Overview

1. Python/IDE installation
2. Data types in Python
3. Assigning the value into variable
4. Mathematic operators
5. Input/print function

NAZARBAYEV UNIVERSITY

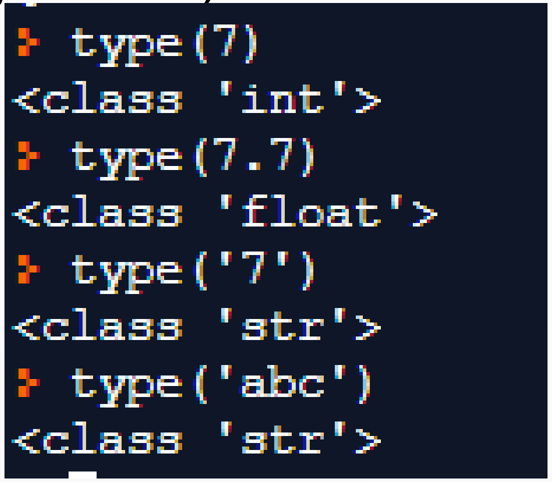
Data Types

Integer
- Whole numbers (4, 1000, -400, 10)
- Binary(0b10), octal(0o10), Hexadecimal(0x10)

Float
- Numbers with decimal points, 1.15, 0.4

String
- A text. For example: “word”, “17”, ,“hello world”,
- Blank is also string (it has length)
- The so-called *empty string*, "", has no characters (its length is zero).

Boolean - Truth values (True and False).
List
Tuple
dict



```
> type(7)
<class 'int'>
> type(7.7)
<class 'float'>
> type('7')
<class 'str'>
> type('abc')
<class 'str'>
```

Be careful: 17 is a number, while ‘17’ is a string!

5, -5, 5.8, 10.2, ‘54’, ‘world’, ‘-485.0’

Variables

- Variables are used to store the data and retrieve it
- Every variable has a value (which, in Python, *can* be undefined) and a type (partly hidden in Python).
- Programming languages allow you to define variables.
- In Python, you use = to assign a value to a variable.

```
>>> r = 10
```

```
>>> r = ‘hello’
```

```
>>> r = 3.32
```

Assignment vs. equations

Assignment vs. equations

In algebra,

$$t = t + 10$$

doesn’t make sense unless you believe

$$t - t = 10$$

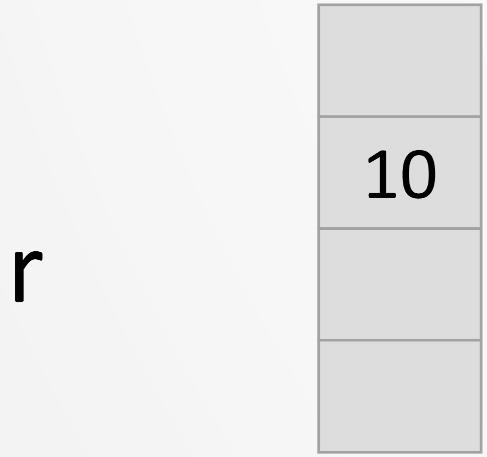
In Python,

```
t = t + 10
```

means add 10 to the value of t and store the result in t.

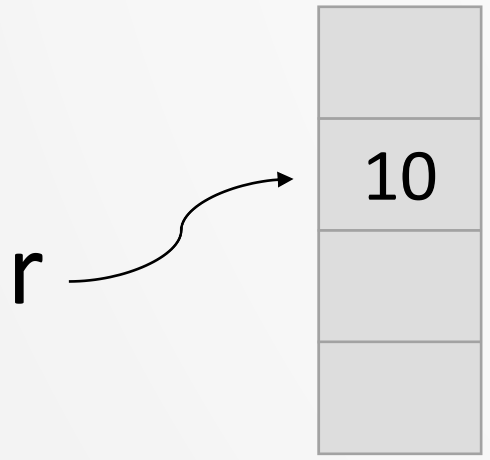
Assignment statement

```
>>> r = 10
```



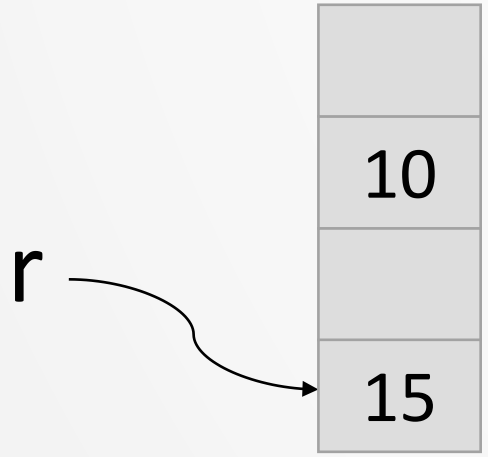
Assignment statement

```
>>> r = 10
```



