.5  4.5  4.5 14.5           0         0      1     0
## 3  0.0  0.0 24.5  0.0 44.5 24.5 24.5           0         1      0     0
## 4  0.0  0.0  0.0  0.0  0.0 34.5 45.5           0         0      1     0
## 5 14.5 14.5  4.5  0.0  0.0  0.0 24.5           0         0      0     0
## 6 14.5 14.5 14.5  4.5  4.5 14.5 14.5           0         0      0     1
##   Documentary Romance Short
## 1           0       0     0
## 2           0       0     0
## 3           0       0     1
## 4           0       0     0
## 5           0       0     0
## 6           0       0     0
```

```
colnames(movies)
```

```
##  [1] "title"       "year"        "length"      "budget"      "rating"
##  [6] "votes"       "r1"          "r2"          "r3"          "r4"
## [11] "r5"          "r6"          "r7"          "r8"          "r9"
## [16] "r10"         "mpaa"        "Action"      "Animation"   "Comedy"
## [21] "Drama"       "Documentary" "Romance"     "Short"
```

```
# movies[movies$title]
head(movies[grep("skies", movies$title, ignore.case = T), c("title","year","rating")])
```

```
##                                title year rating
## 38             'Neath Canadian Skies 1946    5.4
## 39           'Neath the Arizona Skies 1934    4.6
## 853  Ace Eli and Rodger of the Skies 1973    5.7
## 6512              Blue Montana Skies 1939    5.8
## 6527                     Blue Skies 1946    6.3
## 6528               Blue Skies Again 1983    4.9
```

grep을 이용하여 조건에 맞는 문자열 추출하기

```
pattern="^[Ss]ummer .*?"
ndx <- grep(pattern, movies$title)
ndx
```

```
##  [1] 49826 49827 49828 49829 49830 49831 49832 49833 49834 49835 49836
## [12] 49837 49838 49839 49840 49841 49842 49843 49844 49845 49846 49847
## [23] 49848 49849 49850 49851 49852 49853 49854 49855 49856 49857 49858
## [34] 49859 49860 49861
```

```
head(movies[ndx,"title"])
```

```
## [1] "Summer Blues"          "Summer Camp"           "Summer Camp Girls"
## [4] "Summer Camp Nightmare" "Summer Catch"          "Summer City"
```

```
grep("[Ss]ummer", movies$title)
```

```
##   [1]   147   327  1891  1896  2198  2449  5679  5680  5681  9193  9803
##  [12] 10607 11276 11458 12385 12516 15506 16015 16016 16215 18049 19098
##  [23] 19099 19302 21301 22175 22272 22765 23874 23875 24071 24087 24475
##  [34] 24548 25147 25285 25286 25287 25288 26279 28972 29115 29116 29117
##  [45] 29191 30569 32872 33481 33482 33483 33484 33485 33486 33487 33488
##  [56] 35355 37082 37853 39339 39941 39958 45739 47115 47499 49211 49764
##  [67] 49825 49826 49827 49828 49829 49830 49831 49832 49833 49834 49835
##  [78] 49836 49837 49838 49839 49840 49841 49842 49843 49844 49845 49846
##  [89] 49847 49848 49849 49850 49851 49852 49853 49854 49855 49856 49857
## [100] 49858 49859 49860 49861 49862 49863 49864 49865 49866 49867 49868
## [111] 49869 49870 49871 50360 51484 56102 56513 56871
```

```
length(grep("Summer", movies$title))
```

```
## [1] 109
```

```
grep("[Ss]ummer", movies$title)
```

```
##   [1]   147   327  1891  1896  2198  2449  5679  5680  5681  9193  9803
##  [12] 10607 11276 11458 12385 12516 15506 16015 16016 16215 18049 19098
##  [23] 19099 19302 21301 22175 22272 22765 23874 23875 24071 24087 24475
##  [34] 24548 25147 25285 25286 25287 25288 26279 28972 29115 29116 29117
##  [45] 29191 30569 32872 33481 33482 33483 33484 33485 33486 33487 33488
##  [56] 35355 37082 37853 39339 39941 39958 45739 47115 47499 49211 49764
##  [67] 49825 49826 49827 49828 49829 49830 49831 49832 49833 49834 49835
##  [78] 49836 49837 49838 49839 49840 49841 49842 49843 49844 49845 49846
##  [89] 49847 49848 49849 49850 49851 49852 49853 49854 49855 49856 49857
## [100] 49858 49859 49860 49861 49862 49863 49864 49865 49866 49867 49868
## [111] 49869 49870 49871 50360 51484 56102 56513 56871
```

