# [실습2]R 기초

- ✓ 조건문
- ✓ 반복문
- ✓ 배열

# 조건문

- If~else 문
  - 조건식의 결과가  참, 거짓에 따라 나누어서 처리하기 위한 명령

```
형식 :
if ( 조건식 ) {
   <조건식이 참일 경우 실행되는 식>
} else {
   <조건식이 거짓일 경우 실행되는 식>
}
```

#예 1 : x가 0이상이면 “양수”를 반환하여 s2에 대입, x가 0보다 작으면 “음수”를  반환하여 s2에 대입

```
> s2 = {   if (x >= 0){
              "양수"
            }else {
              "음수“
           }
       }
```

#예: 2 x가 0이상이면 “양수”를 print 하고 1증가, 아니면 “음수”를 print하고 부호변환

```
> if (x >= 0){
   print("양수")
   x = x + 1
 }else {
   print("음수")
   x = x * (-1)
 }
```

# 조건함수

- ifelse()
  - 주어진 조건식이 참일 경우와  거짓일 경우 반환 값을 다르게 받기 위한 함수

형식 :
ifelse ( 조건식 , 참일 경우 반환 값, 거짓일 경우 반환 값)

```
#x가 0이상이면 “양수”를 반환하여 s에 대입, x가  0보다 작으면 “음수”를  반환하여 s에 대입
> x= 10
> s = ifelse(x >= 0, "양수", "음수")
```

# 반복문

- 여러 식을 반복하여 처리하는 구문

<반복문 형식>

for(변수 in 벡터 ) 반복식

while(조건) 반복식

repeat 반복식

break 반복 종료

```
#for 예
# 벡터 안의 값을 하나씩 꺼내어  모든 요소가 추출될 때까지 반복
x=1:5
for (i  in  x){    #x벡터의 요소를 i에 대입 반복
  print(i)
}
```

```
#repeat 예
#1에서 5까지 1씩 증가하면서 반복하여 출력
i=1
repeat {    #무조건 반복
   print(i)
   if(i == 5) break;   #i가 5이면 반복 종료
   i=i+1     #1증가
}
```

```
#while 예
#1에서 5까지 1씩 증가하면서 반복하여 출력
j=1
while(j <= 5) {    //j가 5이하이면 반복
   print(j)
   j=j+1              //1증가
}
```

# 반복문

```
# 2- 10사이의 짝수의 합
x=seq(2,10,2)
s = 0
for (i in x){
    s = s + i
}

#구구단
for (i in 2:9){        #2단-9단 반복
   for (j in 1:9){     #1-9 숫자를 반복
         print(i * j)        #곱한 결과 출력
  }
}
```

# 연습문제(1)

1) x에 값을 넣어 100-70이면 "x는 합격"을 출력 아니면 "x는 불합격"을 출력

(if-else 문, sprintf 사용)

2) 1에서 100사이의 5의 배수의 합을 구하여 출력

(for문, cat 사용 )

<출력 예시>

sum =  1050

# R 배열

- ✓ Vector
- ✓ Matrix
- ✓ Array

# Vector 생성

- 한가지 데이터 형에 대한 집합 구조(배열)
- 생성 방법
  - 목록을 나열하여 생성 : c(목록 나열)
  - 연속값 으로 생성: 시작값:최종값
  - 시작 값에서 증가치 만큼 최종 값으로 생성

    : seq(시작 값, 최종 값,증가치)
    - 특정 값들이 횟수만큼 반복된 형태로 생성 :
      - rep(벡터,횟수) : 벡터를 횟수만큼 반복
      - rep(벡터, each=횟수) : 벡터의 요소 각각을 회수만큼 반복

```
> #벡터생성방법
> c(10,20,30,40,50)  #요소나열로 벡터 생성
[1] 10 20 30 40 50
> c(T,T,F,F,T)
[1]  TRUE  TRUE FALSE FALSE  TRUE
> c('a','b','c')
[1] "a" "b" "c"
> 1:5              #1씩 증가하는 연속값
[1] 1 2 3 4 5
> 10:20
 [1] 10 11 12 13 14 15 16 17 18 19 20
> seq(1,5)
[1] 1 2 3 4 5
> seq(1,10, 2)      #1부터 10까지 2씩 증가
[1] 1 3 5 7 9
> rep(1:3, 2)       #1-3까지 2회 반복
[1] 1 2 3 1 2 3
> rep(c('a','b','c'), 3)  #'a''b''c' 3회 반복
[1] "a" "b" "c" "a" "b" "c" "a" "b" "c"
>rep(1:2, each=3)  #1:2 각각 3회 반복
[1] 1 1 1 2 2 2
> (a = 1:5)  # a벡터 생성, ()로 결과 확인
[1] 1 2 3 4 5
```