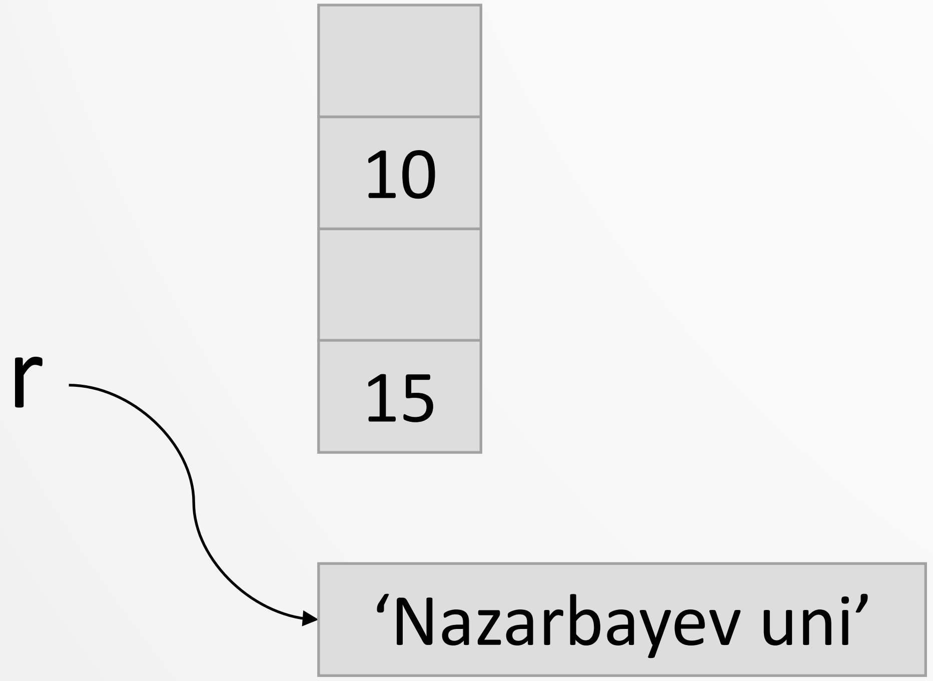
Assignment statement

```
>>> r = 10
>>> r = 15
```



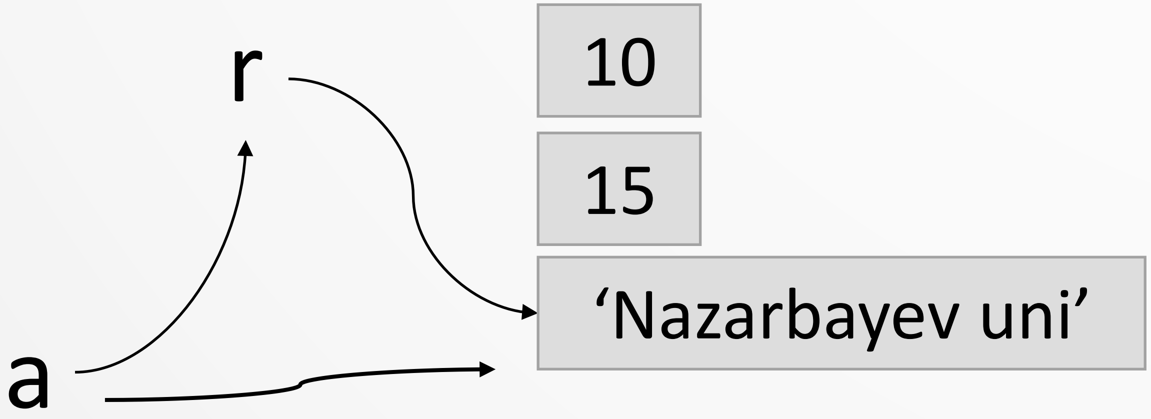
Assignment statement

```
>>> r = 10
>>> r = 15
>>> r = ‘Nazarbayev uni’
```



