if Statements

In this lesson, we will learn about if statements.

We'll cover the following

While writing a program, you may have to change the flow of the code depending on certain conditions and inputs.

For example, if we are dealing with numbers in our program we may want to print “positive” if the number we received is a positive integer.

Here, we will have to write a *condition* that tells us to print “positive” if we receive a positive number.

For such a situation, we need **Conditional** statements.

What are Conditional Statements?

Conditional statements are used in programming languages to perform different computations or actions depending on whether a certain condition is met.

Let’s have a look at an example:

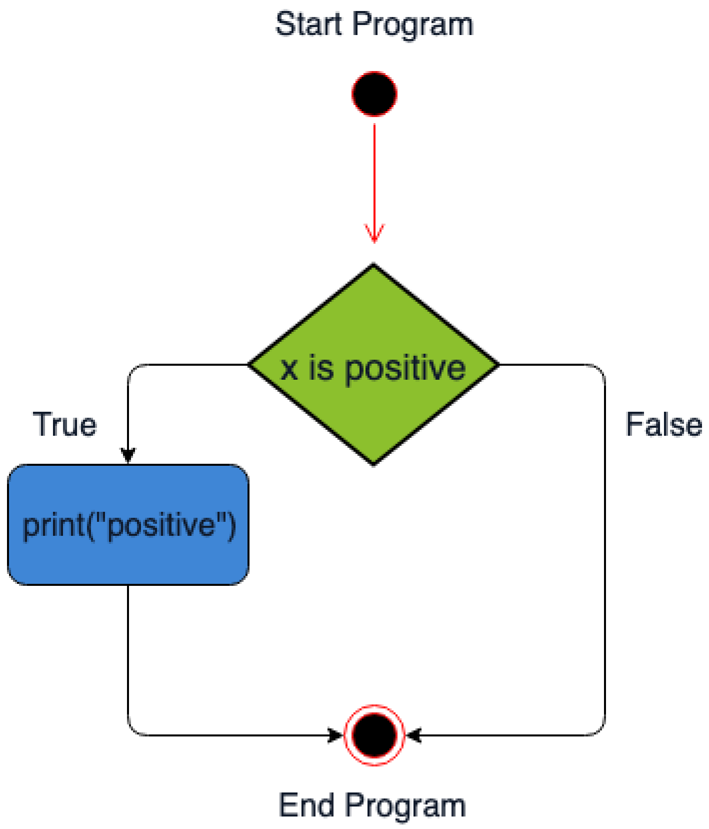

Diagram to show how code flow is altered.

The above diagram is an example of the `if` statement. It is called a **flow chart**. Here, the diamond boxes usually contain **conditions** and the rectangular boxes contain statements. The arrows show the flow of execution.

What is an `if` statement?

An `if` statement consists of a Boolean expression that when satisfied performs certain actions.

condition_satisfied | condition_unsatisfied

```
x <- 5
if(x > 0) # condition for checking positive numbers (all positive numbers are greater than 0)
{
  print("Positive number")
}
```

Conditional statement is satisfied

Notice, we have used a **Relational Operator** greater than `>` that returns `TRUE` or `FALSE` depending on the value of $x > 0$. This, in turn, determines whether or not

FALSE depending on the value of $x > 0$. This, in turn, determines whether or not to print “positive”.

Conditional statements allow programmers to deal with inputs or features of the data depending on different **R expressions**.

Let’s look at another example:

```
myVector <- c(1, 2, 3, 5)
testNumber <- 4

if(!testNumber %in% myVector)
{
  print("Not Found!")
}
```

Let’s breakdown the condition `!testNumber %in% myVector` **Line number 4**.

Conditional Statement Breakdown:

1. `testNumber %in% myVector` checks whether the number contained in the variable $testNumber$ is present in $myVector$. In this case, whether the number 4 is present in the vector.
2. For this particular code snippet, the answer is `FALSE`.
3. Now, the symbol `!` takes the **NOT** of the result of `testNumber %in% myVector`.
4. In our case, it takes **NOT** of `FALSE` which is `TRUE`.
5. Hence, our conditional statement is *satisfied*. We can now proceed to the statement.

```
print("Not Found!")
```

In the next lesson, we will be learning about **if-else statements**.