# 벡터 요소

- 벡터요소 : 벡터 안의 각 수치
- 벡터인덱스 : 벡터의 요소번호
- 벡터내의 요소접근
  - 단일요소 접근 : 벡터명[인덱스]
  - 특정 요소 제외 접근 : 벡터명[-인덱스]
  - 여러 요소접근
    - 벡터명[벡터]
    - 벡터명[시작인덱스:끝인덱스]
  - 벡터의 요소 변경
    - 벡터명[인덱스] = 값
- 벡터 요소 결합
  - 벡터에 값 추가 : c(벡터, 값)
  - 벡터에 벡터 결합 : c(벡터, 벡터)
- 중복요소 제거
  - unique(벡터)

```
#벡터 생성, x에 저장
>x <- c('a', 'b', 'c')
#벡터요소 접근
> #벡터요소 접근
> x [1]
[1] "a"
> x [3]
[1] "c"
> #벡터요소 제외 접근
> x[ -1]
[1] "b" "c"
> x[ -2]
[1] "a" "c"
> #여러 요소 접근
> x[c(1, 2)]
[1] "a" "b"
> x[c(1, 3)]
[1] "a" "c"
> x[1:2]
[1] "a" "b"
> x[1] = "e"
> #벡터 결합
>(y = c(x, 'd'))    #단일값 결합
>(z = c(x, c('d','e','f')))   #벡터 결합
#중복제거
>unique(z)
[1] "e" "b" "c" "d" "f"
```

# 벡터 연산

- 여러 값을 한꺼번에 연산
- 벡터요소끼리 연산처리

```
#벡터 산술 연산 예
> (a = 1: 5)
[1] 1 2 3 4 5
> (b = 11:15)
[1] 11 12 13 14 15
> (a1 = a - 1 )  #a벡터의 각 요소에서 1을 뺀 연산
[1] 0 1 2 3 4
> (sum = a + b)  #a벡터와 b벡터의 각 요소 연산
[1] 12 14 16 18 20
> (sub = a - b)
[1] -10 -10 -10 -10 -10
> (mul = a * b)
[1] 11 24 39 56 75
> (div = b / a)
[1] 11.000000  6.000000  4.333333  3.500000  3.000000
> round(div)     #div벡터의 각 요소 반올림
[1] 11  6  4  4  3
```

# 벡터 연산

```
#벡터 비교연산
(a > 4)              #a의 요소가 4보다 큰 값인가
(eq = (a == b))      #a와 b의 요소가 같은가
(neq = (a != b))     #a와 b의?요소가 다른가
(l = (a < b))        #a의 요소가 b의 요소보다 작은가
(g = (a > b))        #a의 요소가 b의 요소보다 큰가

#벡터 집합연산
(u = union(a,b))     #합집합
(intersect(u,b))       #교집합
(d=setdiff(u,b))     #차집합
setequal(a,d) ?      #같은 집합인가

#요소 k는 집합 a에 속하는가
k=3
is.element(k, a)
k %in% a
```

# 벡터함수

길이,최대, 최소 : length(), max(), min()

평균, 분산, 표준편차 : mean(), var(), sd()

중앙값, 정렬, 순위 : median(), sort(), rank()

```
#벡터함수 예
# 5명의 점수
> score = c(60, 100, 50, 80, 89)
> length(score)  # 벡터의 길이
[1] 5
> max(score)     #최대값
[1] 100
> min(score)     #최소값
[1] 50
> mean(score)    #평균
[1] 75.8
> sd(score)      #표준편차
[1] 20.57183
> median(score)  #중앙값
[1] 80
> sort(score)    #요소정렬
[1]  50  60  80  89 100
> rank(score)    #각 요소의 순위
[1] 2 5 1 3 4
```

# 연습문제(2)

다음 문제를 R스크립트로 작성하여 결과 확인

(1)짝수, 홀수 처리

- 1-10사이의 홀수 벡터(odd), 짝수벡터(even) 생성
- 홀수, 짝수 벡터 결합하여 total 벡터 생성
- total 벡터를 정렬하여 stotal생성
- stotal 벡터에서 짝수를  제외

