Numbers and If statement

Data Structure, second stage, computer department, college of science for women.

Dr. Amer Al-Mahdawi.

Types of Numbers

- **Integers**
- All integer values, no matter how big or small are represented by the integral (or int) type in Python 3. For example:
- `x = 1`
- `print(x)`
- `print(type(x))`
- `x =`
- 10000000000000000000000000000000000000000000000000000000000000001
- `print(x)`
- `print(type(x))`

Types of Numbers

- If this code is run, then the output will show that both numbers are of type int:
- `1`
- `<class 'int'>`
- 10000000000000000000000000000000000000000000000000000000001
- `<class 'int'>`

• Converting to Ints

- It is possible to convert another type into an integer using the int() function. For example, if we want to convert a string into an int (assuming the string contains an integer number) then we can do this using the int() function. For example:
- `total = int('100')`
- This can be useful when used with the input() function. The input() function always returns a string. If we want to ask the user to input an integer number, then we will need to convert the string returned from the input() function into an int. We can do this by wrapping the call to the input() function in a call to the int() function, for example:

Converting to Ints

- `age = int(input('Please enter your age:'))`
- `print(type(age))`
- `print(age)`
- Running this gives:
- `Please enter your age: 21`
- `<class 'int'>`
- `21`
- The int() function can also be used to convert a floating-point number into an
- int, for example:
- `i = int(1.0)`

Floating Point Numbers

- The type used to represent a floating-point number is called float. Python represents floating point numbers using a decimal point to separate the whole part from the fractional part of the number, for example:
- `exchange_rate = 1.83`
- `print(exchange_rate)`
- `print(type(exchange_rate))`
- This produces output indicating that we are storing the number 1.83 as a floating
- point number:
- `1.83`
- `<class 'float'>`

Converting to Floats

- As with integers it is possible to convert other types such as an int or a string into a
- float. This is done using the float() function:
- 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))
- The output from this code snippet is:
- int value as a float: 1.0
- <class 'float'>
- string value as a float: 1.5
- <class 'float'>

Converting an Input String into a Floating Point Number

- As we have seen the input() function returns a string; what happens if we want the user to input a floating point (or real) number? As we have seen above, a string can be converted into a floating-point number using the float() function and therefore we can use this approach to convert an input from the user into a float:
- `exchange_rate = float(input("Please enter the exchange rate to use: "))`
- `print(exchange_rate)`
- `print(type(exchange_rate))`
- Using this we can input the string 1.83 and convert it to a floating-point number:
- `Please enter the exchange rate to use: 1.83`
- `1.83`
- `<class 'float'>`

Boolean Values

- Python supports another type called Boolean; a Boolean type can only be one of True or False (and nothing else). Note that these values are True (with a capital T) and False (with a capital F); true and false in Python are not the same thing and have no meaning on their own
- The equivalent of the int or float class for Booleans is bool. The following example illustrates storing the two Boolean values into a variable all_ok:
- all_ok = True
- print(all_ok)
- all_ok = False
- print(all_ok)
- print(type(all_ok))
- The output of this is
- True
- False
- <class 'bool'>

Boolean Values

- The Boolean type is actually a sub type of integer (but with only the values True and False) so it is easy to translate between the two, using the functions int() and bool() to convert from Booleans to Integers and vice versa. For example:
- print(int(True))
- print(int(False))
- print(bool(1))
- print(bool(0))
- Which produces
- 1
- 0
- True
- False

Arithmetic Operators

Operator	Description	Example
+	Add the left and right values together	`1 + 2`
−	Subtract the right value from the left value	`3 − 2`
*	Multiple the left and right values	`3 * 4`
/	Divide the left value by the right value	`12/3`
//	Integer division (ignore any remainder)	`12//3`
%	Modulus (aka the remainder operator)—only return any remainder	`13%3`
**	Exponent (or power of) operator—with the left value raised to the power of the right	`3 ** 4`

Integer Operations

- `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))`
- The output from this is
- `25`
- `<class 'int’>`
- `40`
- `<class 'int’>`
- `3`
- `<class 'int'>`

Integer Operations

- `print(100 / 20)`
- `print(type(100 / 20))`
- The output is
- `5.0`
- `<class 'float’>`
- `res1 = 3/2`
- `print(res1)`
- `print(type(res1))`
- `The output is:`
- `1.5`
- `<class 'float'>`

Integer Operations

- `res1 = 3//2`
- `print(res1)`
- `print(type(res1))`
- which produces
- `1`
- `<class 'int'>`
- `Ex:`
- `print('Modulus division 4 % 2:', 4 % 2)`
- `print('Modulus division 3 % 2:', 3 % 2)`
- Which produces:
- `Modulus division 4 % 2: 0`
- `Modulus division 3 % 2: 1`

Assignment 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