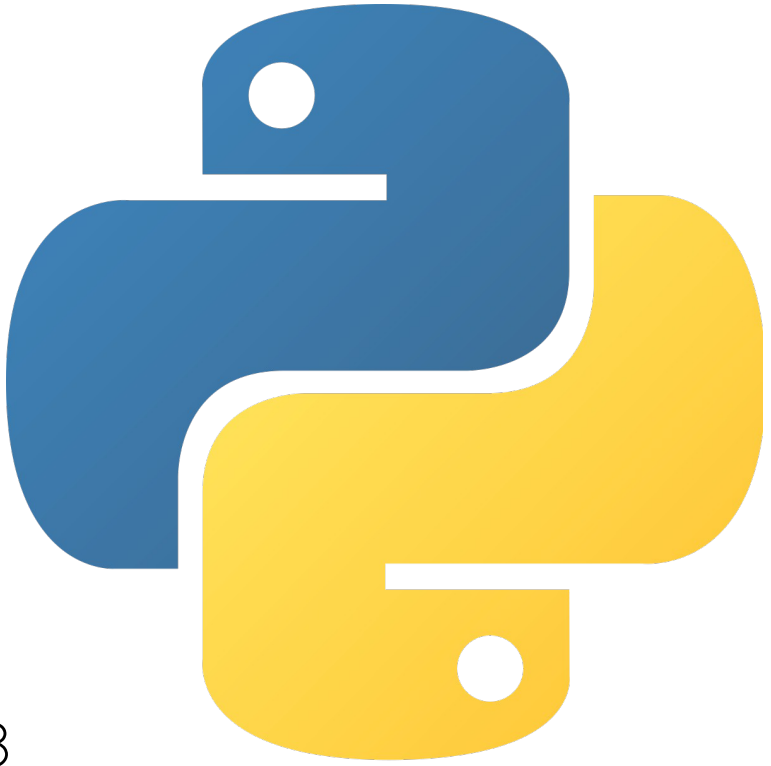
HackerFrogs Afterschool
Python Programming Basics: Part 4

Class:
Programming (Python)

Workshop Number:
AS-PRO-PY-04

Document Version:
1.75

Special Requirements:
Completion of AS-PRO-PY-03



What We Learned Before

This workshop is the fourth intro class to Python programming.

During our last workshop, we learned about a few programming concepts through Python, including the following:

String Formatting

```
name = "theshyhat"
daily_coffee = 3
print(f"{name} drinks {daily_coffee} cups of coffee a day.")
```

```
theshyhat drinks 3 cups of coffee a day.
```

String formatting is a way for us to include variable values in strings by using f-strings.

Basic String Operations

```
>>> city = "Seattle"
>>> city.count("e")
2
```

Strings in Python have a number of different functions and methods that can be useful for obtaining specific details about the strings.

Basic String Operations

```
>>> bug = "centipede"
>>> bug[0:9:2]
cniee
```

We can also output slices of strings through string indexing.

This Workshop's Topics

- Boolean Types

- Conditions

Conditions

Conditions are how programs make decisions as to which instructions to perform at a specific point in program execution.

Conditions

Generally, conditions trigger when a statement evaluates to **True** or **False**, so let's discuss what this means, exactly.

Conditions

The basic conditional statement in Python is the **if** statement, and **if** statements are written in code blocks like the following:

Conditions

```
password = 'mysecretpassword'

if password == 'mysecretpassword':
    print('password correct')
else:
    print('incorrect password')
```

Here, the program will print one message if the password variable is the string 'mysecretpassword' (Boolean True), and print another message if it is not (Boolean False).

Conditions

```
password = 'mysecretpassword'

if password == 'mysecretpassword':
    print('password correct')
else:
    print('incorrect password')
```

If statements start with if, then a Boolean equation, then a semicolon.

Conditions

```
password = 'mysecretpassword'

if password == 'mysecretpassword':
    print('password correct')
else:
    print('incorrect password')
```

Then an indentation on the next line, followed by instructions if the condition is met, then on the next line else:, then an indent on the next line, followed by instructions if the condition is not met.

Conditions

```
password = 'mysecretpassword'

if password == 'mysecretpassword':
    print('password correct')
else:
    print('incorrect password')

print('Was your password correct?')
```

The **else** portion of the code block is optional. After the **if** statement code block finishes, program execution continues on the next line.

Boolean Operators

Operator	Meaning
==	Equal to
<	Less than
>	Greater than
!=	Not equal to
<=	Less than or equal to
>=	Greater than or equal to

We can use any of the above Boolean operators in If Statements.

= and == are not the same

```
number = 42


if number == 42
  print("The number is 42!")
```

A common mistake using conditionals is to use the = symbol for comparisons. The == symbol is used for comparisons in conditionals.

