R_Variables

2023-03-04

variables in R

Naming convension

A variable provides us with named storage that our programs can manipulate.

A variable in R can store an atomic vector, group of atomic vectors or a combination of many R-objects.

A valid variable name consists of letters, numbers and the dot or underline characters. The variable name starts with a letter or the dot not followed by a number.

Valid variable names : var1, var, var_name2. , .var_name , var.name

In-Valid variable names : .1var, %var, _var_name2

Variable Assignment

The variables can be assigned values using leftward (<-), rightward (->) and equal to (=) operator.

Also following all operators act as assignment in R programming

<-, =, <<- :: left assignment

->, ->> :: right assignment

All following lines work in R programming

```
v1 = 10
v2 <- 20
30 -> v3
```

Printing a variable

By Name of the variable

Just writing name of variable and nothing else on single line in script will print the variable value

```
v1
```

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

By print()

print can be used to print a single variable at a time

```
print(v2)
```

```
## [1] 20
```

By cat()

cat function allows to print multiple variables at a time

```
cat(v1,v2,v3)
```

```
## 10 20 30
```

Finding Variables

To know all the variables currently available in the workspace we use the ls() function. Also the ls() function can use patterns to match the variable names.

```
print(ls())
```

```
## [1] "v1" "v2" "v3"
```

pattern finding in ls()

The ls() function can use patterns to match the variable names.

```
# List the variables starting with the pattern "v".
print(ls(pattern="v"))
```

```
## [1] "v1" "v2" "v3"
```

Deleting Variables

Variables can be deleted by using the rm() function. Below we delete the variable v3. On printing the value of the variable error is thrown.

```
rm(v3)

#print(v3)
# Print gives error : Error in print(v3) : object 'v3' not found
```

Deleting all variables in environment

It can be done using rm and ls

This is used in beginning of any R script to make sure that no old variables affect the execution of the script

Following line deletes all environment variables

```
rm(list=ls())
print(ls())
```

```
## character(0)
```

Controlling Visibility of Variables

This is way to use what variables are shown in current environment.

```
rm(list = ls())
x=10

print("List after creating x")
```

```
## [1] "List after creating x"
```

```
ls()
```

```
## [1] "x"
```

```
.xyz= 90

print("List after creating .xyz")
```

```
## [1] "List after creating .xyz"
```

```
ls()
```

```
## [1] "x"
```

```
print("List of all variables")
```

```
## [1] "List of all variables"
```

```
ls(all.names = TRUE)
```

```
## [1] ".xyz" "x"
```