# Python Programming Language

What is python !?

Python is a high-level interpreted programming language. Python's design philosophy emphasizes code readability through the use of clean syntax and significant whitespace. And it makes for a no-brainer decision to use it for both beginner and experienced programmers.

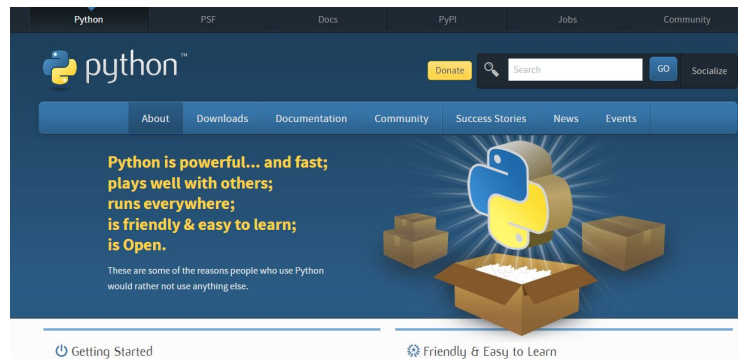In additional, Python is a general-purpose programming language.

Example

- Game
- Data Analytic
- Website
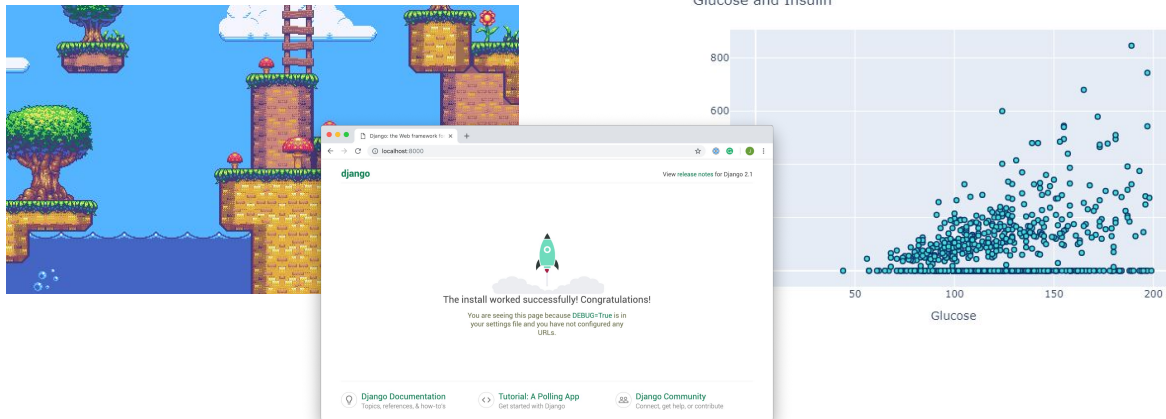
How to install python?

1. Download python in
   https://www.python.org/downloads/
2. Run the installer
3. Customize the Installation
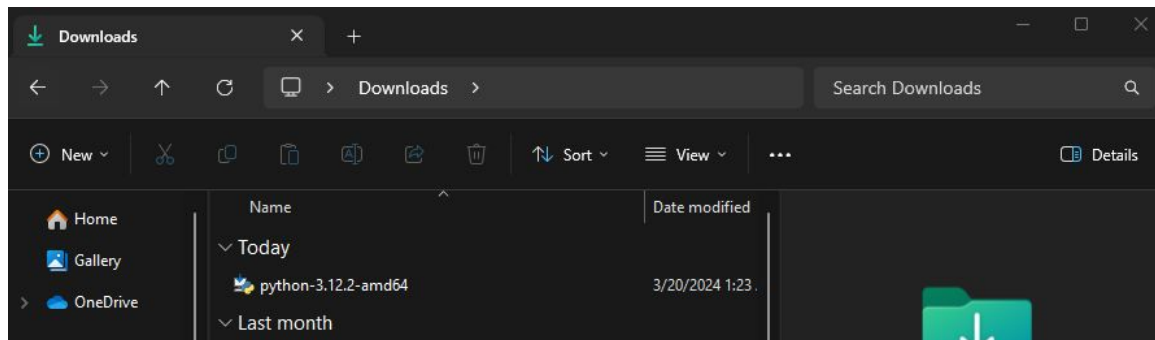4. Install Python
5. Verify the installation

Download Python

Visit the official Python website and download the latest version for Windows. The website will automatically detect your operating system and offer the appropriate installer for your system (32-bit or 64-bit).