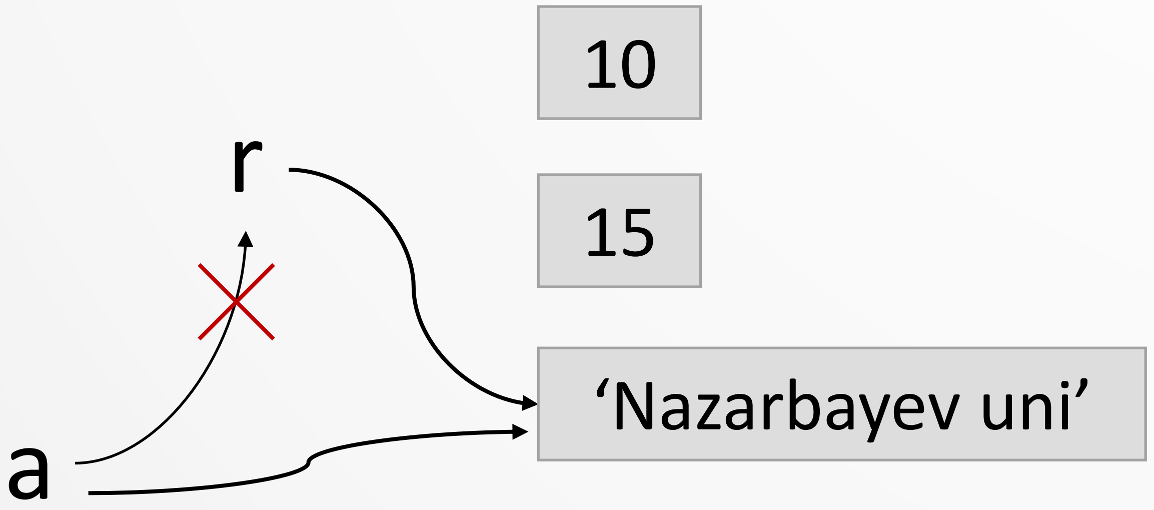
Assignment statement

```
>>> r = 10
>>> r = 15
>>> r = 'Nazarbayev uni'
>>> a = r
>>> r = 3
```



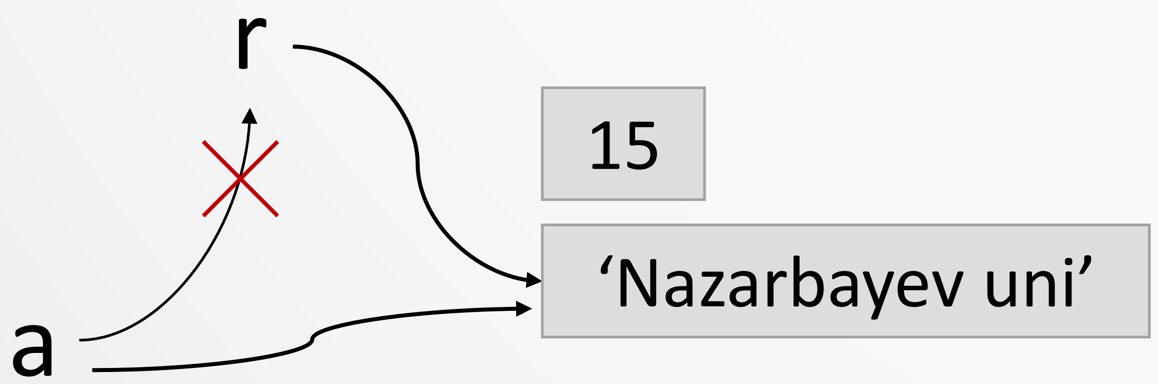
Assignment statement

```
>>> r = 10
>>> r = 15
>>> r = 'Nazarbayev uni'
>>> a = r
```



Assignment statement

```
>>> r = 10                      -> Integer
>>> r = 15                      -> Integer
>>> r = ‘Nazarbayev uni’        -> String
>>> a = r                       -> String
>>> r = 3.5                     -> Float
>>> print(a)
Nazarbayev uni
```



Swap two variables

```
>>> a = 3
>>> b = 10
```

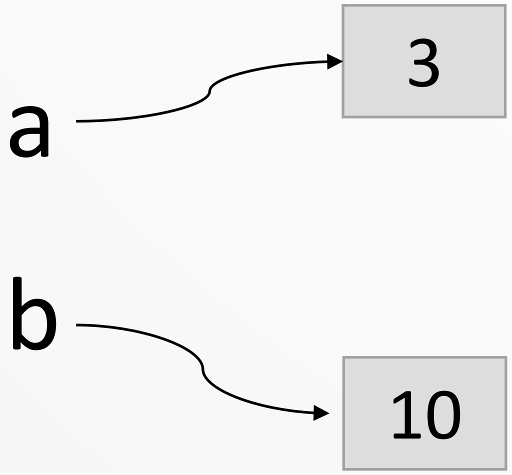
Can you swap those two variable
(i.e., b = 3, a = 10)

Swap two variables

```
>>> a = 3
>>> b = 10
```

```
>>> a = 10
>>> b = 3
```

Can you swap those two variable
(i.e., b = 3, a = 10)



