6.1.3

# Explain the difference between | , || , & and &&

In R, the symbols |, ||, &, and && are used as logical operators to combine logical expressions or test conditions.

The "|" (vertical bar) and "&" (ampersand) operators are used for element-wise comparisons. They compare corresponding elements of two vectors or data frames and return a vector of the same length as the input vectors. The "|" operator returns TRUE if either one or both of the elements being compared are TRUE. The "&" operator returns TRUE only if both elements being compared are TRUE.

The "||" (double vertical bar) and "&&" (double ampersand) operators also compare logical expressions, but they are used for short-circuit evaluations. The "||" operator evaluates the left-hand side expression first, and if it is TRUE, it does not evaluate the right-hand side expression. If the left-hand side is FALSE, it evaluates the right-hand side expression and returns the result. The "&&" operator works similarly but evaluates the right-hand side expression only if the left-hand side is TRUE.

Here is an example to illustrate the difference between these operators:

x <- 5

y <- 10

# Element-wise comparisons

x > 4 | y < 5 # returns TRUE

x > 4 & y < 5 # returns FALSE

# Short-circuit evaluations

x > 4 || y < 5 # returns TRUE

x > 4 && y < 5 # returns FALSE

x < 10 && y < 5 # returns FALSE without evaluating y < 5

6.2.1

# What are the components of an if-else statement? Define the syntax.

An if-else statement is a control flow statement in programming that allows you to execute different blocks of code based on certain conditions. The components of an if-else statement are:

1. The "if" keyword, which starts the if-else block.
2. A condition or test expression enclosed in parentheses, which is evaluated to either TRUE or FALSE.
3. The block of code to be executed if the condition is TRUE, enclosed in curly braces {}.
4. The "else" keyword, which starts the else block.
5. Another block of code to be executed if the condition is FALSE, enclosed in curly braces {}.

The general syntax for an if-else statement in most programming languages is as follows:

if (condition) {

# code block to be executed if condition is TRUE

} else {

# code block to be executed if condition is FALSE

}

# Write a statement that prints "Rotten!" if the input is equal to or larger than 1 and less than 60,

# "Fresh" if the input is equal to or larger than 60 and less than 75,

# and "Certified Fresh!" if the input is equal to or larger than 75.

# For any other value, the statement should print "Please input a number between 1 and 100"

if (input >= 1 & input < 60) {

print("Rotten!")

} else if (input >= 60 & input < 75) {

print("Fresh")

} else if (input >= 75 & input <= 100) {

print("Certified Fresh!")

} else {

print("Please input a number between 1 and 100")

}

# Bonus exercise:

# Imagine the following: Your mother announced yesteday that she's going to win the lottery. She did it with the casual

# nonchalance of a person who's going to the shop to pick up some pasta. Today is the number draw.

# If she gets 6 numbers correct, she wins the lottery. If she doesn't get 6 numbers correct, she will sit staring

# at the screen wondering how on earth she did not manage to achive this very simple task.

# Write a statement that will print "Mom wins the lottery!" if the six numbers she guessed matched the lottery numbers,

# and "Mom did not win the lottery." if the numbers do not match.

# Hint: it will be best if you create one object with randomly generated numbers, and another object which you populate

# manually (as if you're playing the lottery), and compare the two.

# Hint: look into the length() and setdiff() functions

drawLottery <- function() {

return (round(runif(6)\*100))

}

momNums <- c(1,2,3,4,5,6)

test <- c(1,2,3,4,5,6)

lotteryNums <- drawLottery()

if (length(setdiff(momNums,drawLottery())) == 0) {

print("Mom wins the lottery!")

} else {

print("Mom did not win the lottery.")

}

A screenshot of a computer game

Description automatically generated

A white background with blue text

Description automatically generated

6.3

# Write a for loop that sums all the numbers from 1 to n

n <- 10

sum <- 0

for(i in 1:n) {

sum = sum + i

}

A white background with black text

Description automatically generated