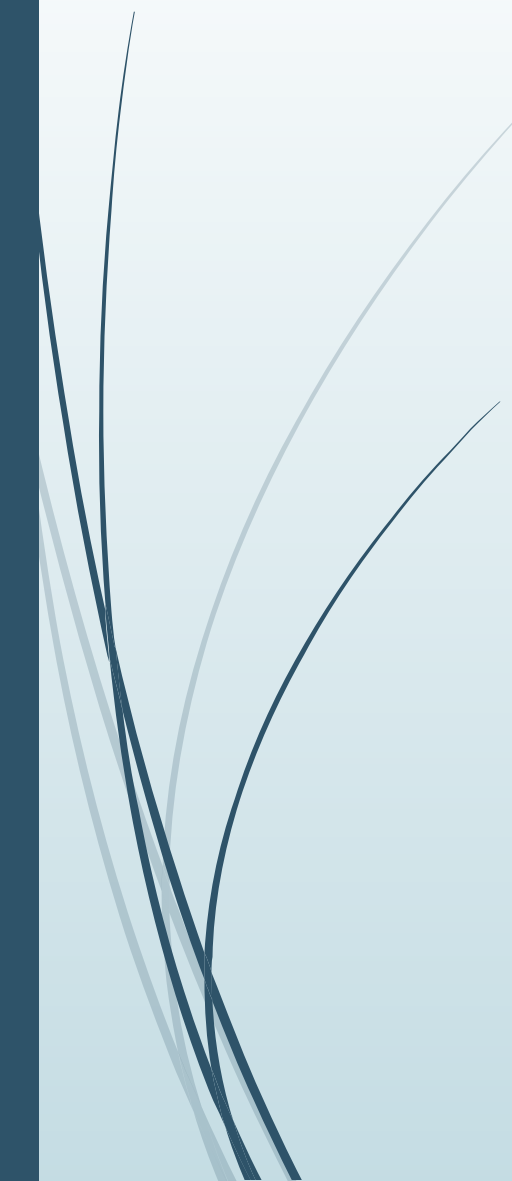




Agenda for Today

- What is Python?
 - Why Python?
 - Features of Python
 - Components of Python
 - History of Python?
- 



What is Python?

- **Python** is a General Purpose object-oriented programming language, which means that it can model real-world entities. It is also dynamically-typed because it carries out type-checking at runtime.
- Python is a [multi-paradigm programming language](#). [Object-oriented programming](#) and [structured programming](#) are fully supported, and many of its features support functional programming and [aspect-oriented programming](#).



Why Python?

- ❑ Easy to learn.
- ❑ Popular
- ❑ Wide application Area
 - Web development
 - Scientific programming
 - [Data science](#)
 - [Machine learning](#)
 - [Numerical analysis](#)
 - [Statistics](#)
 - Desktop [GUIs](#) applications
 - Network programming
 - [Game](#) programming.
 - Complex [algorithms](#) creation
 - Automation scripts
 - [Machine learning](#) and [artificial intelligence](#)
 - Audio and video applications



EASY JAVA VS PYTHON

```
public class AddTwoIntegers {  
    public static void main(String[] args) {  
  
        int a = 10;  
        int b = 20;  
  
        int sum = a + b;  
  
        System.out.println("The sum is: " + sum);  
    }  
}
```

```
a = 10  
b = 20  
sum = a + b  
print("The sum is:", sum)
```

COMPANIES USING
PYTHON



Google

Facebook

Quora

Amazon

Stripe

Instagram

Spotify

Netflix

Uber

Reddit

Dropbox

Pinterest

NASA

Instacart

Lyft

Industrial Light and Magic

Intel

IBM

Pixar

JP Morgan Chase



Features of Python



- ❑ Interpreted
- ❑ Interactive
- ❑ Object Oriented
- ❑ Platform Independent & portable
- ❑ Dynamic Typing
- ❑ Easier debugging
- ❑ Automatic Memory Management
- ❑ Free and Open source



Components of Python



- ❑ Functions
 - ❑ Collections of Statements , May return a value, reusable
- ❑ Classes
 - ❑ Abstract data type, Blue print of an object
- ❑ Modules
 - ❑ Group of Related Classes and Functions
- ❑ Packages
 - ❑ Group of related Modules



