EXPT NO:	RELATIONAL AND LOGICAL OPERATORS
DATE:29.1.25	

AIM:

To perform the basic relational and logical operator in R programming.

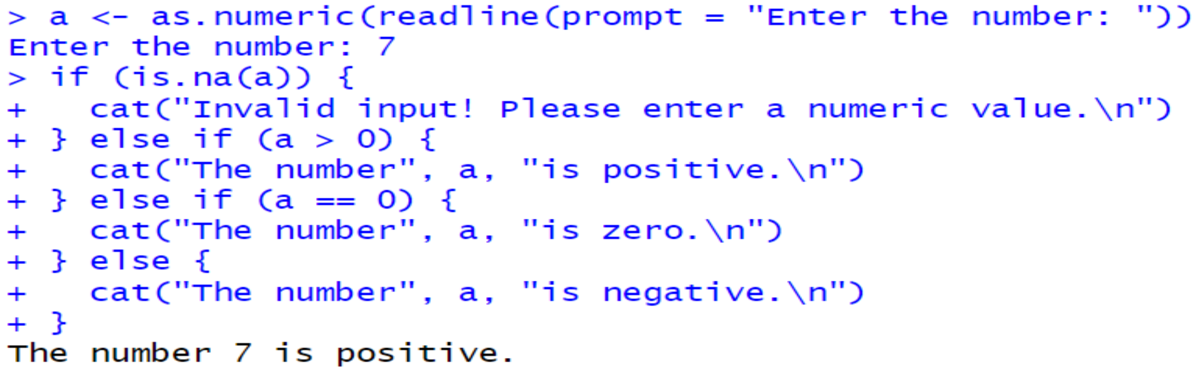
RELATIONAL OPERATORS:

1. Write an R program that checks whether a given number is positive, negative, or zero using relational operators.

Code:

```
a <- as.numeric(readline(prompt = "Enter the number: "))
if (is.na(a)) {
  cat("Invalid input! Please enter a numeric value.\n")}
 else if (a > 0) {
  cat("The number", a, "is positive.\n")}
 else if (a == 0) {
  cat("The number", a, "is zero.\n")}
 else {
  cat("The number", a, "is negative.\n")}
```

Output:



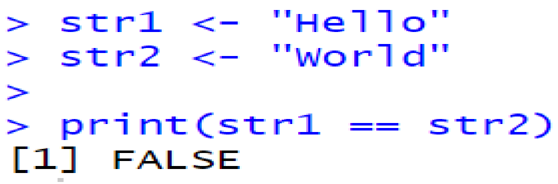
```
> a <- as.numeric(readline(prompt = "Enter the number: "))
Enter the number: 7
> if (is.na(a)) {
+   cat("Invalid input! Please enter a numeric value.\n")
+ } else if (a > 0) {
+   cat("The number", a, "is positive.\n")
+ } else if (a == 0) {
+   cat("The number", a, "is zero.\n")
+ } else {
+   cat("The number", a, "is negative.\n")
+ }
The number 7 is positive.
```

2. Assign two different strings to str1 and str2. Use the relational operator (==) to check if both strings are equal and print the result.

Code:

```
str1 <- "Hello"
str2 <- "World"
print(str1 == str2)
```

Output:



