

Reference guide: Python concepts from module 1

Google Cybersecurity Certificate

Sections

[Comments](#)

[Functions](#)

[Conditional statements](#)

[Iterative statements](#)

Comments

The following syntax is used to create a comment. (A comment is a note programmers make about the intention behind their code.)

#

Starts a line that contains a Python comment

```
# Print approved usernames
```

Contains a comment that indicates the purpose of the code that follows it is to print approved usernames

Functions

The following functions are commonly used in Python.

print()

Outputs a specified object to the screen

```
print("login success")
```

Outputs the string "login success" to the screen

```
print(9 < 7)
```

Outputs the Boolean value of `False` to the screen after evaluating whether the integer 9 is less than the integer 7

type()

Returns the data type of its input

```
print(type(51.1))
```

Returns the data type of float for the input of 51.1

```
print(type(True))
```

Returns the data type of Boolean for the input of `True`

range()

Generates a sequence of numbers

```
range(0, 5, 1)
```

Generates a sequence with a start point of 0, a stop point of 5, and an increment of 1; because the start point is inclusive but the stop point is exclusive, the generated sequence is 0, 1, 2, 3, and 4

```
range(5)
```

Generates a sequence with a stop point of 5; when the start point is not specified, it is set at the default value of 0, and when the increment is not specified, it is set at the default value of 1; the generated sequence is 0, 1, 2, 3, and 4

Conditional statements

The following keywords and operators are used in conditional statements.

if

Starts a conditional statement

```
if device_id != "la858zn":
```

Starts a conditional statement that evaluates whether the `device_id` variable contains a value that is not equal to `"la858zn"`

```
if user in approved_list:
```

Starts a conditional statement that evaluates if the `user` variable contains a value that is also found in the `approved_list` variable

elif

Precedes a condition that is only evaluated when previous conditions evaluate to `False`; previous conditions include the condition in the `if` statement, and when applicable, conditions in other `elif` statements

```
elif status == 500:
```

When previous conditions evaluate to `False`, evaluates if the `status` variable contains a value that is equal to `500`

else

Precedes a code section that only evaluates when all conditions that precede it within the conditional statement evaluate to `False`; this includes the condition in the `if` statement, and when applicable, conditions in `elif` statements

```
else:
```

When previous conditions evaluate to `False`, Python evaluates this `else` statement

and

Requires both conditions on either side of the operator to evaluate to `True`

```
if username == "bmoreno" and login_attempts < 5:
```

Evaluates to `True` if the value in the `username` variable is equal to `"bmoreno"` and the value in the `login_attempts` variable is less than 5

or

Requires only one of the conditions on either side of the operator to evaluate to `True`

```
if status == 100 or status == 102:
```

Evaluates to `True` if the value in the `status` variable is equal to 100 or the value in the `status` variable is equal to 102

not

Negates a given condition so that it evaluates to `False` if the condition is `True` and to `True` if it is `False`

```
if not account_status == "removed"
```

Evaluates to `False` if the value in the `account_status` variable is equal to `"removed"` and evaluates to `True` if the value is the `account_status` variable is not equal to `"removed"`

Iterative statements

The following keywords are used in iterative statements.

for

Signals the beginning of a `for` loop; used to iterate through a specified sequence

```
for username in ["bmoreno", "tshah", "elarson"]:
```

Signals the beginning of a `for` loop that iterates through the sequence of elements in the list `["bmoreno", "tshah", "elarson"]` using the loop variable `username`

```
for i in range(10):
```

Signals the beginning of a `for` loop that iterates through a sequence of numbers created by `range(10)` using the loop variable `i`

while

Signals the beginning of a `while` loop; used to iterate based on a condition

```
while login_attempts < 5:
```

Signals the beginning of a `while` loop that will iterate as long as the condition that the value of `login_attempts` is less than `5` evaluates to `True`

break

Used to break out of a loop

continue

Used to skip a loop iteration and continue with the next one