History



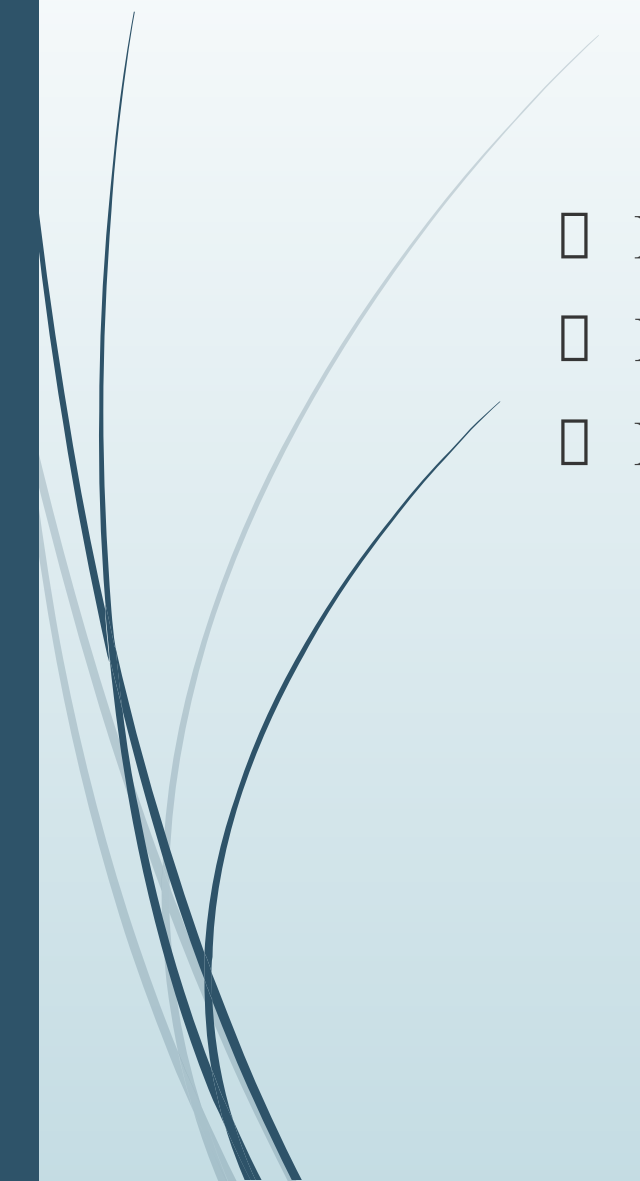
- Conception of Idea 1980
- December 1989 –Implementation Guido Van Rossum
- October 2000 – Python 2.0
- December 2008 – Python 3.0 Released
- Python 2.7 –EOL 2020

References

- ❑ <https://docs.python.org/3/tutorial/>
- ❑ <https://www.tutorialspoint.com/python/index.htm>
- ❑ <https://www.freecodecamp.org/news/learning-python-from-zero-to-hero-120ea540b567/>
- ❑ <https://www.guru99.com/python-tutorials.html>
- ❑ <https://www.programiz.com/python-programming/first-program>
- ❑ <https://data-flair.training/blogs/python-tutorials-home/>
- ❑ <https://data-flair.training/courses/python-course/lessons/1-1-introduction-to-python/>
- ❑ [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))
- ❑ https://en.wikipedia.org/wiki/Programming_paradigm#Support_for_multiple_paradigms
- ❑ https://en.wikipedia.org/wiki/Object-oriented_programming
- ❑ https://en.wikipedia.org/wiki/Structured_programming
- ❑ <http://openbookproject.net/thinkcs/python/english3e/>
- ❑ <https://levelup.gitconnected.com/6-must-try-python-programs-5d92ff36e620> (Projects For Fun)
- ❑ https://book.pythontips.com/en/latest/args_and_kwargs.html



Agenda for Today

- Downloading Options
 - Installing
 - First Program in Python
- 



Download Options Python



- ❑ Download from Microsoft Store. –search python 3.9
 - ❑ Download from Python (<https://www.python.org/downloads/>)
 - ❑ Download Thonny
 - ❑ Download Ananconda
 - ❑ Many others
 - ❑ Online Platforms without downloading.
-
- ❑ Installing Jupyter Notebook - <https://jupyter.org/install>

