제어의 흐름

Jeon Jong-June

Monday, March 09, 2015

조건문

논리 연산

```
3>2
```

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

```
2>3
```

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

```
(3>2)&(2>3)
```

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

```
(3>2)|(2>3)
```

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

```
!(2>3)
```

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

```
3==2
```

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

```
3>=2
```

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

조건문

```
a <- 3
b = 0
if (a <2 )
  {
  b = 1
  }
b
```

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

```
a <- 3
if (a <2 )
  {
    b <- 1
  } else {
    b <- 0
  }
b
```

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

ifelse 문

```
x <- c(6:-4)
sqrt(x)  #- gives warning
```

```
## Warning in sqrt(x): NaNs produced

##  [1] 2.449490 2.236068 2.000000 1.732051 1.414214 1.000000 0.000000
##  [8]      NaN      NaN      NaN      NaN
```

```
sqrt(ifelse(x >= 0, x, NA))
```

```
##  [1] 2.449490 2.236068 2.000000 1.732051 1.414214 1.000000 0.000000
##  [8]       NA       NA       NA       NA
```

반복문

```
x <- 0
for (i in 1:10)
  {
  x <- x + 1
  }
x
```

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

```
x <- 0
v <- 1:10
for (i in v)
  {
  x <- x + 1
  }
x
```

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

```r
x <- 0
v <- seq(0,10, by = 2)
for (i in v)
  {
  x <- x + 1
  }
x
```

```
## [1] 6
```

error 구문의 처리

try 함수의 결과값을 class를 확인한다.

```r
# define a funtion
a.fun = function(x)
  {
    v = solve(x)
    return (v + 1 )
    }

v.vec = sample(c(0,1), 100, replace = T)

result = list()

for ( i in 1:length(v.vec))
  {
  # try function
  a = try(a.fun(v.vec[i]), silent = T)
  # check the return of try
  if (class(a)=='try-error')
    {
    result[[i]] = NA
    } else {
    result[[i]] = a
    }
  }
```