NAZARBAYEV UNIVERSITY

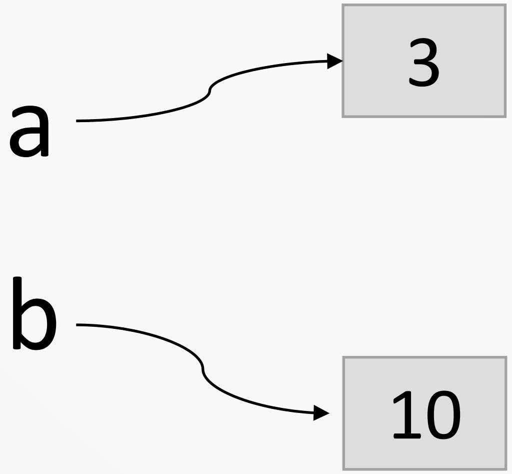
Swap two variables

```
>>> a = 3
>>> b = 10
```

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Can you swap those two variable
(i.e., b = 3, a = 10)

Let’s consider we don’t know what values
were assigned to variables
(a = num1, b = num2)



Swap two variables

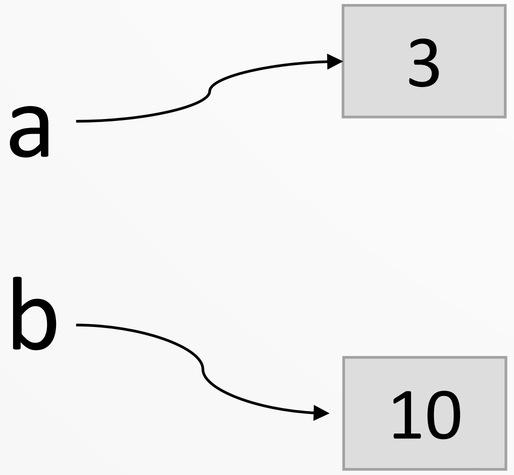
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let's consider we don't know what values
were assigned to variables
(a = num1, b = num2)

```
>>> a = b
>>> b = a
```



NAZARBAYEV UNIVERSITY

Swap two variables

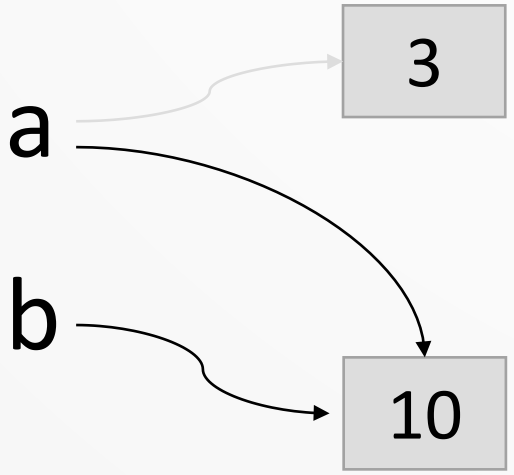
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let's consider we don't know what values were assigned to variables
(a = num1, b = num2)

```
>>> a = b
>>>
```



Swap two variables

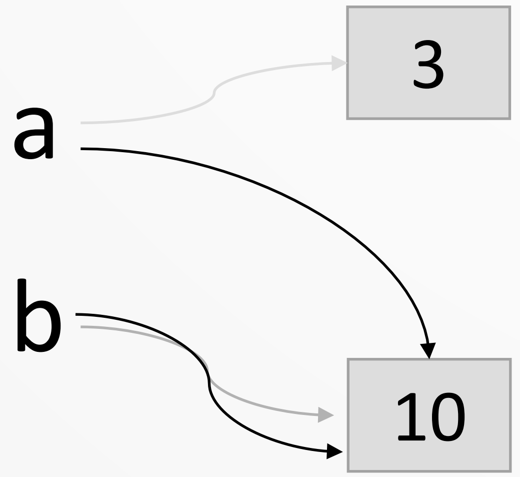
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let's consider we don't know what values were assigned to variables
(a = num1, b = num2)

```
>>> a = b
>>> b = a
```



