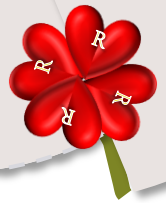
6.조건제어문

모든 프로그램 언어에 공통적으로 들어가는 문법 바로 조건제어문 이죠? r의 조건제어문 특징을 기억합니다.

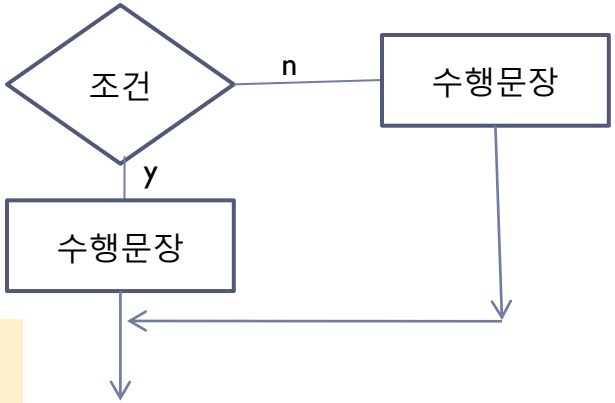


1. if

•조건에 따라 문장을 선택적으로 실행
•형식

```
if (조건식) {
참일 때 수행할 문장
}
else {
거짓일때 수행할 문장
}
```

조건이 거짓일때 실행할 문장이 있을경우 else 블록 코딩



간단한 if문 예)
a에 저장된 값이 짝수/홀수 인지 판별하여 출력하는 간단한 프로그램

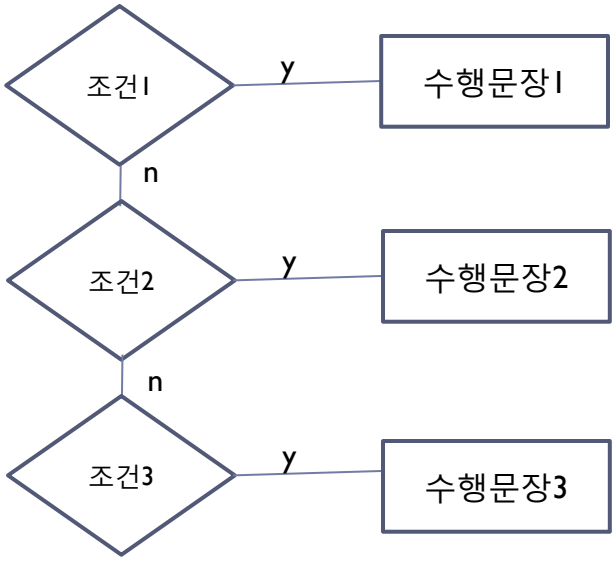
```
> a<-5
> if(a%%2==0){
+   result<-"짝수"
+   result
+ }else{
+    result<-"홀수"
+    result
+ }
[1] "홀수"
```

조건비교연산자	기능
> , <	크다, 작다
>= , <=	크거나 같다, 작거나 같다
==	같다(equal)
!=	같지않다

❑논리연산자

연산자	기능
&&	And
|	Or
!	not

```
> a<-75
> if(a%%3==0 && a%%5==0){
+ result<-"3의 배수이면서 5의 배수"
+ }else if(a%%3==0){
+ result<-"3의 배수"
+ }else if(a%%5==0){
+ result<-"5의 배수"
+ }
> result
[1] "3의 배수이면서 5의 배수"
>
```



2. ifelse 문

•다수의 데이터에 대한 조건을 한꺼번에 처리할 때 사용
•형식

ifelse (조건대상,참일때 실행문장,거짓일때 실행문장)

```
> a<-c(10,13,7,8,100)
> if(a%%2==0){
+ result<-"짝수"
+ }else{
+ result<-"홀수"
+ }
경고메시지(들):
In if (a%%2 == 0) { :
  length > 1 이라는 조건이 있고, 첫번째 요소만이 사용될 것입니다
> result
[1] "짝수"
```

a에 저장된 첫번째 요소만 조건식에 사용됩니다.

```
> a<-c(10,13,7,8,100)
> ifelse(a%%2==0,"짝수","홀수")
[1] "짝수" "홀수" "홀수" "짝수" "짝수"
```

a에 저장된 모든 데이터들에 대한 조건판별이 이뤄집니다. →정말 편하죠?