Else Keyword

```
number = int(input("Input a number"))
if number % 2 == 0:
    print(f"{number} is an even number")
else:
    print(f"{number} is an odd number")
```

As mentioned earlier, the **else** keyword can be used with **if** conditionals to provide instructions to be executed if the **if** condition is not triggered.

Elif Keyword

```
number = input("Input a number")
if int(number) % 2 == 0:
    print(f"{number} is an even number")
elif int(number) % 2 == 1:
    print(f"{number} is an odd number")
else:
    print(f"{number}? Is that an integer?")
```

However, there's a third keyword, **elif**, which can be used if there is more than one condition to check for. We can include as many elif keywords as we want in an if codeblock. See the following..

Elif Keyword

```
age = int(input("Please enter your age: "))
if age < 0:
    print("Invalid age. Please enter a positive number.")
elif age <= 1:
    print("You are an infant.")
elif age <= 3:
    print("You are a toddler.")
elif age <= 12:
    print("You are a child.")
elif age <= 19:
    print("You are a teenager.")
elif age <= 35:
    print("You are a young adult.")
elif age <= 50:
    print("You are an adult.")
elif age <= 65:
    print("You are middle-aged.")
else:
    print("You are a senior."))
```

The AND Operator

```
birthday = 'October 24th'
todays_date = 'October 24th'

if birthday == 'October 24th' and todays_date == 'October 24th':
  print('Happy birthday!')
```

```
Happy birthday!
```

When using the AND operator, both statements to the right and left of the operator must be True in order for the overall statement to evaluate to True.

The AND Operator

```
birthday = 'October 24th'
todays_date = 'October 24th'

if birthday == 'October 24th' and todays_date == 'October 24th':
  print('Happy birthday!')
```

```
Happy birthday!
```

In this case, since the variables **birthday** and **todays_date** are the same day (October 24th), the **if** statement returns True, and the program will print **Happy Birthday!**

The OR Operator

```
favorite_food = 'spaghetti'
hobby = 'watching movies'

if favorite_food == 'spaghetti' or hobby == 'classical painting':
    print("Let's take a vacation to Italy!")
```

```
Let's take a vacation to Italy!
```

When using the OR operator either statement to the left or right of the operator must evaluate True for the overall statement to evaluate to True.

The OR Operator

```
favorite_food = 'spaghetti'
hobby = 'watching movies'

if favorite_food == 'spaghetti' or hobby == 'classical painting':
    print("Let's take a vacation to Italy!")
```

```
Let's take a vacation to Italy!
```

In this example, the if condition would not trigger if **both** statements to the left and right of the OR operator were False.

The IN Operator

```
name = "John"
party_guests = ["John", "Sue", "Bobby"]
if name in party_guests:
    print("You're invited to the party!")
```

```
You're invited to the party!
```

The IN operator is used to check if there is a specific item within a container object, such as a list.

The IN Operator

```
name = "John"
party_guests = ["John", "Sue", "Bobby"]
if name in party_guests:
    print("You're invited to the party!")
```

```
You're invited to the party!
```

Here the print instruction is executed because the string listed in the **name** variable is included as an item in the **party_guests** list.

The IS Operator

```
x = [2, 5, 7]
y = x
z = [2, 5, 7]
print(x == z)
print(x is z)
print(x is y)
```

```
True
False
True
```

The IS operator is used to compare whether an object is exactly referencing another object, as opposed to matching the values contained within.

The NOT Operator

```
telling_the_truth = False

if telling_the_truth == (not False):
  print("You're telling the truth!")
else:
  print("You're lying!")
```

```
You're lying!
```

The NOT operator is essentially an inversion of the whatever Boolean value is paired with it.

True is 1 and False is 0

```
var_a = 1
var_b = 0

if var_a == True:
  print("In Python, 1 and True are considered the same")
if var_b == False:
  print("Likewise, 0 and False are considered the same")
```

```
In Python, 1 and True are considered the same
Likewise, 0 and False are considered the same
```

Conditionals Exercise

Let's practice our if conditionals in Python at the following URL:

https://learnpython.org/en/Conditions

Part 1 – Conditionals Quiz

What will the output be?

```
hacking = 1
if hacking == True:
  print("We're hacking!")
```

A) We're hacking!
B) True
C) Nothing will print, hacking does not equal True
D) Nothing will print, the syntax is incorrect

Part 1 – Conditionals Quiz

What will the output be?

```
hacking = 1
if hacking == True:
  print("We're hacking!")
```

A) We're hacking!
B) True
C) Nothing will print, hacking does not equal True
D) Nothing will print, the syntax is incorrect

Workshop Review Exercise

Let's write a program that uses some of the concepts we learned in this workshop.

We could write the program on the online-python.com webpage.