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG	Sigstore
Gzipped source tarball	Source release		f6b5226ccba5ae1ca9376aaba0b0f673	26437858	SIG	CRT SIG
XZ compressed source tarball	Source release		a957cffb58a89303b62124896881950b	19893284	SIG	CRT SIG
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later	e038c3d5cee8c5210735a764d3f36f5a	42835777	SIG	CRT SIG
Windows embeddable package (32-bit)	Windows		64853e569d7cb0d1547793000ff9c9b6	9574852	SIG	CRT SIG
Windows embeddable package (64-bit)	Windows		ae7de44ecbe2d3a37dbde3ce669d31b3	10560465	SIG	CRT SIG
Windows embeddable package (ARM64)	Windows		747090b80a52e8bbcb5cb65f78fee575	9780864	SIG	CRT SIG
Windows installer (32-bit)	Windows		2123016702bbb45688baedc3695852f4	24155760	SIG	CRT SIG
Windows installer (64-bit)	Windows	Recommended	4331ca54d9eacdbe6e97d6ea63526e57	25325400	SIG	CRT SIG
Windows installer (ARM64)	Windows	Experimental	040ab03501a65cc26bd340323bb1972e	24451768	SIG	CRT SIG

Run installer

Locate the downloaded installer file (usually in your Downloads folder) and double-click on it to run the installation process. You may be prompted by the User Account Control (UAC) to allow the installation. Click Yes to proceed.

Customize the Installation

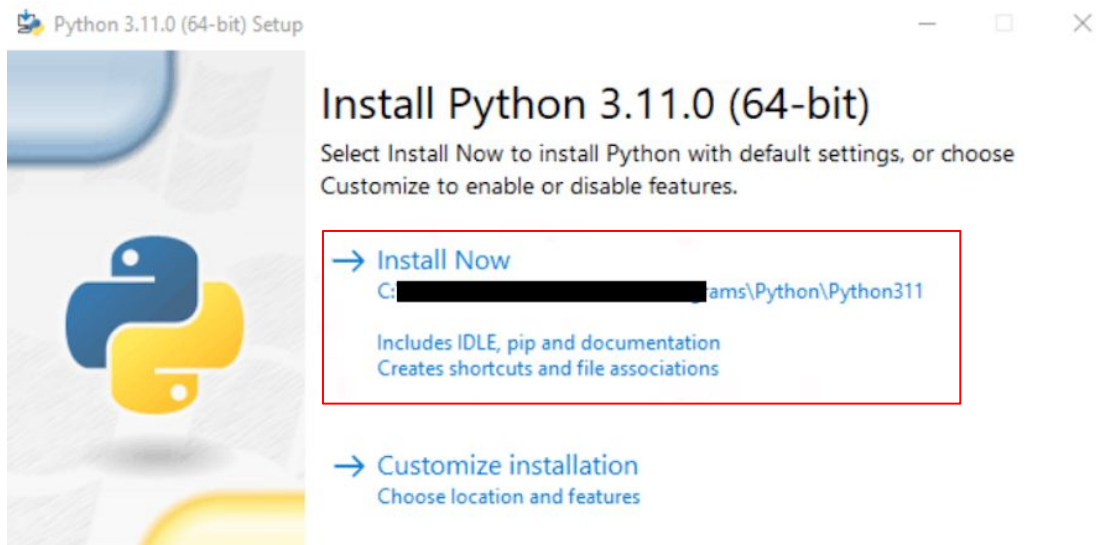On the installer's welcome screen, you'll see two options

1. Install Now
2. Customize installation

Customize the Installation

If you want to install Python with the default settings, simply click Install Now.

