



[실습3] List, Factor , DataFrame



List



- 값, 벡터, 행렬 등을 하나의 데이터 집합으로 관리하는 구조
- (키, 값) 형태의 데이터를 담는 연관 배열(associative array) 구조
- 리스트의 각 성분은 리스트 형태로 저장
- list()로 생성

```
>(x= list(1:5, c(T,F,T), "good day", matrix(1:6, ncol=2 )))
```

```
[[1]]
```

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

```
[[2]]
```

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

```
[[3]]
```

```
[1] "good day"
```

```
[[4]]
```

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

```
[1,]   1   4
```

```
[2,]   2   5
```

```
[3,]   3   6
```

```
>(y= list(name="hong", height=179, weight=77, score=c(100,50,80)))
```

```
$name
```

```
[1] "hong"
```

```
$height
```

```
[1] 179
```

```
$weight
```

```
[1] 77
```

```
$score
```

```
[1] 100 50 80
```

```
> y[1]    #1성분 리스트 접근
```

```
$name
```

```
[1] "hong"
```

```
>y$name    #name 성분 접근
```

```
[1] "hong"
```

```
>y$age = 20; #age 추가
```

List 요소접근



```
> (x = list(1:5, c(T,F,T), "good day"))
```

```
[[1]]
```

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

```
[[2]]
```

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

```
[[3]]
```

```
[1] "good day"
```

```
> #리스트 요소로 접근되어 추출
```

```
> x[[1]] # 1성분의 요소 추출
```

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

```
> x[[1]][2] # 1성분요소의 2요소 추출
```

```
[1] 2
```

```
>
```

```
> #리스트의 성분 서브리스트 추출
```

```
> x[1] # 1성분 추출
```

```
[[1]]
```

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

```
> x[1:2] # 1,2성분 추출
```

```
[[1]]
```

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

```
[[2]]
```

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

```
> x[c(T,F,T)] #1,3 성분 추출
```

```
[[1]]
```

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

```
[[2]]
```

```
[1] "good day"
```

```
> x[1] = list(5:9) #1성분 변경, list로 치환하여 변경
```

```
> x
```

```
[[1]]
```

```
[1] 5 6 7 8 9
```

```
[[2]]
```

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

```
[[3]]
```

```
[1] "good day"
```

```
>x[4:5] = list ("happy day", c(10,20,30)) #4,5 성분  
에 리스트 추가
```

List 결합, 분리



```
> #list 결합
> (newlist = c(x,y))
[[1]]
[1] 1 2 3 4 5

[[2]]
[1] TRUE FALSE TRUE

[[3]]
[1] "good day"

[[4]]
      [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6

$name
[1] "hong"

$height
[1] 179

$weight
[1] 77

$score
[1] 100 50 80
```

```
> #list 를 벡터로 분리
> unlist(newlist)

"1"      "2"      "3"      "4"      "5"

"TRUE"    "FALSE"    "TRUE" "good day"    "1"

"2"      "3"      "4"      "5"      "6"
name     height  weight  score1  score2
"hong"   "179"    "77"    "100"   "50"
score3
"80"
```



Factor

- 범주형(Categorical) 변수를 위한 데이터 구조
- 벡터의 요소를 그룹화하여 범주형 벡터(인자벡터)로 변환
- 생성 방법

순서없는 범주형 벡터: `factor(vector)`

순서있는 범주형 벡터: `ordered(vector) //== factor(vector, ordered=TRUE)`

```
> bt = c("A","A","B","AB","O","B") #6개의 혈액형 벡터
> (bloodtype = factor(bt) ) #bt요소를 범주화
[1] A  A  B  AB O  B
Levels: A AB B O
> levels(bloodtype) #그룹 레벨 확인
[1] "A" "AB" "B" "O"
> str(bloodtype) # bloodtype의 내부구조 확인
Factor w/ 4 levels "A","AB","B","O": 1 1 3 2 4 3
> table(bloodtype) #각 범주에 대한 개수 집계
bloodtype
 A AB  B  O
 2  1  2  1
> (obloodtype = ordered(bt)) #bt요소를 순서있는 factor로 그룹화
[1] A  A  B  AB O  B
Levels: A < AB < B < O
> str(obloodtype) # bloodtype의 내부구조 확인
Ord.factor w/ 4 levels "A"<"AB"<"B"<"O": 1 1 3 2 4 3
```