NAZARBAYEV UNIVERSITY

Swap two variables

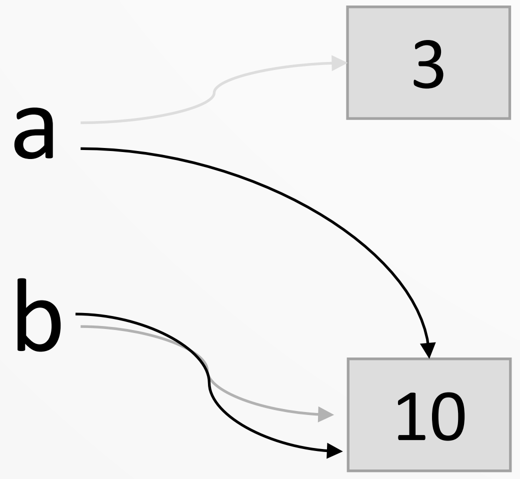
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let's consider we don't know what values were assigned to variables
(a = num1, b = num2)

```
>>> a = b
>>> b = a
```



a = 10, b= 10

Swap two variables

```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let’s consider we don’t know what values were assigned to variables
(a = num1, b = num2)

```
>>> temp = a
>>> a = b
>>> b = temp
```

NAZARBAYEV UNIVERSITY

Swap two variables

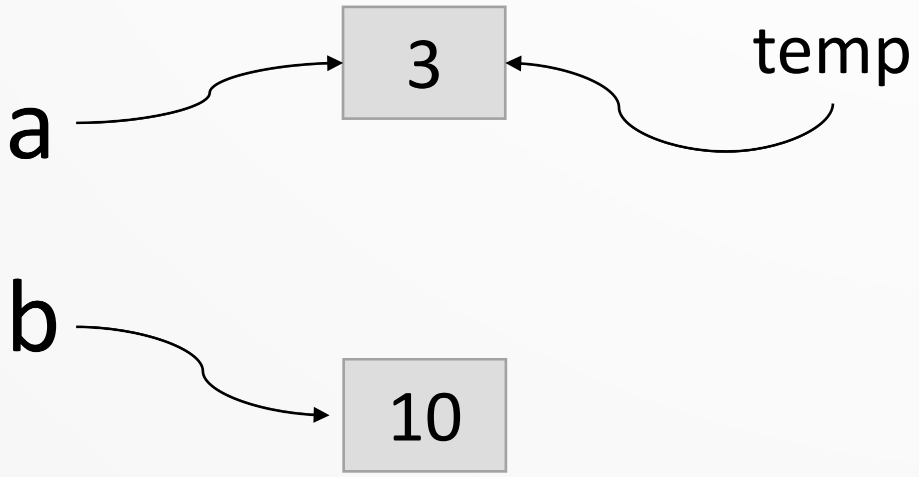
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let’s consider we don’t know what values were assigned to variables
(a = num1, b = num2)

```
>>> temp = a
```



Swap two variables

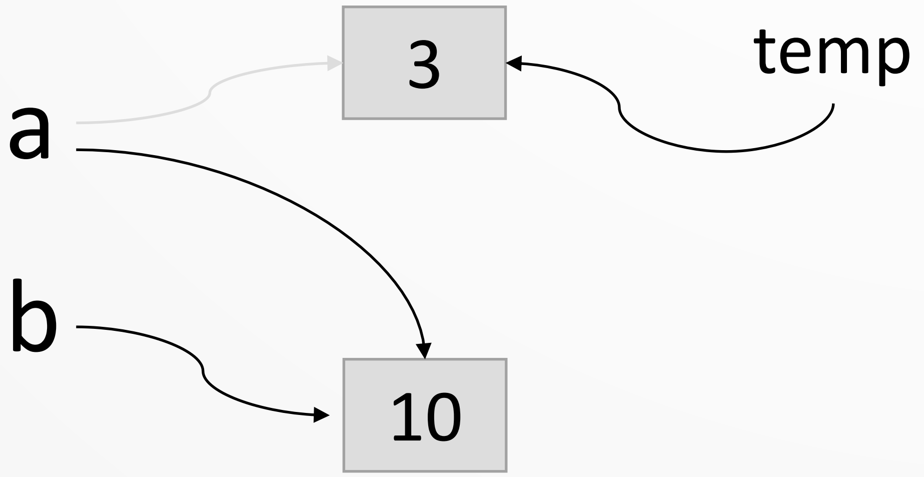
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let’s consider we don’t know what values were assigned to variables
(a = num1, b = num2)

