Non Numeric Values

In this we are generally focusing on working with non-numeric data types, an essential skill for effective statistical programming in R. These data types play a crucial role when dealing with real-world data sets and become especially important when working with more complex R programming.The chapter covers three important non-numeric data types:

1. Logical: representing TRUE/FALSE values.
2. Characters: representing textual data (strings).
3. Factors: representing categorical data.

Logical Values

Logical values, often referred to as logicals, are fundamental to programming. They are based on the principle that a logical-valued object can be either TRUE or FALSE. These values can represent concepts like yes/no, one/zero, or satisfied/not satisfied, and they have numerous applications in programming.

Reserved Values: In R, the terms TRUE and FALSE are reserved specifically for representing logical values. This means you cannot use these terms to name objects or assign them different values.

Unreserved Equivalents:R also allows you to use T and F as shorthand for TRUE and FALSE.However, T and F are not reserved words in R. This means a user could potentially redefine them, leading to unexpected behavior. For this reason, it's generally recommended to use TRUE and FALSE for clarity and to avoid potential errors.

```r
x<-TRUE
x
```

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

```r
y<-F
y
```

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

```r
Logical <- c(T,F,F,F,T,F,T,T,T,F,T,F)
Logical
```

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

```r
length(Logical)
```

```
## [1] 12
```

```r
Logical_1<- matrix(data=Logical,nrow=3,ncol=4)
Logical_1
```

```
##       [,1]  [,2] [,3]  [,4]
## [1,]  TRUE FALSE TRUE FALSE
## [2,] FALSE  TRUE TRUE  TRUE
## [3,] FALSE FALSE TRUE FALSE
```

Relational Operators

Relational operators are used to compare values,returning a logical result (TRUE or FALSE) based on the comparison.That are ==(Equal),=(Not Equal to),>(Greater than),<(Less than),>=(Greater than Equal to),<=(Less than Equal to),any() checks if at least one element in a logical vector is TRUE and all() checks if all elements in a logical vector are TRUE.

```r
1==2  # Equal
```

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

```r
1!=2  # Not Equal
```

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

```r
1>=2  # Greater than Equal to
```

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

```r
1<=2  # Less than Equal to
```

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

```r
x_1 <- c(3,2,1,4,1,2,1,-1,0,3)
x_2 <- c(4,1,2,1,1,0,0,3,0,4)

x_1==x_2    # Equal
```

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

```r
x_1!=x_2    # Not Equal
```

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

```r
x_1>=x_2    # Greater than Equal to
```

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

```
x_1<=x_2    # Less than Equal to
```

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

Logical Operators

Logical operators are used to combine or modify logical values.The key operators are:

1. **AND** (&, &&):Returns TRUE if both operands are TRUE.& is vectorized,meaning it operates on entire vectors element-wise.&& is element-wiseand only compares the first elements of vectors. It is primarily used in conditional statements (if, while).

```
TRUE & TRUE
```

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

```
TRUE && FALSE
```

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

2. **OR** (|, ||): Returns TRUE if at least one operand is TRUE.| is vectorized.|| is element-wise and mainly used in conditional statements.

```
TRUE | FALSE
```

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

```
FALSE || TRUE
```

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

3. **NOT** (!): Negates a logical value, flipping TRUE to FALSE and vice versa.

```
!TRUE
```

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

```
!FALSE
```

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

Logical as Numeric

R can interpret logical values as numeric values:TRUE is equivalent to 1 FALSE is equivalent to 0.This allows you to use logical values in arithmetic calcula- tions and with functions that expect numeric input.

```r
TRUE + TRUE + FALSE
```

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

```r
sum(c(TRUE, FALSE, TRUE))
```

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

Logical Sub-setting and Extraction

A key use of logical values is for sub-setting and extracting elements from vectors and other data structures.You can use a logical vector as an index inside square brackets ([ ]) to select elements. Only elements corresponding to TRUE values in the logical vector will be extracted.

```r
myvec <- c(5,-2.3,4,4,4,6,8,10,40221,-8) # for getting specific values.
myvec[c(F,T,F,F,F,F,F,F,F,T)]
```

```
## [1] -2.3 -8.0
```

```r
myvec[myvec<0]
```

```
## [1] -2.3 -8.0
```

The logical vector used for sub-setting must be the same length as the vector being subsetted, although recycling can occur if the logical vector is shorter.

```r
A <- matrix(c(0.3,4.5,55.3,91,0.1,105.5,-4.2,8.2,27.9),nrow=3,ncol=3)
A
```

```
##      [,1]  [,2] [,3]
## [1,]  0.3  91.0 -4.2
## [2,]  4.5   0.1  8.2
## [3,] 55.3 105.5 27.9
```

```r
A[c(T,F,F),c(F,T,T)]
```

```
## [1] 91.0 -4.2
```

```r
A<1
```

```
##       [,1]  [,2]  [,3]
## [1,]  TRUE FALSE  TRUE
## [2,] FALSE  TRUE FALSE
## [3,] FALSE FALSE FALSE
```

```r
A[A<1] <- -7
```