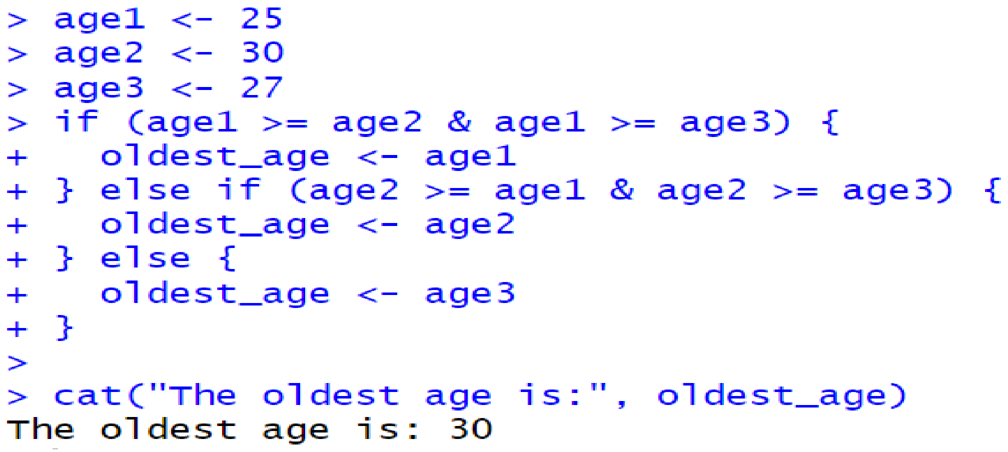
```
> str1 <- "Hello"
> str2 <- "World"
>
> print(str1 == str2)
[1] FALSE
```

3. Assign values to three variables: age1, age2, and age3. Use relational operators to determine and print the oldest age among the three.

Code:

```
age1 <- 25
age2 <- 30
age3 <- 27
if (age1 >= age2 & age1 >= age3) {
  oldest_age <- age1 }
 else if (age2 >= age1 & age2 >= age3) {
  oldest_age <- age2 }
 else {
  oldest_age <- age3 }
 cat("The oldest age is:", oldest_age)
```

Output:



```
> age1 <- 25
> age2 <- 30
> age3 <- 27
> if (age1 >= age2 & age1 >= age3) {
+   oldest_age <- age1
+ } else if (age2 >= age1 & age2 >= age3) {
+   oldest_age <- age2
+ } else {
+   oldest_age <- age3
+ }
>
> cat("The oldest age is:", oldest_age)
The oldest age is: 30
```

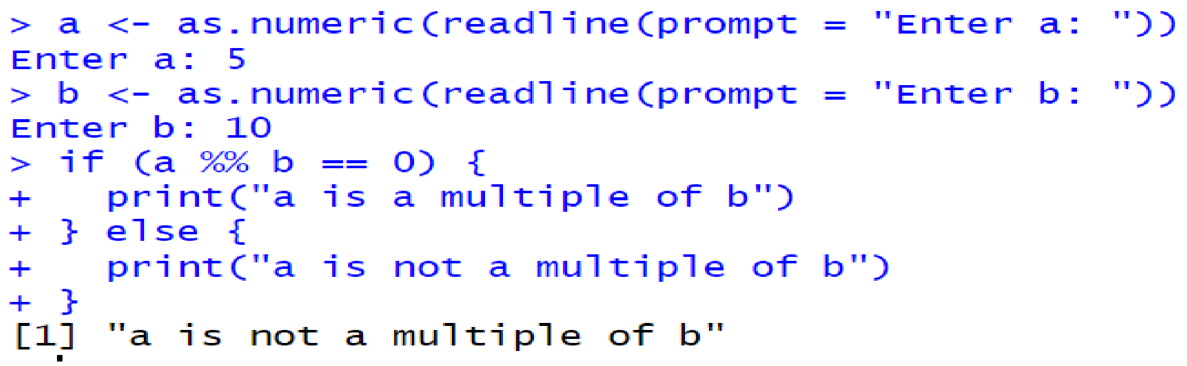
4. Write an R program that takes two numbers a and b and checks if a is a multiple of b using relational operators.

