Operators in python

int213

The modulus operator

- The modulus operator works on integers (and integer expressions) and yields the remainder when the first operand is divided by the second.
- In Python, the modulus operator is a percent sign (%).

Example

- The syntax is the same as for other operators:

```
>>> quotient = 7 / /3
>>> print (quotient)
        2
>>> remainder = 7 % 3
>>> print (remainder)
        1
```

- So 7 divided by 3 is 2 with 1 left over.

Uses

- Check whether one number is divisible by another if x % y is zero, then x is divisible by y.
- you can extract the right-most digit or digits from a number.
- For example,

x % 10 yields the right-most digit of x (in base 10). Similarly x % 100 yields the last two digits.

Boolean expressions

- A Boolean expression is an expression that is either true or false.
- One way to write a Boolean expression is to use the operator ==, which compares two values and produces a Boolean value:

```
>>> 5 == 5
        True
>>> 5 == 6
        False
```

- True and False are special values that are built into Python.

Comparison Operators

- x != y # x is not equal to y
- x > y # x is greater than y
- x < y # x is less than y
- x >= y # x is greater than or equal to y
- x <= y # x is less than or equal to y

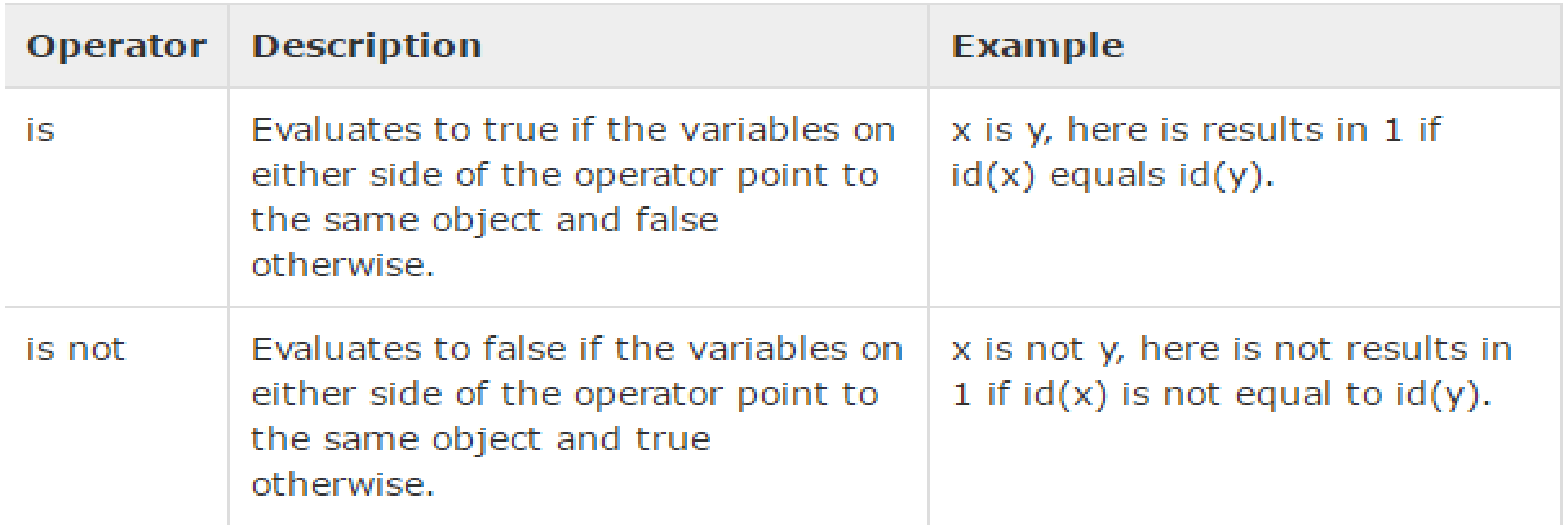
NOTE: “= is an assignment operator and == is a comparison operator”. Also, there is no such thing as =< or =>.

Logical operators

- There are three logical operators:
  - and,
  - or
  - not
- For example, x > 0 and x < 10 is true only if x is greater than 0 and less than 10.
- n%2 == 0 or n%3 == 0
- not(x > y) is true if (x > y) is false, that is, if x is less than or equal to y.

