In programming, an if statement can be used to evaluate a given test expression. If statements can be very useful for controlling the flow of a program, as well as deciding which instructions should be run at a given point in the code sequence. Should the given expression evaluate to true (or a nonzero value), then the statements within the body of the if is executed. Similarly, should the expression evaluate to false (or zero), then the statements inside the body of the if are skipped. This is also known as conditional logic, since the expression being tested by the if statement can also be considered a condition that has to be met in order to execution to proceed. An example program using an if statement may go as follows:

print("---")

num = int(input("Enter an integer: "))

if num > 0:

print("You entered a positive number")

print("Your number is",num)

print("---")

In this instance, our expression that is being tested is described as num > 0. So, if our entered number is greater than 0, then the program should execute the lines that we have in the body of the if statement. Note that in Python, a line is considered within our if statement if it is indented. Thus, both of our print statements will be executed provided that our expression (num > 0) evaluates to true.

We can test the program ourselves by running two separate instances. In the first one, enter a positive number. In the second one, enter a negative number. You will see that in the case of the negative number, neither of the print statements will be executed.

We can modify our program to include else statements to go along with our if. Else statements may optionally be joined with our if, and will only execute if our expression evaluates to false. Using an if/else statement together, we can ensure that statements can run if our expression evaluates to true, as well as if our expression evaluates to false.

print("---")

num = int(input("Enter an integer: "))

if num > 0:

print("You entered a positive number")

print("Your number is",num)

else:

print("You entered a negative number")

print("Your number is",num)

print("---")

Now when we enter a negative number, the statements following the else statement will be executed.

We can simplify our program as such:

print("---")

num = int(input("Enter an integer: "))

if num > 0:

print("You entered a positive number")

else:

print("You entered a positive number")

print("Your number is",num)

print("---")

Now, regardless of how our number is evaluated, the print statement that displays our number will always be run. This is because this statement is not encompassed within our if/else statement. It not indented, and so the else statement will not notice it.

It is important to note that else statements cannot be used on their own without an if statement in front of it. Also note that there can be no more than one else statement to go alongside any individual if statement.

A useful property about if statements is that they can be used even when inside other if statements. These type of if statements are known as nested if statements, and work exactly the same as non-nested if statements.

print("---")

num = int(input("Enter an integer: "))

if num > 0:

print("You entered a positive number")

if num < 10:

print("Your number is between 0 and 10")

else:

print("You entered a negative number")

if num > -10:

print("Your number is greater than -10")

print("Your number is",num)

print("---")

In order for the nested if statements to be reached, the conditions of the parent if statement must also be met. In this case, our condition for testing if our number is less than 10 will only be reached if our number also happens to be greater than 0, as defined by the parent if statement.

The last variation of if statements we will cover is known as else-if statements (abbreviated to elif in Python). Elif statements provide additional conditions that can be made if our initial if statement evaluates to false and we don’t want to use a regular else statement right away (or at all). Unlike if statements and else statements, there can be as many elif statements as desired so long as there is an initial if statement to lead the group. Elif statements are written very similarly to if statements.

print("---")

num = int(input("Enter an integer: "))

if num > 0:

print("You entered a positive number")

if num < 10:

print("Your number is between 0 and 10")

elif num == 0:

print("You entered 0")

else:

print("You entered a negative number")

if num > -10:

print("Your number is between -10 and 0")

print("Your number is",num)

print("---")

To summarize, if statements can be used for checking for certain conditions to be met through the use of test expressions. These expressions must be able to evaluate to true (nonzero) or false (zero) in order to be properly tested. Else statements can be used in combination with if statements to account for cases in which the expression equals false. Nested if statements can be placed within if statements to provide further logical depth and check other conditions provided that execution reaches it from the parent if statement. Finally, else-if or elif statements can be used to define extra conditions that can be met if the initial if statement evaluates to false.