Code:

```
a <- as.numeric(readline("Enter a: "))
```

```
b <- as.numeric(readline("Enter b: "))
if (a %% b == 0) {
  print("a is a multiple of b")}
 else {
  print("a is not a multiple of b") }
```

Output:



```
> a <- as.numeric(readline(prompt = "Enter a: "))
Enter a: 5
> b <- as.numeric(readline(prompt = "Enter b: "))
Enter b: 10
> if (a %% b == 0) {
+   print("a is a multiple of b")
+ } else {
+   print("a is not a multiple of b")
+ }
[1] "a is not a multiple of b"
```

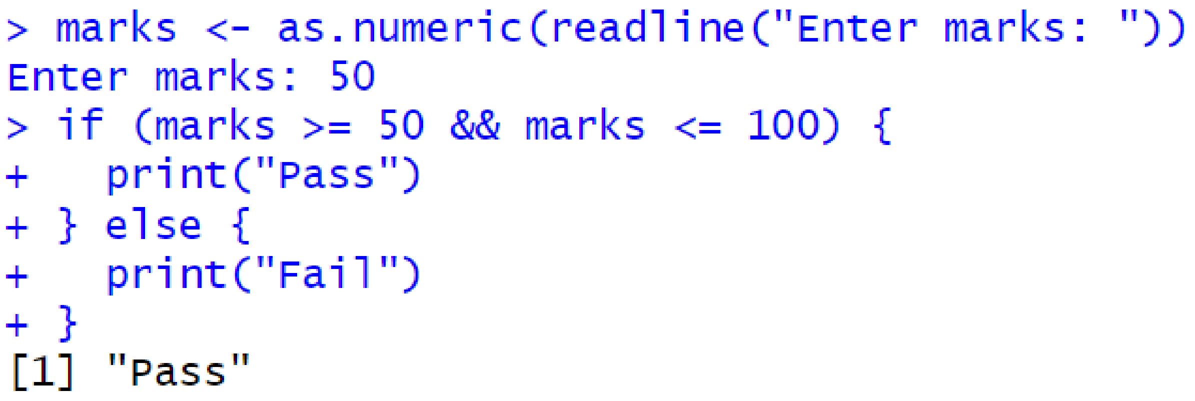
LOGICAL OPERATORS:

1.Write an R program that takes marks as input and prints "Pass" if marks is greater than or equal to 50; otherwise, print "Fail" using logical operators.

Code:

```
marks <- as.numeric(readline("Enter marks: "))
if (marks >= 50 && marks <= 100) {
  print("Pass") }
 else {
  print("Fail") }
```

Output:



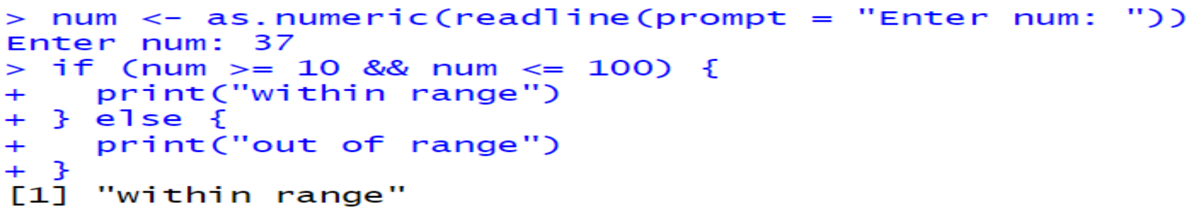
```
> marks <- as.numeric(readline("Enter marks: "))
Enter marks: 50
> if (marks >= 50 && marks <= 100) {
+   print("Pass")
+ } else {
+   print("Fail")
+ }
[1] "Pass"
```

2. Write an R program that takes a number (num) as input and checks whether it is between 10 and 100 (inclusive). Print "Within Range" if true; otherwise, print "Out of Range" using logical operators.

