3 Days Training on Python3

Day 1 : Module 2

Muhammad Saufy Rohmad

Module 1 (90 minutes)

Objectives

1. Python Numbers, Booleans and None
2. Python if/else
2. Python iteration/looping

1. Python Numbers, Boolean and None

- Integers

```
x = 1
print(x)
print(type(x))
x = 100000000000000000000000000000000000000000000000000001
print(x)
print(type(x))
```

1. Python Numbers, Boolean and None(2)

- Converting to Integers

```
total = int('100')


age = int(input('Please enter your age:'))
print(type(age))
print(age)
```

1. Python Numbers, Boolean and None(3)

- Floating Point Numbers. Real numbers, or floating point numbers, are represented in Python using the IEEE 754 double-precision binary floating-point number format.

  exchange_rate = 1.83

  print(exchange_rate)

  print(type(exchange_rate))

1. Python Numbers, Boolean and None(4)

- Converting to floats.

```
int_value = 1
string_value = '1.5'
float_value = float(int_value)
print('int value as a float:', float_value)
print(type(float_value))
float_value = float(string_value)
print('string value as a float:', float_value)
print(type(float_value))
```

1. Python Numbers, Boolean and None(5)

- Converting to floats from string.

  exchange_rate = float(input("Please enter the exchange rate to use: "))

  print(exchange_rate)

  print(type(exchange_rate))

1. Python Numbers, Boolean and None(6)

- Boolean Values.

```
all_ok = True
print(all_ok)
all_ok = False
print(all_ok)
print(type(all_ok))
```

1. Python Numbers, Boolean and None(7)

- Boolean Values.

  print(int(True))

  print(int(False))

  print(bool(1))

  print(bool(0))

1. Python Numbers, Boolean and None(9)

- Arithmetic Operators

Operator	Description	Example	Equivalent
`+=`	Add the value to the left-hand variable	`x += 2`	`x = x + 2`
`−=`	Subtract the value from the left-hand variable	`x −= 2`	`x = x – 2`
`*=`	Multiple the left-hand variable by the value	`x *= 2`	`x = x * 2`
`/=`	Divide the variable value by the right-hand value	`x /= 2`	`x = x/2`
`//=`	Use integer division to divide the variable's value by the right-hand value	`x //= 2`	`x = x//2`
`%=`	Use the modulus (remainder) operator to apply the right-hand value to the variable	`x %= 2`	`x = x % 2`
`**=`	Apply the power of operator to raise the variable's value by the value supplied	`x **= 3`	`x = x ** 3`

1. Python Numbers, Boolean and None(9)

- Integer Operators

```
home = 10
away = 15
print(home + away)
print(type(home + away))
print(10 * 4)
print(type(10*4))
goals_for = 10
goals_against = 7
print(goals_for - goals_against)
print(type(goals_for - goals_against))
```

1. Python Numbers, Boolean and None(10)

- Integer Operators

  print(100 / 20)

  print(type(100 / 20))

1. Python Numbers, Boolean and None(11)

- Floating Point Number Operators

print(2.3 + 1.5)

print(1.5 / 2.3)

print(1.5 * 2.3)

print(2.3 – 1.5)

print(1.5 – 2.3)

1. Python Numbers, Boolean and None(12)

- Any operation that involves both integers and floating point numbers will always produce a floating point number.

  i = 3 * 0.1

  print(i)

1. Python Numbers, Boolean and None(13)

- Assignment Operator

  x = 0

  x += 1

  # has the same behaviour as x = x + 1

1. Python Numbers, Boolean and None(14)

- None Value

```
winner = None
print('winner:', winner)
print('winner is None:', winner is None)
print('winner is not None:', winner is not None)
print(type(winner))
print('Set winner to True')
winner = True
print('winner:', winner)
print('winner is None:', winner is None)
print('winner is not None:', winner is not None)
print(type(winner))
```

2. Flow of Control Using if statements

- Comparison Operators

Operator	Description	Example
==	Tests if two values are equal	3 == 3
!=	Tests that two values are *not* equal to each other	2 != 3
<	Tests to see if the left-hand value is less than the right-hand value	2 < 3
>	Tests if the left-hand value is greater than the right-hand value	3 > 2
<=	Tests if the left-hand value is less than *or* equal to the right-hand value	3 <= 4
>=	Tests if the left-hand value is greater than or equal to the right-hand value	5 >= 4

2. Flow of Control Using if statements(2)

- Logical Operators

Operator	Description	Example
`and`	Returns True if both left and right are true	`(3 < 4) and (5 > 4)`
`or`	Returns two if either the left or the right is truce	`(3 < 4) or (3 > 5)`
`not`	Returns true if the value being tested is False	`not 3 < 2`

2. Flow of Control Using if statements(3)

- The if statement

```
num = int(input('Enter a number: '))
if num < 0:
print(num, 'is negative')
```

2. Flow of Control Using if statements(4)

- The if statement

```
num = int(input('Enter another number: '))
if num > 0:
    print(num, 'is positive')
    print(num, 'squared is ', num * num)
```

2. Flow of Control Using if statements(5)

- Else in an if statement

```
num = int(input('Enter yet another number: '))
if num < 0:
    print('Its negative')
else:
    print('Its not negative')
```

2. Flow of Control Using if statements(6)

- The use of elif

```
savings = float(input("Enter how much you have in savings: "))
if savings == 0:
    print("Sorry no savings")
elif savings < 500:
    print('Well done')
elif savings < 1000:
    print('Thats a tidy sum')
elif savings < 10000:
    print('Welcome Sir!')
else:
    print('Thank you')
```

2. Flow of Control Using if statements(8)

- Nesting if statement

```
snowing = True
temp = -1
if temp < 0:
    print('It is freezing')
    if snowing:
        print('Put on boots')
        print('Time for Hot Chocolate')
print('Bye')
```

2. Flow of Control Using if statements(6)

- If expression

```
age = 15
status = None
if (age > 12) and age < 20:
    status = 'teenager'
else:
    status = 'not teenager'
```

3. Iteration / Looping

- While loop

```
count = 0
print('Starting')
while count < 10:
    print(count, ' ', end='') # part of the while loop
    count += 1 # also part of the while loop
print() # not part of the while loop
print('Done')
```

3. Iteration / Looping(2)

- For loop

```
# Loop over a set of values in a range
print('Print out values in a range')
for i in range(0, 10):
    print(i, ' ', end='')
print()
print('Done')
```

3. Iteration / Looping(3)

- Break loop statement

```
print('Only print code if all iterations completed')
num = int(input('Enter a number to check for: '))
for i in range(0, 6):
    if i == num:
        break
    print(i, ' ', end='')
print('Done')
```

3. Iteration / Looping(4)

- Continue loop statement

```
for i in range(0, 10):
    print(i, ' ', end='')
    if i % 2 == 1:
        continue
    print('hey its an even number')
    print('we love even numbers')
print('Done')
```

3. Iteration / Looping(5)

- For Loop with else

```
# Only print code if all iterations completed over a list
print('Only print code if all iterations completed')
num = int(input('Enter a number to check for: '))
for i in range(0, 6):
    if i == num:
        break
    print(i, ' ', end='')
else:
    print()
    print('All iterations successful')
```

3. Iteration / Looping(5)

- Dice roll game

```
import random
MIN = 1
MAX = 6
roll_again = 'y'
while roll_again == 'y':
    print('Rolling the dices...')
    print('The values are....')
    dice1 = random.randint(MIN, MAX)
    print(dice1)
    dice2 = random.randint(MIN, MAX)
    print(dice2)
    roll_again = input('Roll the dices again? (y / n): ')
```