Conditional Statements

by Sophia

WHAT'S COVERED

In this lesson, you will learn about conditional statements and how to use them in Python. Specifically, this lesson covers:

1. if Statements

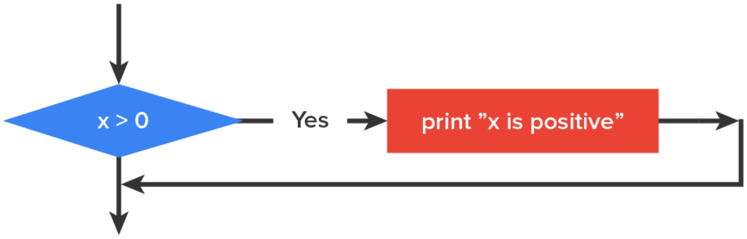
In order to write useful programs, we almost always need the ability to check conditions and change the behavior of the program accordingly. Conditional statements give us this ability. A **conditional statement** is a statement that controls the flow of execution depending on some condition. The simplest form of a conditional statement is the if statement:

EXAMPLE

```
temperature = 82
if temperature > 0 :
  print('temperature is positive')
```

The boolean expression after the if statement (the “>”) is called the condition. A **condition** is a boolean expression in a conditional statement that determines which branch is executed. To finish this simple if statement, we end the statement with a colon character (:) and the line(s) after the if statement is indented.

Here is a visual flowchart of what is happening in this statement.



If the logical condition is true, then the indented statement gets executed, and ‘temperature is positive’ is the output to the console. If the logical condition is false, the indented statement is skipped.

Note that if statements have the same structure as function definitions. The if statement consists of a header line that ends with the colon character (:) followed by an indented block. Statements like this are called **compound statements** because they stretch across more than one line.

EXAMPLE

```
#this is assigning variable temperature the integer 82
temperature = 82

#below is the if statement
if temperature > 0 : #this is the header line that ends with a “:”
  print('temperature is positive')  #this is the indented block
```

DID YOU KNOW

Indentation is vital for Python coding. Many other programming languages use indentation as a means to keep the code readable. However, for Python, indentation is a requirement.

Python uses indentation to indicate blocks of code. The `print()` function is indented; that tells Python that the `print()` function is included with the if statement. Python knows it is a compound statement.

Just how important is indenting? Let's see what happens when we forget to indent.

```
temperature = 82

if temperature > 0 :
print('temperature is positive')
```

Here is the accompanying output.

```
  File "/home/main.py", line 4
    print('temperature is positive')
    ^
IndentationError: expected an indented block after 'if' statement on line 3
```

Oops, we received an indentation error.

There is no limit on the number of statements that can appear in the body, but there must be at least one. The body consists of the lines of code that are part of the condition. In essence, the body includes all of the lines of code that are indented. Occasionally, it is useful to have a body with no statements (usually as a placeholder for code you haven't written yet). In that case, you can use the pass statement.

EXAMPLE

```
if temperature < 0 :
  pass #need to handle negative temperatures!!
```

If you remember, `pass` is a reserved keyword in Python. A **pass statement** is essentially a statement that will do nothing. The difference between a `pass` statement and a comment is that comments are completely ignored by the interpreter, whereas the `pass` statement is not. It is seen, but it does nothing.

TERMS TO KNOW

Conditional Statement
A conditional statement is a statement that controls the flow of execution depending on some condition.

Condition
A condition is a boolean expression in a conditional statement that determines which branch is executed.

Compound Statement
A statement that consists of a header line that ends with the colon character (:) followed by an indented block. It is called compound since it stretches across more than one line.

pass
A `pass` statement is a reserved keyword that is essentially a statement that will do nothing.

2. Boolean Expressions

Previously we mentioned that a condition is a boolean expression in a conditional statement that determines which branch is executed. A **boolean expression** is an expression that is either true or false. In the previous example, we used the greater than symbol (">") to check if the temperature was greater than 0 and if so, print it to the screen. We used the greater than symbol as the comparison operator. A **comparison operator** is one of a group of operators that compares two values. Let's list out and define the six (6) comparison operators.