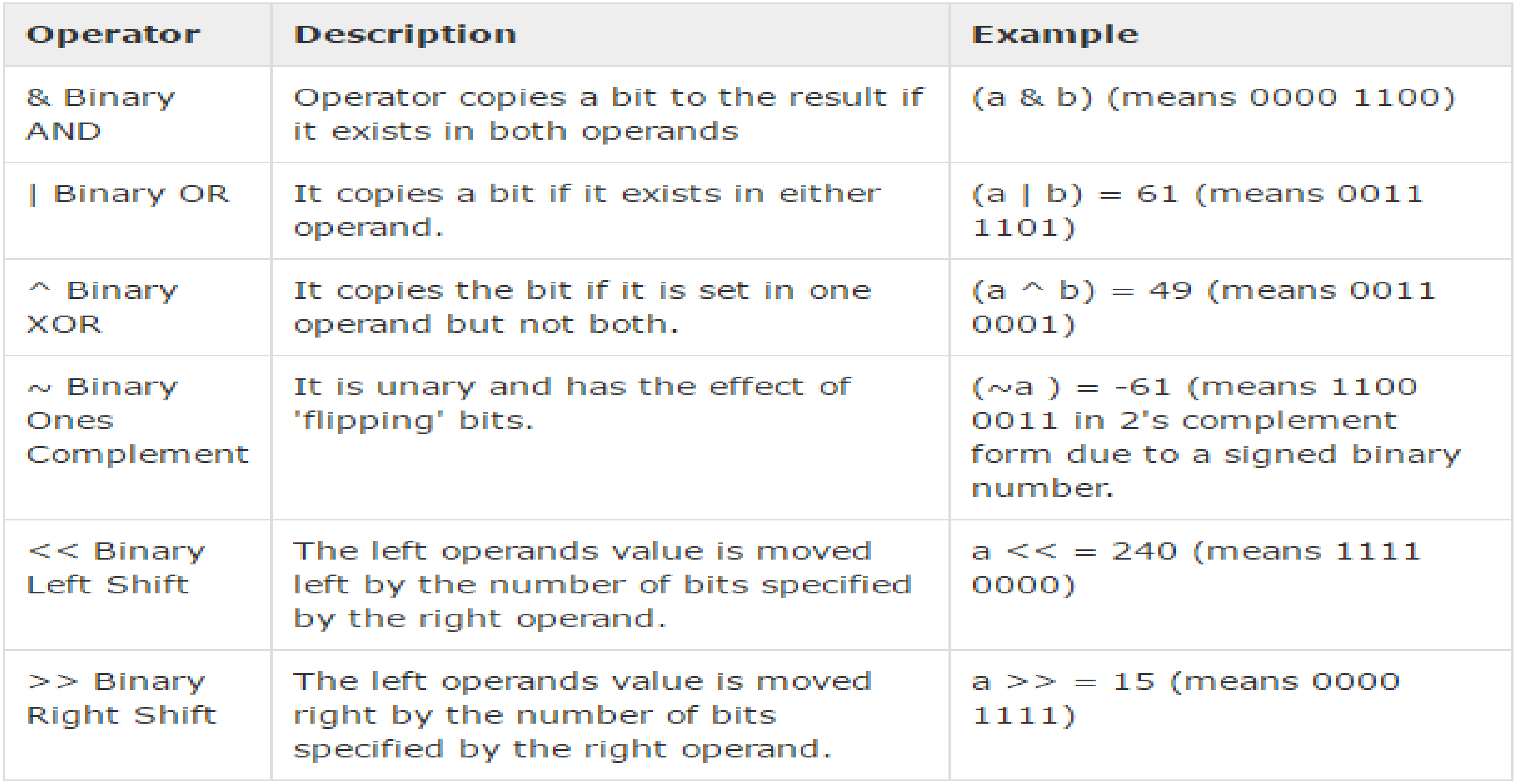
Identity operators

- Identity operators compare the memory locations of two objects. There are two Identity operators as explained below