앞의 조건비교결과를 데이터 프레임타입으로 표현볼까요?

```
> a<-c(10,13,7,8,100)
> result<-c(ifelse(a%%2==0,"짝수","홀수"))
> df<-data.frame(a,result)
> df
     a result
1   10   짝수
2   13   홀수
3    7   홀수
4    8   짝수
5  100   짝수
>
```

조건비교결과를 벡터로 생성해요

두 벡터를 결합하여 프레임 생성하는거 아시죠?^^

```
> a<-c(10,13,22,15,9)
> result<-ifelse(a>mean(a),"평균초과",ifelse(a<mean(a),"평균미달","평균값"))
> df2<-data.frame(a,result)
> df2
   a    result
1 10  평균미달
2 13  평균미달
3 22  평균초과
4 15  평균초과
5  9  평균미달
```

ifelse(조건식,참,
ifelse(조건식,참,거짓))
→이렇게도 쓸 수 있어요.

2. 반복문 - for

•한 개 이상의 문 장을 특정횟수만큼 반복처리

형식

```
for (변수명 in 반복횟수){
    반복할 실행 문장
}
```

변수에 반복횟수를 할당하여 그 값에 도달할때까지 블록을 반복실행

for 기본예문)

```
> y<-0
> for(x in 1:5) {
+ y<-y+x
+ }
> y
[1] 15
```

1부터 5까지 값을 x에 할당하여 y에 x값을 누적하여 결과 출력

```
> y<-0
> for(x in 1:5) {
+ y<-y+x
+ cat("1부터 ", x, "까지 합=",y,"\n")
+ }
1부터  1 까지 합= 1
1부터  2 까지 합= 3
1부터  3 까지 합= 6
1부터  4 까지 합= 10
1부터  5 까지 합= 15
```

cat함수를 이용하여 누적과정을 그림처럼 표시할 수 있어요

for – vector 접근 예문)

```
> a<-c(4,25,8,33,17,20)
> for(x in 1:length(a)){
+ if(a[x]%%2==0)evensum<-evensum+a[x]
+ }
> evensum
[1] 32
```

3. 반복문 - while

•조건이 "참"인 동안 블록을 반복실행

형식

```
while (조건){
    반복할 실행 문장
}
```

조건이 참일 때 블록안의 문장 수행

while 기본예문1)

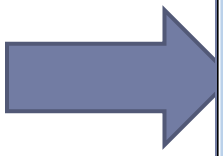
```
> i<-1
> isum<-0
> while(i<=10) {
+ isum<-isum+i
+ i<-i+1
+ }
> isum
[1] 55
```

i값이 10 이하 인 동안 블록을 반복수행

while 기본예문2)

```
> i<-1
> isum<-0
> while(i!=0) {
+ isum<-isum+i
+ cat("1부터 ",i,"까지 합=",isum,"\n")
+ if(isum>=500) break
+ i<-i+1
+ }
```

무한loop에서 특정 조건을 만족하면 loop를 벗어나는 프로그램



```
1부터  1 까지 합= 1
1부터  2 까지 합= 3
1부터  3 까지 합= 6
1부터  4 까지 합= 10
1부터  5 까지 합= 15
1부터  6 까지 합= 21
1부터  7 까지 합= 28
1부터  8 까지 합= 36
1부터  9 까지 합= 45
1부터  10 까지 합= 55
1부터  11 까지 합= 66
1부터  12 까지 합= 78
1부터  13 까지 합= 91
1부터  14 까지 합= 105
1부터  15 까지 합= 120
1부터  16 까지 합= 136
1부터  17 까지 합= 153
1부터  18 까지 합= 171
1부터  19 까지 합= 190
1부터  20 까지 합= 210
1부터  21 까지 합= 231
1부터  22 까지 합= 253
1부터  23 까지 합= 276
1부터  24 까지 합= 300
1부터  25 까지 합= 325
1부터  26 까지 합= 351
1부터  27 까지 합= 378
1부터  28 까지 합= 406
1부터  29 까지 합= 435
1부터  30 까지 합= 465
1부터  31 까지 합= 496
1부터  32 까지 합= 528
```

4. 반복문 - repeat

블록안의 문장을 반복실행

형식

```
repeat{
    반복할 실행 문장
}
```

반복영역안에서 if조건을 지정하여 해당 조건에 만족되는 경우 loop를 벗어나게끔 작업

벡터 평균을 계산하여 미달하는 첫 번째 원소값을 화면에 출력하는 프로그램

```
> a<-c(42,25,16,7,23,12,9)
> i<-1
> repeat{
+ if(a[i]<mean(a)) break
+ i<-i+1
+ }
> a[i]
[1] 16
```

반복 loop 벗어남