Operator	Name	Example	Comment
==	Equal	x == y	x is equal to y

!=	Not equal	x != y	x is not equal to y
>	Greater than	x > y	x is greater than y
<	Less than	x < y	x is less than y
>=	Greater than or equal to	x >= y	x is greater than or equal to y
<=	Less than or equal to	x <= y	x is less than or equal to y

Note: Although these operations are probably familiar to you, the Python symbols are different from the mathematical symbols for the same operations. A common error is to use a single equal sign (=) instead of a double equal sign (==). Remember that `=` is an assignment operator and `==` is a comparison operator. There is no such thing as =< or =>.

TERMS TO KNOW

Boolean Expression
A boolean expression is an expression whose value is either true or false.

Comparison Operator
A comparison operator is one of a group of operators that compares two values. These include ==, !=, >, <, >=, and <=.

3. elif Statements

Using only if statements on their own can be problematic because they each run independently. Let’s take a look at this snippet of code.

EXAMPLE

```
grade = 85
if grade > 90:
  print("You got an A")
if grade > 80:
  print("You got a B")
if grade > 70:
  print("You got a C")
if grade > 60:
  print("You got a D")
if grade <= 60:
  print("You got a F")
```

In looking at this code, it seems that if the grade was an 85, “You got a B” (without the quotation marks) should be output to the console. However, this is what is the output:

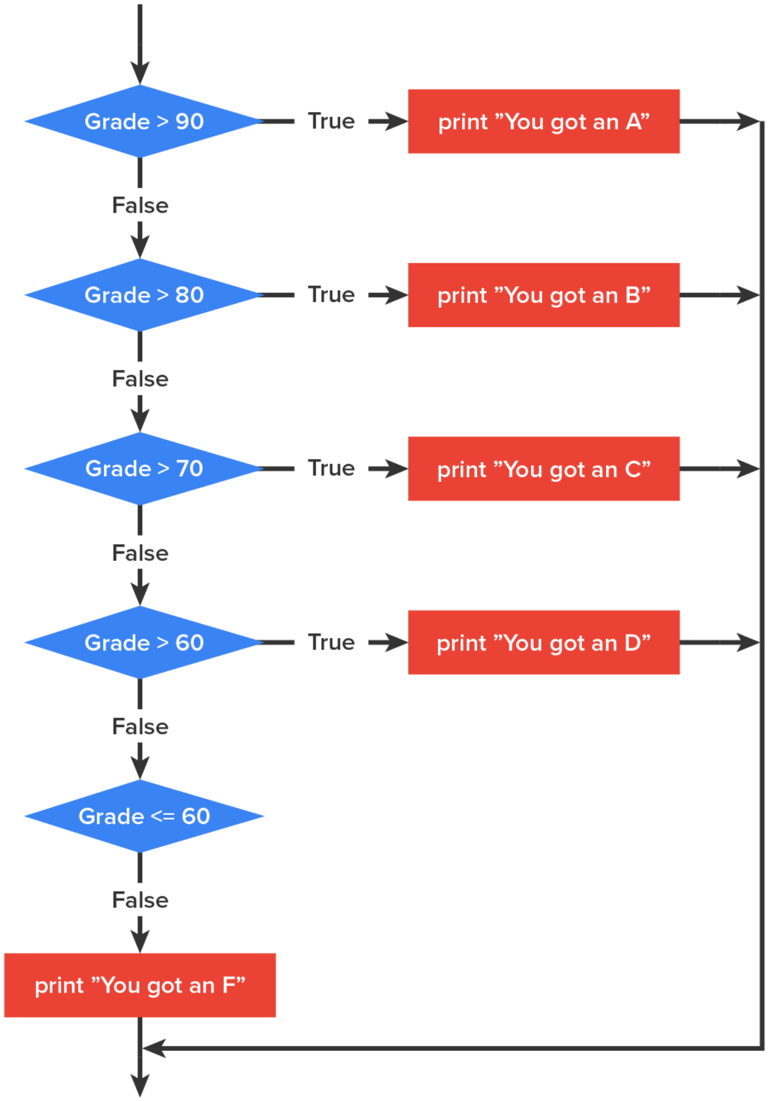
```
You got a B
You got a C
You got a D
```

That’s certainly not what should happen. Since each of the if statements run independently and aren’t connected, each if statement is tested separately.