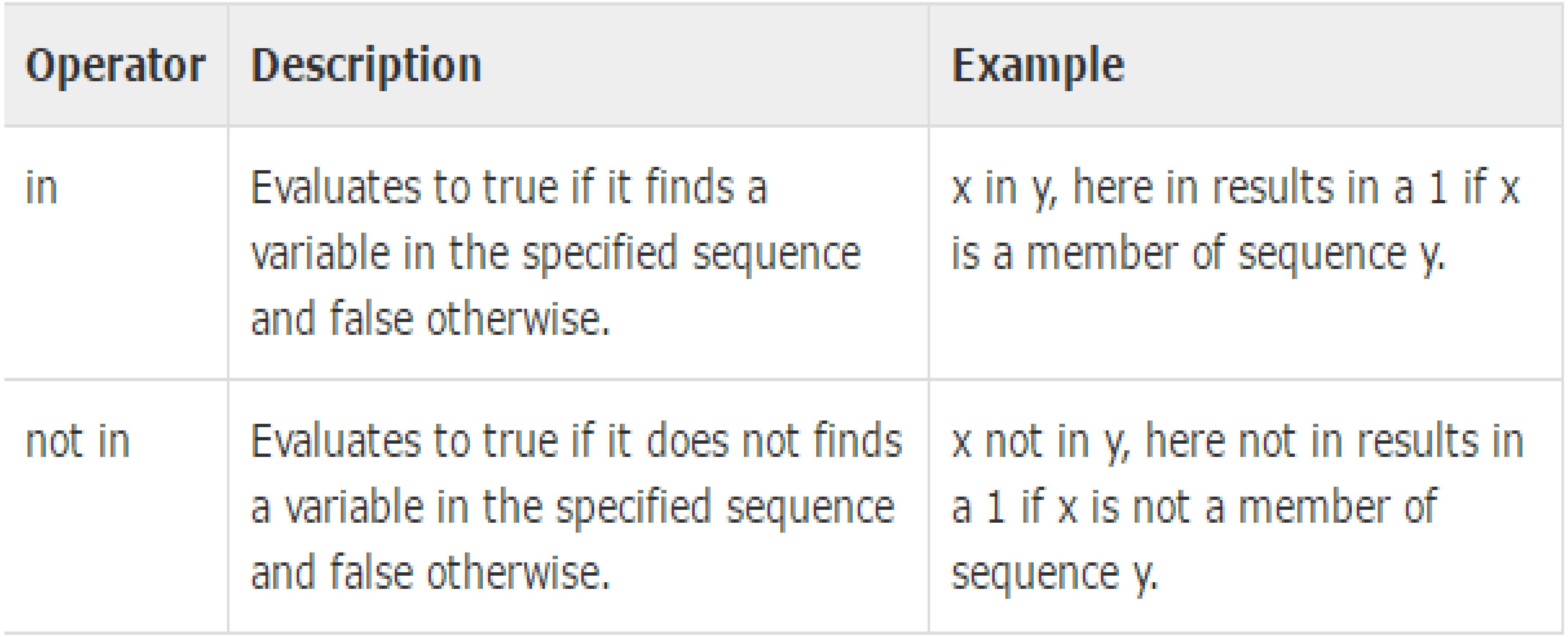
Operator	Description	Example
is	Evaluates to true if the variables on either side of the operator point to the same object and false otherwise.	x is y, here is results in 1 if id(x) equals id(y).
is not	Evaluates to false if the variables on either side of the operator point to the same object and true otherwise.	x is not y, here is not results in 1 if id(x) is not equal to id(y).

Bitwise Operators



Operator	Description	Example
& Binary AND	Operator copies a bit to the result if it exists in both operands	(a & b) (means 0000 1100)
| Binary OR	It copies a bit if it exists in either operand.	(a | b) = 61 (means 0011 1101)
^ Binary XOR	It copies the bit if it is set in one operand but not both.	(a ^ b) = 49 (means 0011 0001)
~ Binary Ones Complement	It is unary and has the effect of 'flipping' bits.	(~a) = -61 (means 1100 0011 in 2's complement form due to a signed binary number.
<< Binary Left Shift	The left operands value is moved left by the number of bits specified by the right operand.	a << = 240 (means 1111 0000)
>> Binary Right Shift	The left operands value is moved right by the number of bits specified by the right operand.	a >> = 15 (means 0000 1111)

Membership Operators



Operator	Description	Example
in	Evaluates to true if it finds a variable in the specified sequence and false otherwise.	x in y, here in results in a 1 if x is a member of sequence y.
not in	Evaluates to true if it does not finds a variable in the specified sequence and false otherwise.	x not in y, here not in results in a 1 if x is not a member of sequence y.