```
>>> temp = a
>>> a = b
```



Swap two variables

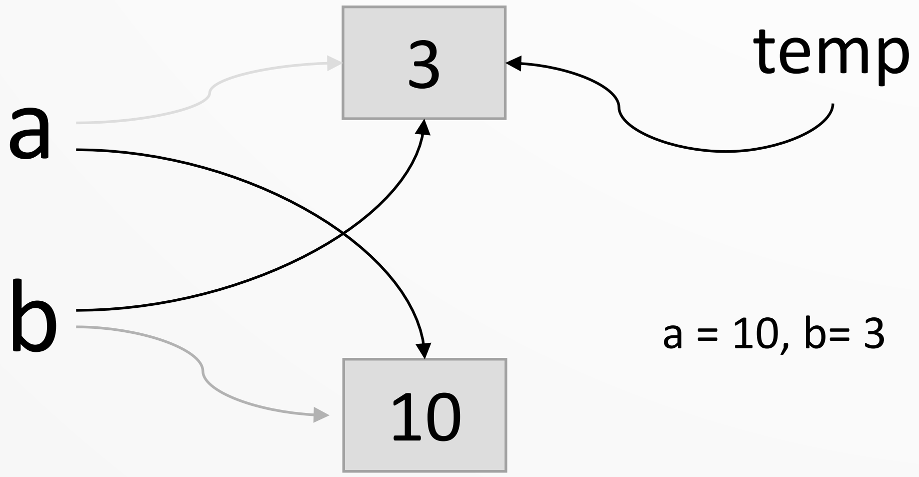
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let's consider we don't know what values were assigned to variables
(a = num1, b = num2)

```
>>> temp = a
>>> a = b
>>> b = temp
```



a = 10, b= 3

Swap two variables

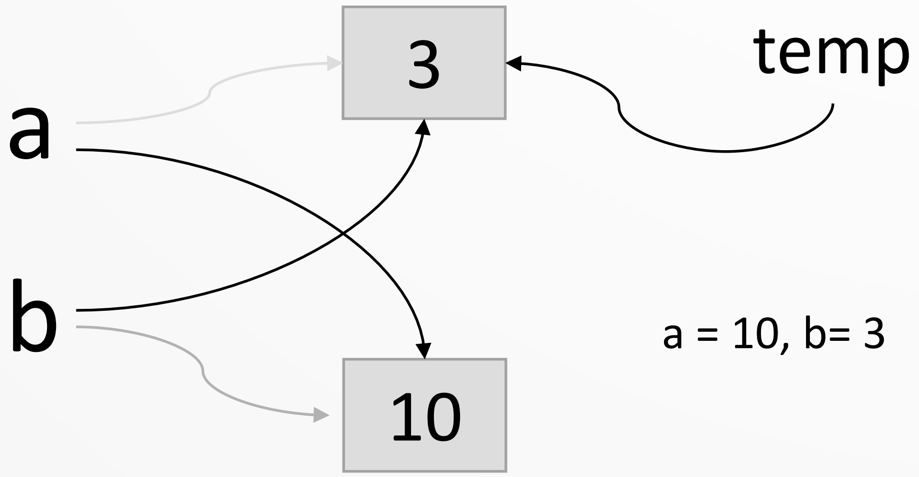
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let's consider we don't know what values were assigned to variables
(a = num1, b = num2)

```
>>> temp = a
>>> a = b
>>> b = temp
```



a = 10, b= 3

NAZARBAYEV UNIVERSITY

Swap two variables

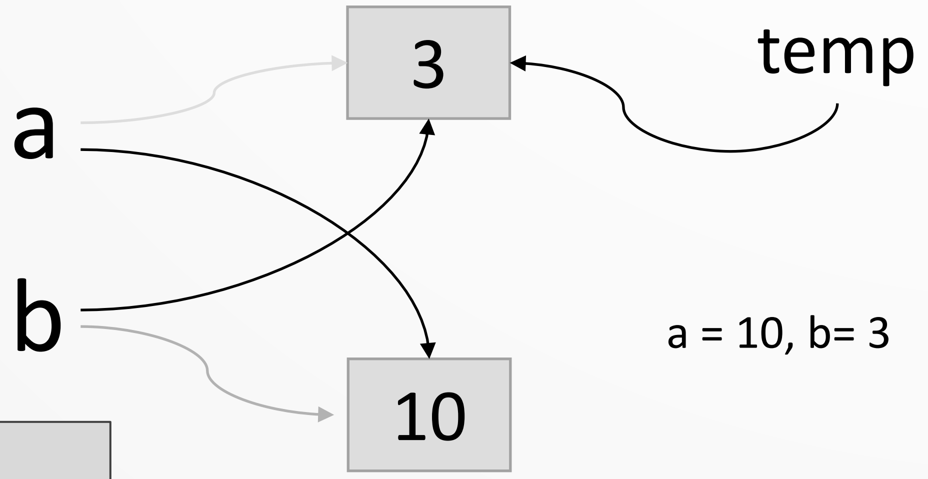
```
>>> a = 3
>>> b = 10
```

Can you swap those two variable
(i.e., b = 3, a = 10)