When there are more than two possibilities in a conditional statement, we need more than two branches. One way to express a computation like that is a chained conditional. A **chained conditional** is a conditional statement with a series of alternative branches. The conditions are checked one at a time and the program exits the conditional statement if any branch is true.

In order to make our previous code a cohesive single set of statements, we can make a chained conditional by making use of the elif statement. The elif stands for “else if” and it matches an existing if statement. It is basically stating that if the current condition is not true, try the next condition. This way, once any conditional statement is true, only that branch is output, and the overall if statement ends.

Let’s see this in flowchart form first.



Let's go back to our example code using elif.

EXAMPLE

```
grade = 85
if grade > 90:
  print("You got an A")
elif grade > 80:
  print("You got a B")
elif grade > 70:
  print("You got a C")
elif grade > 60:
  print("You got a D")
```

```
elif grade <= 60:
  print("You got a F")
```

Here is the output from using the elif statements.

```
You got a B
```

When running the code with the elif statements, the first if statement came back false. Then, the next condition (first elif) was checked. That came back true (85 > 80), so we received that elif statement's indented output as we expected.

TERM TO KNOW

Chained Conditional

A chained conditional is a conditional statement with a series of alternative branches.

4. else Statements

The else statement can be viewed as a catchall case whereby if none of the if or elif statements were satisfied, the else statement would run. Note that we do not need to have any conditional criteria on the else statement. Using the same example, we can change it such that the last condition is an else.

EXAMPLE

```
grade = 15
if grade > 90:
  print("You got an A")
elif grade > 80:
  print("You got a B")
elif grade > 70:
  print("You got a C")
elif grade > 60:
  print("You got a D")
else:
  print("You got a F")
```

With the output, we received an "F".

```
You got a F
```

Since we changed the original grade variable to 15 points, we received what we expected, an F. Each of the elif statements proved false and it continued to the else statement.

We can use our first example with the temperature check to use all three statements to consider the scenario of the temperature being positive, 0, or negative. We will change the temperature variable to -5 for this test run.

EXAMPLE

```
temperature = -5
if temperature > 0 :
  print('temperature is positive')
elif temperature == 0:
  print('temperature is 0')
else:
  print('temperature is negative')
```

Here is that output.

```
temperature is negative
```

TRY IT

You have seen examples using if, elif, and else statements.

Directions: Use the IDE to test out some of the statements you have just learned.

SUMMARY

In this lesson, we learned about conditional statements, starting with the **if statement**. We found out that conditional statements are also called compound statements because they stretch across more than one line to include an indented block of code. We also saw the importance of indenting these blocks of code since Python requires it to function correctly. We were then introduced to the **boolean expressions** that can be used when creating conditional statements. Finally, we learned that the if statement can be problematic since if statements each run independently. Therefore, it makes sense to create a chained conditional utilizing the **elif** and **else statements** to ensure all conditions are tested.

Best of luck in your learning!

Source: THIS CONTENT AND SUPPLEMENTAL MATERIAL HAS BEEN ADAPTED FROM “PYTHON FOR EVERYBODY” BY DR. CHARLES R. SEVERANCE ACCESS FOR FREE AT www.py4e.com/html3/ LICENSE: CREATIVE COMMONS ATTRIBUTION 3.0 UNPORTED.

TERMS TO KNOW

Boolean Expression

A boolean expression is an expression whose value is either true or false.

Chained Conditional

A chained conditional is a conditional statement with a series of alternative branches.

Comparison Operator

A comparison operator is one of a group of operators that compares two values. These include ==, !=, >, <, >=, and <=.

Compound Statement

A statement that consists of a header line that ends with the colon character (:) followed by an indented block. It is called compound since it stretches across more than one line.

Condition

A condition is a boolean expression in a conditional statement that determines which branch is executed.

Conditional Statement

A conditional statement is a statement that controls the flow of execution depending on some condition.

pass

A pass statement is a reserved keyword that is essentially a statement that will do nothing.