Customize the Installation

If you want to customize the installation (changing the installation directory or selecting specific components, for instance), click Customize installation.
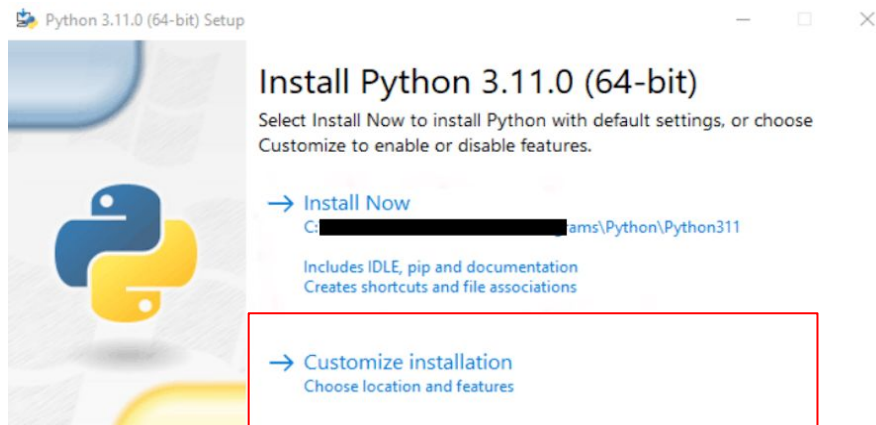
Install Python

After selecting your desired installation settings, click Install to begin the installation process. The installer will copy the necessary files to your computer and set up Python. This process may take a few minutes.
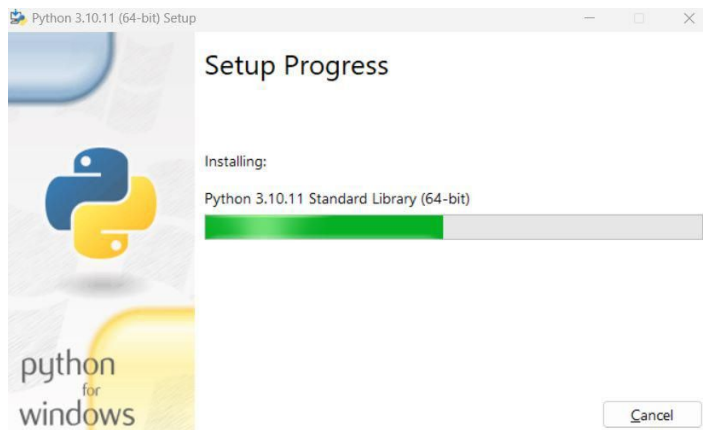
Verify the installation

Once the installation is complete, you can verify that Python has been installed correctly by opening the Command Prompt and typing the following command:

```
python --version
```

Integrated Development Environment (IDE)

Popular IDE of python

1. VScode
2. Pycharm
3. Spyder

Structure

1. Print function
2. Variable
3. Data Types
4. Operator
5. If-else
6. Loop
7. Function
8. Return Data
9. Scope Variable

Print function

Pattern print("Hello World!") **Result** **Hello World!**	**Pattern** print(1+20) **Result** **21**

Variable

1. Variable
2. Value

Assignment

age = 25

Variable ⟶   ⟵ Value

Data Types

1. Numeric (Integer, Float, Complex)
   - Integer: 1
   - Float: 2.0
   - Complex: 3+2i
2. String
   - "Cherry"
3. List
   - ["Apple", "Orange"]
4. Dictionary
   - {"name": "HALL", "age": "30"}
5. Tuple
   - ("1", "2", "3") look like List but it cannot edit when you created
6. Boolean
   - True, False

Data Types

Numeric
Bool
Sequence

int   float   Complex

Mapping

Set

Dictionary

List
Tuple
Range

Frozenset

Change Data Types

Function	Data Type
str('Hello World')	str
int(20)	int
float(20.5)	float
complex(1j)	complex
list(('apple', 'orange', 'banana'))	list
tuple(('apple', 'orange', 'banana'))	tuple