```
~~>>> a = 10~~
~~>>> b = 3~~
```

Let's consider we don't know what values were assigned to variables
(a = num1, b = num2)

```
>>> temp = a
>>> a = b
>>> b = temp
```



```
>>> [a, b] = [b, a]
```

Operators

Assume variable **a** holds the value 10 and variable **b** holds the value 20, then

Operator		Description	Example	Shortcut
+	Addition	Adds values on either side of the operator .	a + b = 30	x += y
-	Subtraction	Subtracts right hand operand from left ha nd operand.	a – b = -10	x -= y
*	Multiplication	Multiplies values on either side of the ope rator	a * b = 200	x *= y
/	Division	Divides left hand operand by right hand o perand	b / a = 2.0	x /= y
%	Modulus	Divides left hand operand by right hand o perand and returns **remainder**	b % a = 1	x %= y
**	Exponentiation	Performs exponential (power) calculation on operators	a**b =10 to the power 20	x **= y
//	Integer division	Floor Division - The division of operands where the result is the quotient in which t he digits after the decimal point are remo ved.	9//2 = 4 and 9.0 //2.0 = 4.0, -11// 3 = -4, -11.0//3 = -4.0	x //= y

https://www.tutorialspoint.com/python3/arithmetic_operators_example.htm

NAZARBAYEV UNIVERSITY

Mathematic operation

```
>>> 123
123
>>> +123
123
>>> -123
-123
>>> 5 + 3 -1
7
>>> 5      +   3      -1
7
>>> 9/5
1.8
>>> 9//5
1
```

A sequence of digits specifies a positive integer

To specify a negative integer, insert '–' before the digit

Addition and subtraction work as you'd expected

You're not required to have a space between-

/ carries out floating-point (decimal) division

// performs integer division

Mathematic operation

```
>>> a = 95
95
>>> a-3
92
>>> a
95
>>> a = a – 3
>>> a
92
```

You can mix literal integers and variables

We didn’t assign the result to variable ‘a’

We reassign the variable ‘a’ to new value

Order of operator

- The order of evaluation in programming language is the same as an arithmetic expression

- Exponentiation & negation comes before multiplication & division which in turn come before addition & subtraction

This:		Is the same as:
A + B * C	$\longrightarrow$	A + (B * C)
-A**2/4	$\longrightarrow$	-(A**2)/4
A*B/C*D	$\longrightarrow$	((A*B)/C)*D

Priority of operator

```
>>> 2 + 3 * 4
14
>>> 2 + (3 * 4)
14
>>> (2 + 4) * 4
20
```

What would you get if you typed the following?

If you do the addition first, 2 + 3 is 5, and 5 * 4 is 20.
But if you do the multiplication first, 3 * 4 is 12, and 2 * 12 is 14.

Common rule in most programming languages

Just add parentheses to clarify the code for you and others

Operators	Meaning
()	Parentheses
**	Exponent
+x, -x, ~x	Unary plus, Unary minus, Bit wise NOT
*, /, //, %	Multiplication, Division, Floor division, Modulus
+, -	Addition, Subtraction
<<, >>	Bitwise shift operators

&	Bitwise AND
^	Bitwise XOR
|	Bitwise OR
==, !=, >, >=, <, <=, is, is not, in, no t in	Comparisions, Identity, Membership operators
not	Logical NOT
and	Logical AND
or	Logical OR

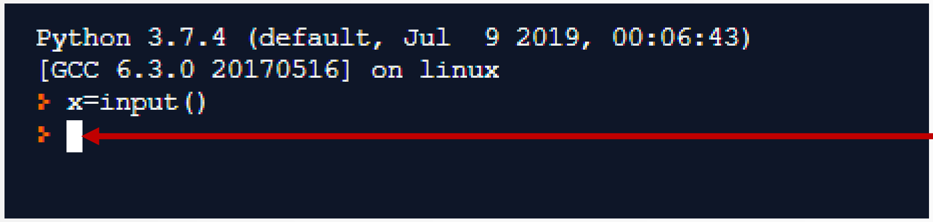
Operator precedence rule in Python

NAZARBAYEV UNIVERSITY

Input

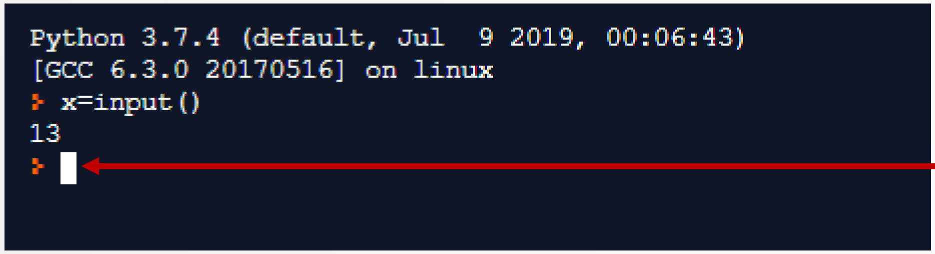
- How do we **input** some data from the user?
- Not surprisingly, using the function input()