DataFrame



- 다양한 변수, 관측치(observations), 범주 등을 하나의 집합으로 표현하기 위한 구조

- 행, 열에 반드시 라벨을 가짐 (라벨을 이용하여 데이터 조작이 용이)
- 벡터나 행렬을 사용하여 데이터프레임 생성

```
d1 = data.frame(열이름 1=벡터1, ...)
```

```
d2 = data.frame(행렬)
```

```
names(d2) = c("열이름1", ...);
```

```
#dataframe , vector로 생성
> n=c("hong","min","kim") #성명
> h=c(172.6,155.7,180.9) #키
> a = c(20,50,19)          #나이
> (d1 = data.frame(name=n,height=h,age=a))
  name height age
1 hong  172.6  20
2 min   155.7  50
3 kim   180.9  19
```

```
#dataframe matrix로 생성
#같은 데이터형으로 구성된 번호, 키, 나이 행렬
> dv=c(1,173,20,2,156,50,3,181,19)
#행렬로 데이터프레임 생성
> m =matrix(dv, nrow=3, byrow=T)
> (d2= data.frame(m))
  X1 X2 X3
1  1 173 20
2  2 156 50
3  3 181 19
> cn = c("id", "height", "age")
> names(d2) = cn;      # 열이름 지정
> d2
  id height age
1  1   173  20
2  2   156  50
3  3   181  19
```

DataFrame



문자열 벡터를 데이터프레임의 항목으로 포함시 범주형으로 변환
되는 것을 막기 위해 l() 이용

```
>n=c("hong","min","kim", "jin") #성명
>h=c(172.6,155.7,180.9, 170.5) #키
>a = c(20,50,19,23)          #나이
>d1 = data.frame(name=n,height=h,age=a)
> d1$name #d1 name접근, factor로 처리됨
[1] hong min  kim  jin
Levels: hong jin kim min
>d3 = data.frame(name=l(n),height=h,age=a) #l()사용
> d3$name #d3 name접근, vector로 처리됨
[1] "hong" "min" "kim" "jin"
```

DataFrame



- 데이터 프레임 접근

:행, 열의 인덱스나, 열의 이름으로 접근

d\$n : d 데이터프레임의 n 열이름 모든 행 확인

d[1,] : 1번 줄에 해당하는 모든 열 확인

d[,c(1,3)] : 1,3열만 확인

d[,-2] : 2열 제외 확인

head() : 데이터의 앞부분 3행까지 확인

tail() : 데이터의 뒤부분 3행까지 확인

```
> #데이터프레임 접근
> d1$name #d1 name접근, factor로 처리됨
[1] hong min kim jin
Levels: hong jin kim min
> d1[1,] #d1 1행
  name height age
1 hong  172.6  20
> d1[,2] #d1 2열
[1] 172.6 155.7 180.9 170.5
> d1[1,2] #d1 1행2열
[1] 172.6
> d2$id #d2 id접근
[1] 1 2 3
> d2[2,] #d2 2행
  id height age
2  2   156  50
> d2[1,3] #d2 1행3열
[1] 20
> head(d1, n=3) #d1의 상위3개, default=6
  name height age
1 hong  172.6  20
2 min  155.7  50
3 kim  180.9  19
> tail(d1, n=2) #d1의 하위2개
  name height age
3 kim  180.9  19
4 jin  170.5  23
```


DataFrame



- 조건에 따른 데이터 프레임 접근
데이터프레임[조건식] : 조건식이 참인 데이터를 접근
- 조건에 따른 데이터 추출
subset(데이터프레임, 조건, 대상)

```
#데이터 조건 접근
> d1[d1$name=="kim",]    #name이 kim인 요소접근
  name height age
3  kim  180.9  19
> d1[d1$height <= 170,]  #height가 170이하 요소 접근
  name height age
2  min  155.7  50
> d1[d1$age >= 20 & d1$age < 30,]  #20대 요소 접근
  name height age
1 hong  172.6  20
4  jin  170.5  23
> d1[d1$height >= 180 & d1$age <=20,]  #키가 180이상이면서 나이가 20이하인 요소 접근
  name height age
3  kim  180.9  19
> subset(d1, height <= 170)  #d1의 height가 170이하 요소 추출
  name height age
2  min  155.7  50
> subset(d1, height <= 170, c(name, height))  #d1의 height가 170이하인 성명, 키만 추출
  name height
2  min  155.7
```