Arithmetic Operators

Operator	Result
+	10 + 20 = 30
-	20 - 10 = 10
*	20 * 2 = 40
/	20 / 3 = 6.666666666666667
//	20 // 3 = 6
%	20 % 3 = 2

Comparison Operators

a = 10 , b = 6

Operator	Result
==	a == b: False
!=	a != b: True
<	a < b: False
>	a > b: True
<=	a <= b: False
>=	a >= b: True

Logical Operators

Operator	Result
and	True and True: True True and False: False False and True: False False and False: False
or	True or True: True True or False: True False or True: True False or False: False
not	not True: False not False: True

Identify Operators

Operator	Result
is	a=10 b=10 a is b = False a=10 a=b a is b = True
is not	a=10 b=10 a is not b = True

Membership Operator

Operator	Result
in	a=3 b=[1,2,3] a in b = True
not in	a=3 b=[1,2,3] a not in b = False

Assignment Operators

Operator	Result
=	a = 10
+=	a += 10 a = a + 10
-=	a -= 10 a = a - 10
*=	a *= 10 a = a * 10
/=	a /= 10 a = a / 10
%=	a %= 10 a = a % 10

Bitwise Operator

a = 10 (binary: 1010) , b = 6   (binary: 0110)

Operator	Result
& (AND)	a & b = 2 (binary: 0010)
| (OR)	a | b = 14 (binary: 1110)
^ (XOR)	a ^ b = 12 (binary: 1100)
~ (NOT)	~a = 10 (binary: 1010)
<< (Shift bit left)	a << 2 = 40 (binary: 101000)
>> (Shift bit right)	a >> 2 = 2 (binary: 000010)

How to Shift Bit Left (<<)

x = 10: (binary: 1010) shift bit from the left 2 bit

Sol

1010 -> Add zero 2 bit into right side -> 101000

Ans 101000

How to Shift Bit Right (>>)

x = 10: (binary: 1010) shift bit from the right 2 bit

Sol

1010 -> Shift bit right 2 -> 10 -> Add zero 2 bit into left side -> 0010

Ans 0010

Sequence Operator

Operator	Result
()	10 + (20 - 20) = 10
**	10 ** 2 = 100
*, /, //, %	20 * 2 = 40
+,-	20 - 10 = 10
==, !=, <, >, <=, >=	10 == 10: True
and, or, not	True and True: True
=, +=, -=, *=, /=, %=, //=	10 + 10 = 20
=	A = 10

if-elif-else

Pattern

a = 10, b = 2

If (condition):

    Do somethings…

elif (condition):

    Do somethings…

else:

    Do somethings…

Use Case

a = 10, b = 2

If (a == b):

    print("a == b")

elif (a/2 == 10):

    print("a/2 == 10")

else:

    print("a !=b and a/2 != 10")

for-loop

Pattern

value

Start | End

```
for value in range(0,10):

    print(value)
```

Result

0

1

2

3

…

9

While loop

Pattern	Result

i=0
 	
While (i>=10):
 	
    i+=1
 	
    print(i)
	1
	
	2
	
	3
	
	4
	
	…
	
	10

Function

Pattern

def function_name():

    print("Test Function")

Use Case

function_name()

Result

Test Function

Return Function

Pattern

def function_name():

    return 10

Use Case

a = function_name() + 10

print(a)

Result:

20

Return if-elif-else

Pattern

If (condition):

    return something

elif (condition):

    return something

else:

    return something

Use Case

a = 10, b = 2

def return_result():

    If (a == b):

        return 10

    elif (a/2 == 10):

        return 20

    else:

        return 30

print(return_result())

Result

30

Scope Variable

Local Scope

def myfunc():

      x=10

myfunc()

print(x)

Result

NameError: name 'x' is not defined

Global Scope

def myfunc():

      global a

      a = 10

myfunc()

print(a)

Result

10