(2)5명의 BMI 처리

- 5명의 키, 몸무게를 벡터로 생성
- 키와 몸무게의 평균, 표준편차, 최대, 최소,중앙값  계산
- 5명의 신체질량지수 (body mass index, BMI) 계산

  BMI = 체중(kg) ÷ {신장(m) $^2$}

# 행렬(Matrix) 생성

- 2차원 형태의 행과 열로 구성된 집합 구조
- 생성 형식

matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)

data : vector, row:행개수, ncol:열개수, byrow:행우선, dimnames:행렬이름

```
#1, 2, 3, 4, 5, 6, 7, 8, 9로 구성된 3행 3열의 열 우선 행렬 생성
> matrix (c(1, 2, 3, 4, 5, 6, 7, 8, 9) , nrow =3)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> matrix (c(1, 2, 3, 4, 5, 6, 7, 8, 9) , ncol =3)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
#행 우선 생성
> matrix (c(1, 2, 3, 4, 5, 6, 7, 8, 9) , nrow =3, byrow =T)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```
#값이 없는(NA)  3행 1열 행렬
>matrix (, nrow =3)
 [,1]
 [1,]   NA
 [2,]   NA
 [3,]   NA
#제로 행렬 (4X3)
>matrix(0, nrow=4, ncol=3)
 [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
[3,]    0    0    0
[4,]    0    0    0
#1:4값 4X3 열 우선 행렬, 값 반복
>matrix(1:4,nrow=4, ncol=3)
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    2    2    2
[3,]    3    3    3
[4,]    4    4    4
```

# 행렬 요소

- 행렬요소 추출

  행렬명[행인덱스, 열인덱스]

  :인덱스 : 값, 벡터 가능

```
> #행렬의 각 요소는 '행렬이름[행인덱스, 열인덱스]'로 접근
> (x = matrix (c(1, 2, 3, 4, 5, 6, 7, 8, 9) , nrow =3, byrow =T))
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
> x[1,2]    #1행 2열 요소
[1] 2
> x[1:2,2]  #1-2행 2열 요소
[1] 2 5
> x[,2]     #모든 행의 2열 요소
[1] 2 5 8
> x[3,]     #3행 모든 열 요소
[1] 7 8 9
> x[3,2:3]  #3행 2-3열 요소
[1] 8 9
> x[-2,]    #2행 제외한 모든 3열 요소
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    7    8    9
> x[c(T,F,T), 2]  #1행, 3행에 대한 2열 요소
[1] 2 8
> x[c(1,3), 2]  #1행, 3행에 대한 2열 요소
[1] 2 8
>x[1:5] #행렬을 벡터로 변환하고 1-5번째 요소 추출
 [1] 1 4 7 2 5
```

# 행렬연산

- 행렬의 각 요소와  산술연산
- 행렬 곱 : %*%

```
> #행렬연산
> x+5     #x 요소에 5를 더한 결과
     [,1] [,2] [,3]
[1,]    6    7    8
[2,]    9   10   11
[3,]   12   13   14
> x*2     #x요소에 2를 곱한 결과
     [,1] [,2] [,3]
[1,]    2    4    6
[2,]    8   10   12
[3,]   14   16   18
> (y = x)   #x행렬로 y행렬 생성
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```
> x+y     #x와 y 행렬 각 요소 더하기
     [,1] [,2] [,3]
[1,]    2    4    6
[2,]    8   10   12
[3,]   14   16   18
> x-y     #x와 y 행렬 각 요소 빼기
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
[3,]    0    0    0
> x*y     #x와 y 행렬 각 요소 곱하기
     [,1] [,2] [,3]
[1,]    1    4    9
[2,]   16   25   36
[3,]   49   64   81
> x %*% y  #x와 y 행렬곱
     [,1] [,2] [,3]
[1,]   30   36   42
[2,]   66   81   96
[3,]  102  126  150
```

# 행렬결합

- 벡터나 행렬을 결합하여 새로운 행렬 생성
  - rbind() : 행단위로 결합
  - crbind() : 열단위로 결합

```
> (rx= rbind(1:3, 4:6, c(7,8,9)))  #rbind() : 행단위로 결합
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9

> (cx= cbind(1:3, 4:6, c(7,8,9))) #crbind() : 열단위로 결합
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

# 행렬함수

- 행렬의 차원 : dim(), ncol(), nrow()
- 전치행렬 : t()
- 대각행렬 , 대각 성분 : diag()