Code:

```
num <- as.numeric(readline("Enter num: "))
if (num >= 10 && num<= 100) {
  print("within range")}
 else {
  print("out of range") }
```

Output:



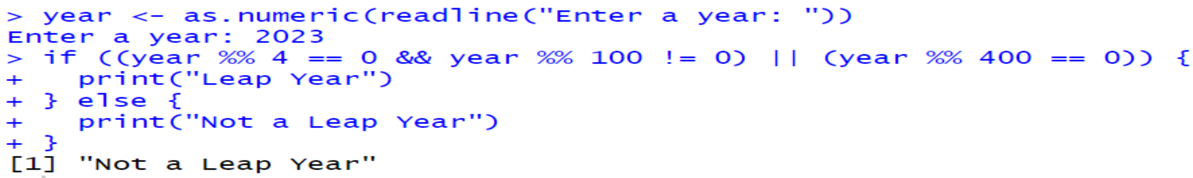
```
> num <- as.numeric(readline(prompt = "Enter num: "))
Enter num: 37
> if (num >= 10 && num <= 100) {
+   print("within range")
+ } else {
+   print("out of range")
+ }
[1] "within range"
```

3. Write an R program that takes a year (year) as input and checks whether it is a leap year (divisible by 4 but not by 100, except if also divisible by 400). Print "Leap Year" if true; otherwise, print "Not a Leap Year" using logical operators.

Code:

```
year <- as.numeric(readline("Enter a year: "))
if ((year %% 4 == 0 && year %% 100 != 0) || (year %% 400 == 0)) {
  print("Leap Year")}
 else {
  print("Not a Leap Year") }
```

Output:



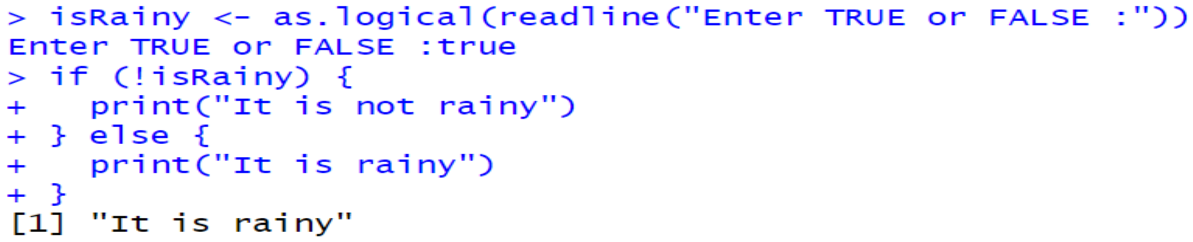
```
> year <- as.numeric(readline("Enter a year: "))
Enter a year: 2023
> if ((year %% 4 == 0 && year %% 100 != 0) || (year %% 400 == 0)) {
+   print("Leap Year")
+ } else {
+   print("Not a Leap Year")
+ }
[1] "Not a Leap Year"
```

4. Write an R program that takes a Boolean variable isRainy as input (TRUE/FALSE)and prints "It is not rainy" if isRainy is FALSE; otherwise, print "It is rainy" using the logical NOT (!) operator.

Code:

```
isRainy <- as.logical(readline("Enter TRUE or FALSE :"))
if (!isRainy) {
  print("It is not rainy")
} else {
  print("It is rainy")
}
```

Output:



```
> isRainy <- as.logical(readline("Enter TRUE or FALSE :"))
Enter TRUE or FALSE :true
> if (!isRainy) {
+   print("It is not rainy")
+ } else {
+   print("It is rainy")
+ }
[1] "It is rainy"
```

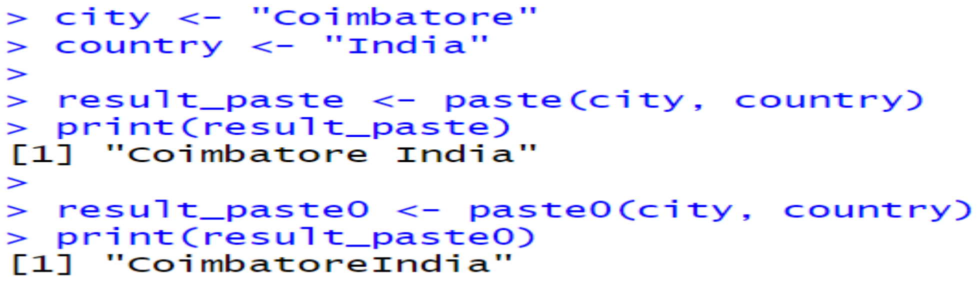
Use of paste() and paste0():

1. Define two variables, city and country, and assign them your city and country names. Use the paste() and paste0() functions to concatenate them and print the result.