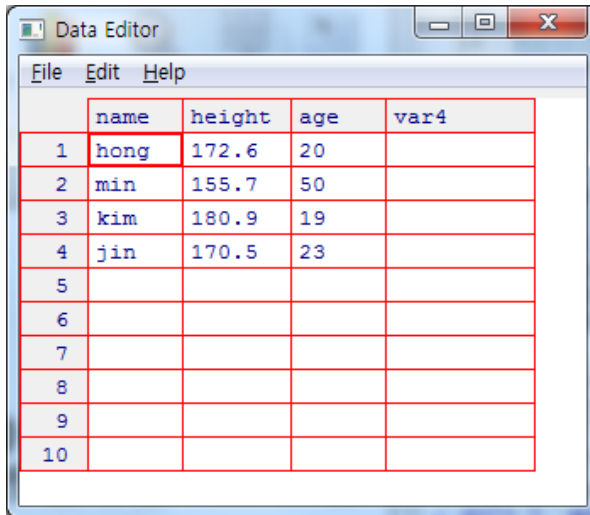


DataFrame

- 데이터 프레임 수정
(1) 데이터 요소 접근으로 수정

데이터프레임명\$열이름 = 벡터
데이터프레임명[1,2] = 값

(2) edit() : 데이터 편집기로 데이터 수정



	name	height	age	var4
1	hong	172.6	20	
2	min	155.7	50	
3	kim	180.9	19	
4	jin	170.5	23	
5				
6				
7				
8				
9				
10				

```
> #데이터프레임 수정
> d3[1,] = c("song", 160.5, 33)
> d3
  name height age
1 song  160.5  33
2 min   155.7  50
3 kim   180.9  19
4 jin   170.5  23
> d3$height = c(166, 177, 188, 178)
> d3
  name height age
1 song   166  33
2 min   177  50
3 kim   188  19
4 jin   178  23
> d3[1,3] = 22
> d3
  name height age
1 song   166  22
2 min   177  50
3 kim   188  19
4  jin   178  23

> edit(d1) #d1 편집
```

DataFrame



- 데이터 프레임에 요소 추가
:새열이름에 벡터 추가

데이터프레임명\$새열이름 = 벡터

- 데이터 프레임 요소 삭제

데이터프레임명\$열이름 = NULL

```
> #데이터프레임 요소 추가
w = c(55.6, 70.4, 54.6, 80.4)
> d1$weight = w
> d1
  name height age weight
1 hong  172.6  20   55.6
2 min  155.7  50   70.4
3 kim  180.9  19   54.6
4 jin  170.5  23   80.4
> # 데이터프레임 요소 삭제
> d1$weight = NULL
> d1
  name height age
1 hong  172.6  20
2 min  155.7  50
3 kim  180.9  19
4  jin  170.5  23
```

DataFrame



- 데이터 프레임 결합

`rbind(데이터프레임1,데이터프레임2)` : 행으로 결합, 열이름이 같은 경우 가능

`cbind(데이터프레임1,데이터프레임2)` : 열로 결합, 같은 행인 경우 가능

`dataframe(데이터프레임1,데이터프레임2)` : 열로 결합, 열이름 같은 경우 변경

`merge(데이터프레임1,데이터프레임2)` : 열로 결합, 같은 키변수(첫번째 항목)로 결합

```
> (d4= rbind(d1,d3)) #행으로 결합
  name height age
1 hong  172.6  20
2 min  155.7  50
3 kim  180.9  19
4 jin  170.5  23
5 hong  172.6  20
6 min  155.7  50
7 kim  180.9  19
8 jin  170.5  23
> (d5= cbind(d1,d3)) #열로 결합
  name height age name height age
1 hong  172.6  20 hong  172.6  20
2 min  155.7  50 min  155.7  50
3 kim  180.9  19 kim  180.9  19
4 jin  170.5  23 jin  170.5  23
```

```
>(df= data.frame(d1[, c(1,3)],d3[,
c(1,2)]))
  name age name.1 height
1 hong  20   hong  172.6
2 min  50   min  155.7
3 kim  19   kim  180.9
4 jin  23   jin  170.5
> (mg= merge(d1[, c(1,3)],d3[,
c(1,2)]))
  name age height
1 hong  20  172.6
2 jin  23  170.5
3 kim  19  180.9
```

DataFrame



- 데이터 프레임 계산
 - lapply() : 벡터, 데이터프레임, 리스트의 요소에 대한 함수를 적용하여 결과를 리스트로 반환
 - sapply() : 벡터, 데이터프레임, 리스트의 요소에 대한 함수를 적용하여 결과를 벡터나 행렬로 반환

```
> #lapply(), sapply()
> (mean1 = lapply(d1[, -1], mean)) #1열을 제외한 열의 평균
$height
[1] 169.925

$age
[1] 28
> (mean2 = sapply(d1[, -1], mean))
height age
169.925 28.000
> (mean3 = sapply(d1[d1$age > 20, -1], mean)) #조건이 참인 값만 평균 계산
height age
163.1 36.5
```

연습문제



1. 아래 그림을 참고로 5명의 두 개의 데이터프레임을 생성하고, 결합하여 다음 조건을 처리

(1) ID, password, name은 범주형이 아님

(2) 나이가 20미만만 추출

(3) Mileage 가 100~200 사이만 추출

(4) 나이와 마일리지의 평균

(5) 혈액형과 성별 추가

ID	password	name

ID	age	mileage

ID	password	name	age	mileage