```
x = input("Enter any value")
print("The value of x is", x)
```



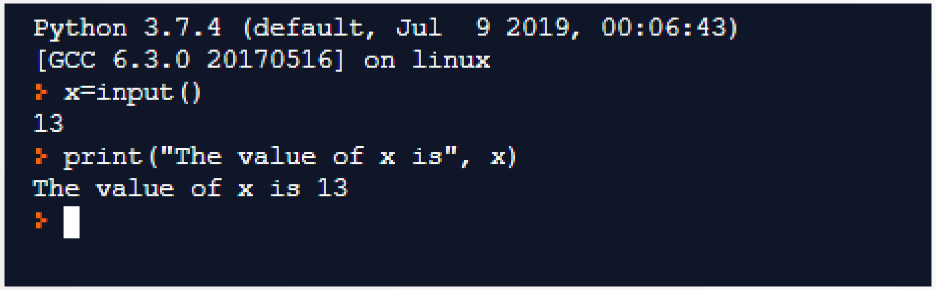
```
Python 3.7.4 (default, Jul  9 2019, 00:06:43)
[GCC 6.3.0 20170516] on linux
› x=input()
›
```

The system is waiting your input



```
Python 3.7.4 (default, Jul  9 2019, 00:06:43)
[GCC 6.3.0 20170516] on linux
› x=input()
13
›
```

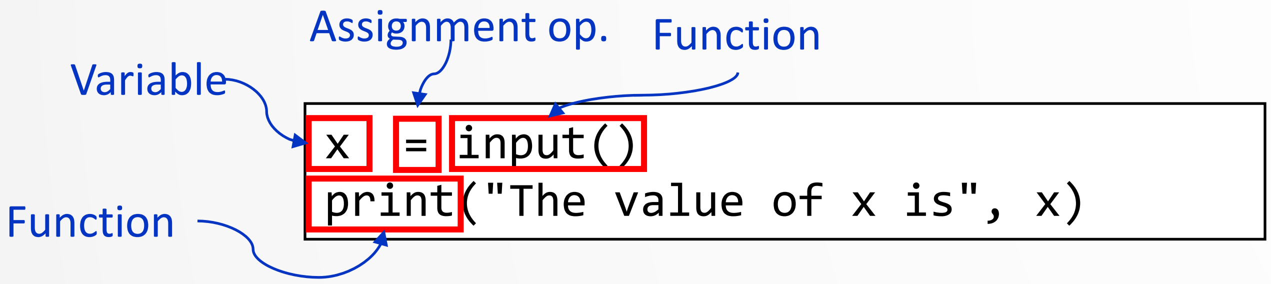
Now the variable x has the input value (i.e., x=13)



```
Python 3.7.4 (default, Jul  9 2019, 00:06:43)
[GCC 6.3.0 20170516] on linux
› x=input()
13
› print("The value of x is", x)
The value of x is 13
›
```

NAZARBAYEV UNIVERSITY

Input



```
x = input()
print("The value of x is", x)
```

- The function input() reads a sequence of characters from the standard input (usually the user's keyboard) and returns it as a **string**.
- That value is then assigned to the variable x (on the left-hand side of the assignment operator =).
- Whatever is on the right-hand side of the assignment = gets computed first. Then the result is assigned to the variable on the left-hand side. When this is done, the next line of code is executed.
- The function print() now outputs its arguments to the standard output (usually the user's screen), in order in which they were given, separated by a single space character. So,
  - First, a string `"The value of x is"` is written out.
  - Then a singe space character is written out.
  - Then the value of `x` is written out (**not** the string "x" itself, because `x` is a variable!).

Concatenation of string variables

```
>>> a = input('Your first name')
>>> b = input('Your Lastname')
>>> print('His name is ', a+b)
His name is MinHoLee
```

Homework

- Ask the user to enter the name, age, and hobby.

```
name = input("Enter your name:")
```

- Print all the variables with proper sentence
  (e.g., your name is ---, and you are --- years old, and your hobby is ---)

Expected output:

```
Enter your name:Min-Ho Lee
Enter your age:34
Enter your favorite hobby:Watching movie
Your name is Min-Ho Lee and you are 34 years old
and your hobby is Watching movie
```

NAZARBAYEV UNIVERSITY