Code:

```
city <- "Coimbatore"
country <- "India"
result_paste <- paste(city, country)
print(result_paste)
result_paste0 <- paste0(city, country)
print(result_paste0)
```

Output:



```
> city <- "Coimbatore"
> country <- "India"
>
> result_paste <- paste(city, country)
> print(result_paste)
[1] "Coimbatore India"
>
> result_paste0 <- paste0(city, country)
> print(result_paste0)
[1] "CoimbatoreIndia"
```

Investigate the DATA TYPES of the Following Values and Print Their Types:

a. 100.5

b. -25

c. "R Programming"

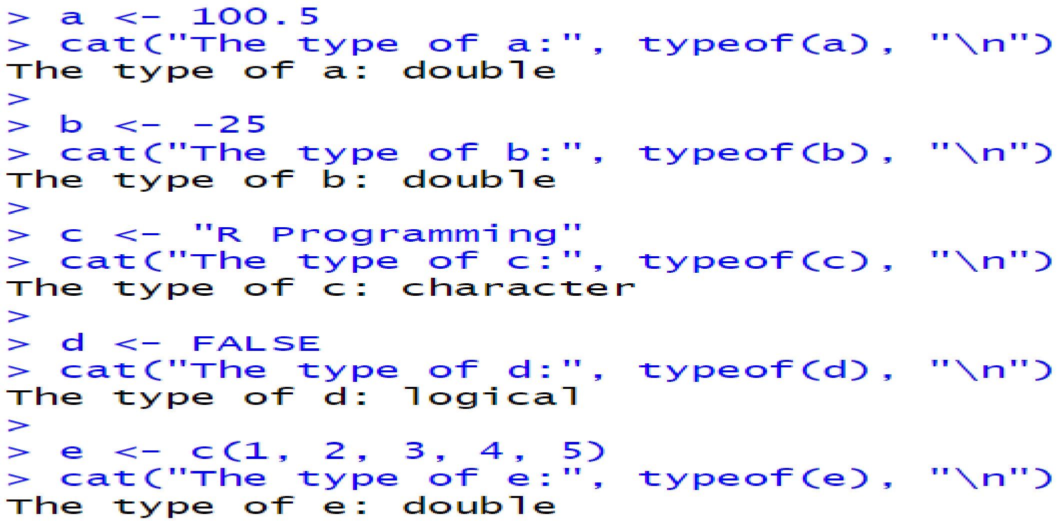
d. FALSE

e. c(1, 2, 3, 4, 5)

code:

```
a <- 100.5

cat("The type of a:", typeof(a), "\n")

b <- -25

cat("The type of b:", typeof(b), "\n")

c <- "R Programming"

cat("The type of c:", typeof(c), "\n")

d <- FALSE

cat("The type of d:", typeof(d), "\n")

e <- c(1, 2, 3, 4, 5)

cat("The type of e:", typeof(e), "\n")
```

OUTPUT:



```
> a <- 100.5
> cat("The type of a:", typeof(a), "\n")
The type of a: double
>
> b <- -25
> cat("The type of b:", typeof(b), "\n")
The type of b: double
>
> c <- "R Programming"
> cat("The type of c:", typeof(c), "\n")
The type of c: character
>
> d <- FALSE
> cat("The type of d:", typeof(d), "\n")
The type of d: logical
>
> e <- c(1, 2, 3, 4, 5)
> cat("The type of e:", typeof(e), "\n")
The type of e: double
```

RESULT:

Thus, the R programming is implemented and executed successfully.