```
#행렬함수 예
>(x = matrix (1:9 , nrow =3, byrow =T))
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
> dim(x)   #x행렬의 크기
[1] 3 3
> nrow(x)  #x행렬의 행의 개수
[1] 3
> ncol(x)  #x행렬의 열의 개수
[1] 3
> t(x)  #x행렬의 전치행렬
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> diag(x)     #x행렬의 대각성분만 추출
[1] 1 5 9
> diag(x) = 4 #x행렬의 대각성분을 4로 설정
> x
     [,1] [,2] [,3]
[1,]    4    2    3
[2,]    4    4    6
[3,]    7    8    4
> diag(1, 5) #대각성분1인 단위행렬
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    1    0    0    0
[3,]    0    0    1    0    0
[4,]    0    0    0    1    0
[5,]    0    0    0    0    1
```

# 행렬 함수

- 행렬 속성에 이름을 부여 : rnames(), colnames()
- 행렬에 함수적용 : apply()

  형식 : apply(X, MARGIN, FUN, ...)

  X : 배열, MARGIN: 1(row)/2(column) FUN:함수

```
> #행렬 이름(label)
> #성적처리 예
> rnames = c("hong", "min","kim")   #행 이름 벡터
> cnames = c("kor","eng","mat") #열 이름 벡터
> (score = rbind(c(100,80,90),c(50,70,80),c(90,99,96)))  #점수벡터 생성
     [,1] [,2] [,3]
[1,]  100   80   90
[2,]   50   70   80
[3,]   90   99   96
> rownames(score) = rnames   #행 이름 설정
> colnames(score) = cnames   #열 이름 설정
> score
     kor eng mat
hong 100  80  90
min   50  70  80
kim   90  99  96
> score["hong","eng"]   #"hong'행의 "eng"열 요소
[1] 80
>
>rownames(score) = NULL   #행 이름 제거
>colnames(score) = NULL   #열 이름 제거
>dimnames(score) = NULL   #행열 이름 모두 제거
```

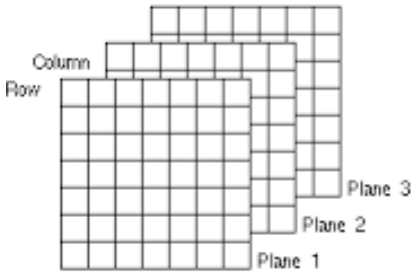
```
> #과목별 총점 평균 최대 최소
> (sum = apply(score, 2 ,sum))
kor eng mat
240 249 266
>(avg=apply(score, 2 ,mean))
> (max=apply(score, 2, max))
kor eng mat
100  99  96
> (min=apply(score, 2, min))
kor eng mat
 50  70  80
>
> #개인별 총점 평균 최대 최소
> (sum = apply(score, 1 ,sum))
hong  min  kim
 270  200  285
> avg=apply(score, 1 ,mean)
> (max=apply(score, 1, max))
hong  min  kim
 100   80   99
> (min=apply(score, 1, min))
hong  min  kim
  80   50   90
```

# 배열(Array)

- 배열(array)은 n차원 행렬
- 생성방법

  array(벡터, dim=c(행수,열수, 면수))
- 요소접근

  배열명[행인덱스,열인덱스,면인덱스]




```
> # 3X4 array
> matrix (1:12 , ncol =4)
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> array (1:12 , dim=c(3, 4))
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

```
> (x = array (1:12 , dim=c(2, 2, 3)))  #2X2X3 array
, , 1

     [,1] [,2]
[1,]    1    3
[2,]    2    4

, , 2

     [,1] [,2]
[1,]    5    7
[2,]    6    8

, , 3

     [,1] [,2]
[1,]    9   11
[2,]   10   12

> x[1,1,1]   #1행 1열 1면
[1] 1
> x[1,,]     #1행의 모든 열, 모든면
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    3    7   11
> x[,,1]     #1면의 모든 행열
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

# 연습문제(3)

1)5명의 키, 몸무게를 행렬로 생성

(1)행렬의 행이름, 열이름 설정 (행:성명, 열:키,몸무게)

(2) 키와 몸무게의 평균, 표준편차, 최대, 최소,중앙값  계산