The program should take one user-provided input in the form of an integer

Workshop Review Exercise

The program should print out the string

X is an even number!
if the variable is an even number, or

X is an odd number!
if the variable is an odd number.

The value of the variable should substitute in for X in the printed statement. E.g., '7 is an odd number!'

Workshop Review Exercise

We'll present a possible solution after this, but you should attempt to write your own program now. Remember, the modulo % operation divides a number and returns its remainder!

```
number = ?
if (number % ?) == ?
  print(f'{number} is an even number!')
```

Workshop Review Exercise

- a variable named **number**

- an if statement that prints out '**X is an even number!**' if the **number** variable is an even number

- an if statement that prints out '**X is an odd number!**' if the **number** variable is an odd number

- run the program three times, with the **number** variable set to 0, 13, and 256, respectively

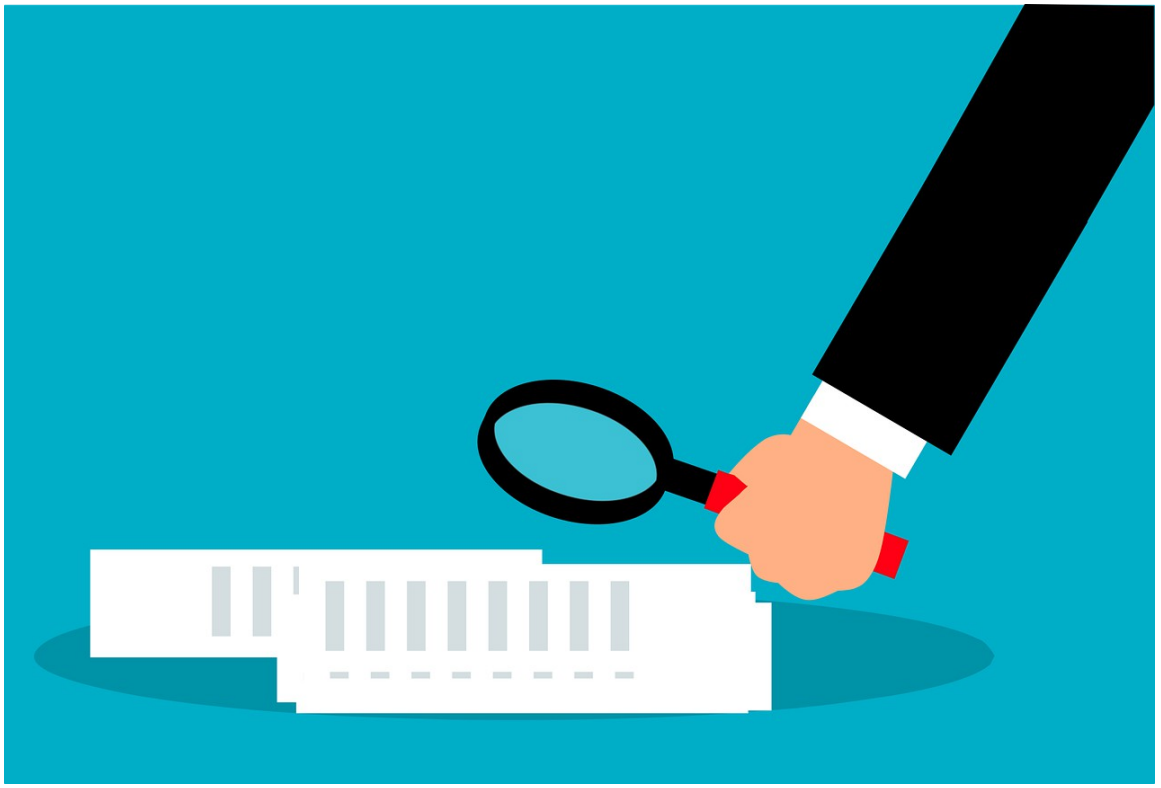
Workshop Review Exercise

A possible solution to this exercise is the following:

```
number = input('Enter a number:')

if (number % 2) == 0:
  print(f'{number} is an even number!')
if (number % 2) == 1:
  print(f'{number} is an odd number!')
```

Summary



Let's review the programming concepts we learned in this workshop:

Conditions

```
password = 'mysecretpassword'

if password == 'mysecretpassword':
    print('password correct')
else:
    print('incorrect password')
```

Conditions are used in programs to determine a program's course of action. They rely on the evaluation of Boolean statements to determine if certain instructions are performed or not.

What's Next?

In the next HackerFrogs Afterschool programming workshop, we'll continue learning Python with the learnpython.org website.

What's Next?

Next workshop topic:

- Programming Loops

Extra Credit

Looking for more study material on this workshop's topics?

See this video's description for links to supplemental documents and exercises!

Until Next Time, HackerFrogs!