Continue…

- Any nonzero number is interpreted as “true."

```
>>> x = 5
>>> x and 1
          1
>>> y = 0
>>> y and 1
     0
```

Keyboard Input

- input(): built in function to get data from keyboard.
- Takes data in the form of **string.**
- Eg:

```
>>> input1 = input ()
What are you waiting for?
>>> print (input1)
What are you waiting for?
```

- Before calling input, it is a good idea to print a message telling the user what to input. This message is called a **prompt.**
- A prompt can be supplied as an argument to input.

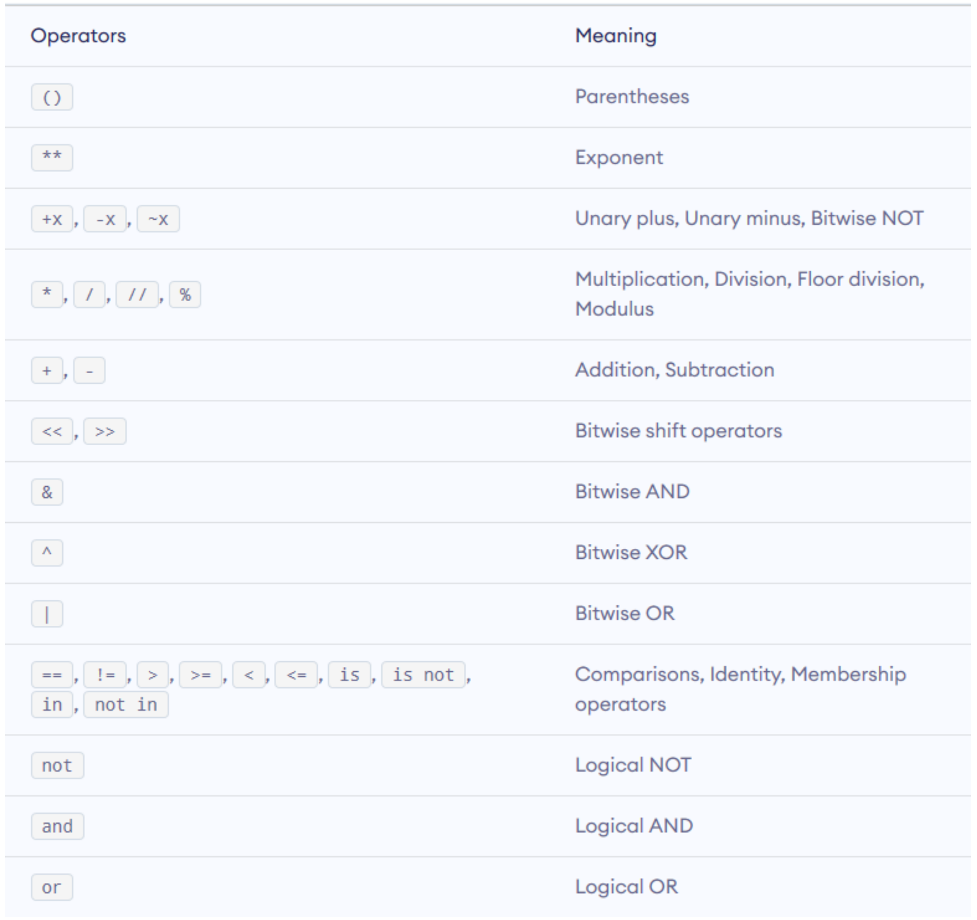
- Eg:

```
>>> name = input ("What...is your name? ")
What...is your name? Arthur, King of the Britons!
>>> print(name)
Arthur, King of the Britons!
```

- If we expect the response to be an integer, then type conversion needs to be done.
- Eg:

```
prompt = "What is the airspeed velocity of an unladen swallow?"
speed =int(input(prompt))
```

The operator precedence in Python is listed in the following table. It is in descending order (upper group has higher precedence than the lower ones).



Operators	Meaning
`()`	Parentheses
`**`	Exponent
`+x`, `-x`, `~x`	Unary plus, Unary minus, Bitwise NOT
`*`, `/`, `//`, `%`	Multiplication, Division, Floor division, Modulus
`+`, `-`	Addition, Subtraction
`<<`, `>>`	Bitwise shift operators
`&`	Bitwise AND
`^`	Bitwise XOR
`|`	Bitwise OR
`==`, `!=`, `>`, `>=`, `<`, `<=`, `is`, `is not`, `in`, `not in`	Comparisons, Identity, Membership operators
`not`	Logical NOT
`and`	Logical AND
`or`	Logical OR