```
length(grep("[Ss]ummer", movies$title))
```

```
## [1] 118
```

```
length(grep("summer", movies$title))
```

```
## [1] 9
```

```
c <- 1:10
d <- 1:5
d[c(1,3)]
```

```
## [1] 1 3
```

```
c[c(2,3)]
```

```
## [1] 2 3
```

```
c[2:3]
```

```
## [1] 2 3
```

```
c[c(3,2)]
```

```
## [1] 3 2
```

```
c[c>5]
```

```
## [1]  6  7  8  9 10
```

```
c>5
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
c[c>5 & c<10]
```

```
## [1] 6 7 8 9
```

```
c[c>5 | c<10]
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

TRUE FALSE 조건을 더하면, OR 조건문과 같은 결과를 얻을 있다.

```
#              .
c[as.logical((c>8 ) + (c<3))]
```

```
## [1]  1  2  9 10
```

```
c[(c>8 )| (c<3)]
```

```
## [1]  1  2  9 10
```

```
#              . as.logcal                .
c[as.logical((c>8 ) + (c<3))]
```

```
## [1]  1  2  9 10
```

```
c[(c>8 ) + (c<3)] # c[c(1,1,0,0,0,0,0,0,1,1)]
```

```
## [1] 1 1 1 1
```

인덱싱 ; 벡터에 이름 붙이기

```
years <- c(1960, 1964, 1976, 1994)

names(years)
```

```
## NULL
```

```
names(years) <- c("Kennedy","Johnson","Carter","Clinton")
```

```
years["Carter"]
```

```
## Carter
##   1976
```

```
years[3]
```

```
## Carter
##   1976
```

자료형 데이터 구조변환하기

```
as.numeric("3.14")
```

```
## [1] 3.14
```

```
as.integer(3.14)
```

```
## [1] 3
```

```
# as.numeric("foo")
as.character(101)
```

```
## [1] "101"
```

```
as.numeric(101)
```

```
## [1] 101
```

```
as.numeric(FALSE)
```

```
## [1] 0
```

```
as.numeric(F)
```

```
## [1] 0
```

```
Sys.Date()
```

```
## [1] "2018-02-01"
```

```
as.Date("2013-08-13")
```

```
## [1] "2013-08-13"
```

```
as.Date("2018.3.1", format="%Y.%m.%d")
```

```
## [1] "2018-03-01"
```

날짜를 문자열로 변환

```
as.Date("08/13/2013", format="%m/%d/%Y")
```

```
## [1] "2013-08-13"
```

```
format(Sys.Date())
```

```
## [1] "2018-02-01"
```

```
as.character(Sys.Date())
```

```
## [1] "2018-02-01"
```

```
format(Sys.Date(), format="%d/%m/%Y")
```

```
## [1] "01/02/2018"
```

```
format(Sys.Date(), "%a") #
```

```
## [1] " "
```

```
format(Sys.Date(), "%A") #
```

```
## [1] "  "
```

```
format(Sys.Date(), "%b") #
```

```
## [1] "2"
```

```
format(Sys.Date(), "%B") #      ..
```

```
## [1] "2 "
```

```
format(Sys.Date(), "%d") #
```

```
## [1] "01"
```

```
format(Sys.Date(), "%D") #
```

```
## [1] "02/01/18"
```

```
format(Sys.Date(), "%m") #
```

```
## [1] "02"
```

```
format(Sys.Date(), "%M")
```

```
## [1] "00"
```

```
format(Sys.Date(), "%h")
```

```
## [1] "2"
```

```
format(Sys.Date(), "%H")
```

```
## [1] "00"
```

```
format(Sys.Date(), "%y") #   2
```

```
## [1] "18"
```

```
format(Sys.Date(), "%Y") # 4
```

```
## [1] "2018"
```

Missing data

```
a <- 0/0
a #
```

```
## [1] NaN
```

무한대로 나가는 값

```
is.nan(a)
```

```
## [1] TRUE
```

```
b <- log(0)
b
```

```
## [1] -Inf
```

```
is.finite(b)
```

```
## [1] FALSE
```

```
c <- c(0:4, NA)
c
```

```
## [1]  0  1  2  3  4 NA
```

```
is.na(c)
```

```
## [1] FALSE FALSE FALSE FALSE FALSE  TRUE
```

벡터의 기본연산 (평균, 표준편차 등 기술통계량의 계산)

```
x<- c(0,1,1,2,3,5,8,13,21,34)
y <- log(x+1)
x
```

```
##  [1]  0  1  1  2  3  5  8 13 21 34
```

```
y
```

```
##  [1] 0.0000000 0.6931472 0.6931472 1.0986123 1.3862944 1.7917595 2.1972246
##  [8] 2.6390573 3.0910425 3.5553481
```

```
mean(x)
```

```
## [1] 8.8
```

```
median(x)
```

```
## [1] 4
```

```
sd(x)
```

```
## [1] 11.03328
```

```
var(x)
```

```
## [1] 121.7333
```

```
cor(x,y)
```

```
## [1] 0.9068053
```