Download & Install Spyder

1

Go to Anaconda website

Click on Download Options

2

3

Choose Installer and Click Download

Complete Setup and Click Finish

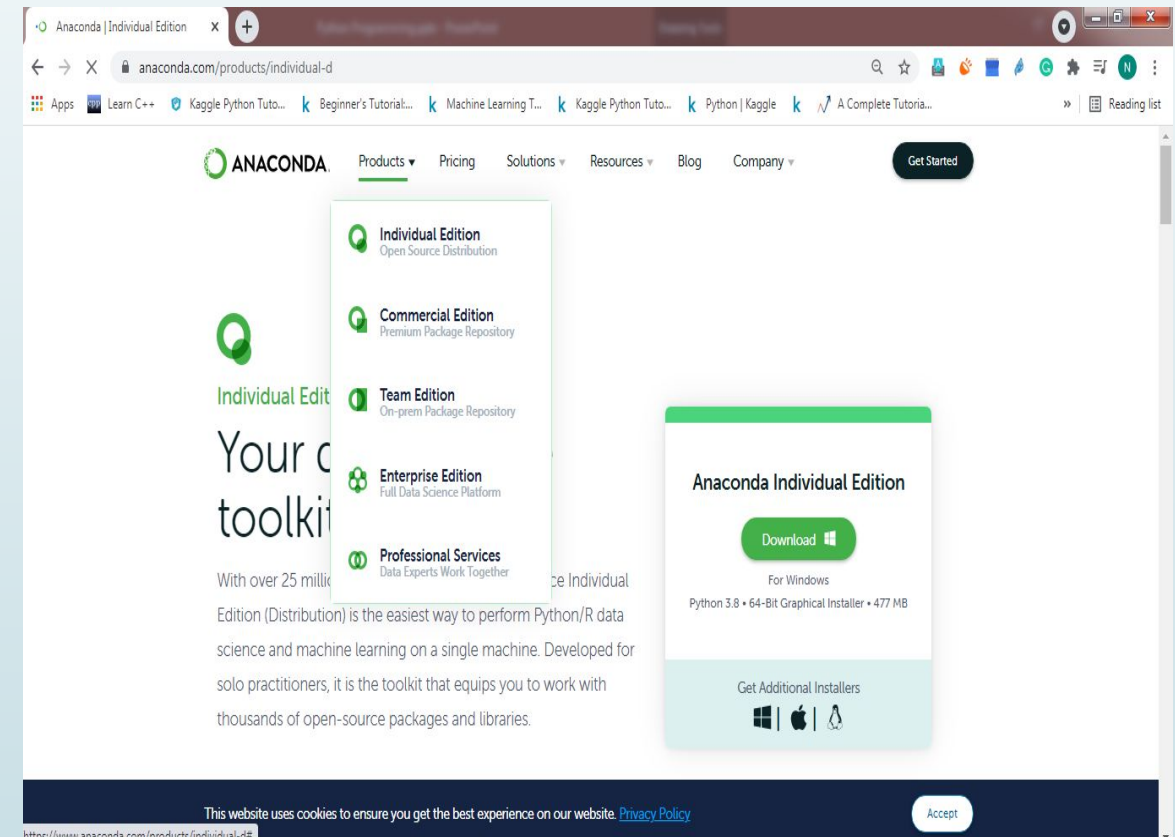
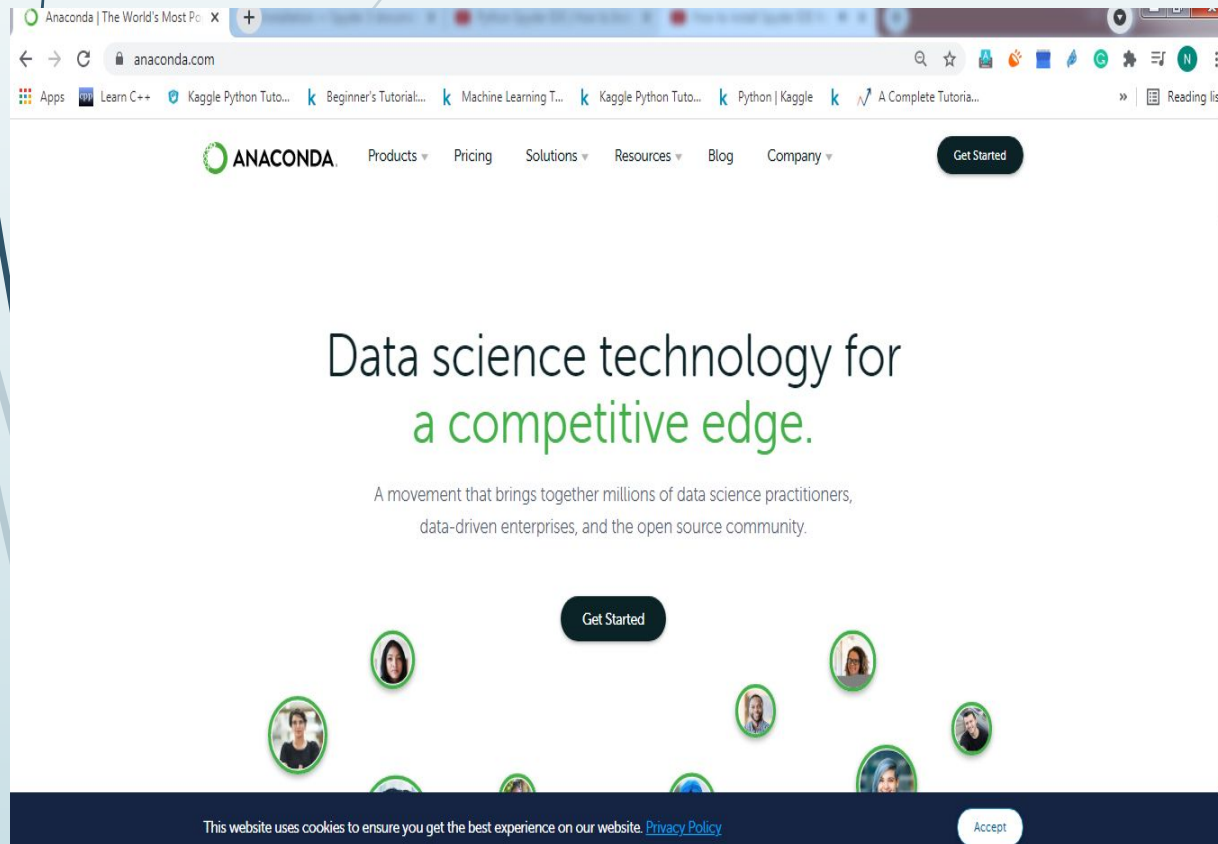
4

5

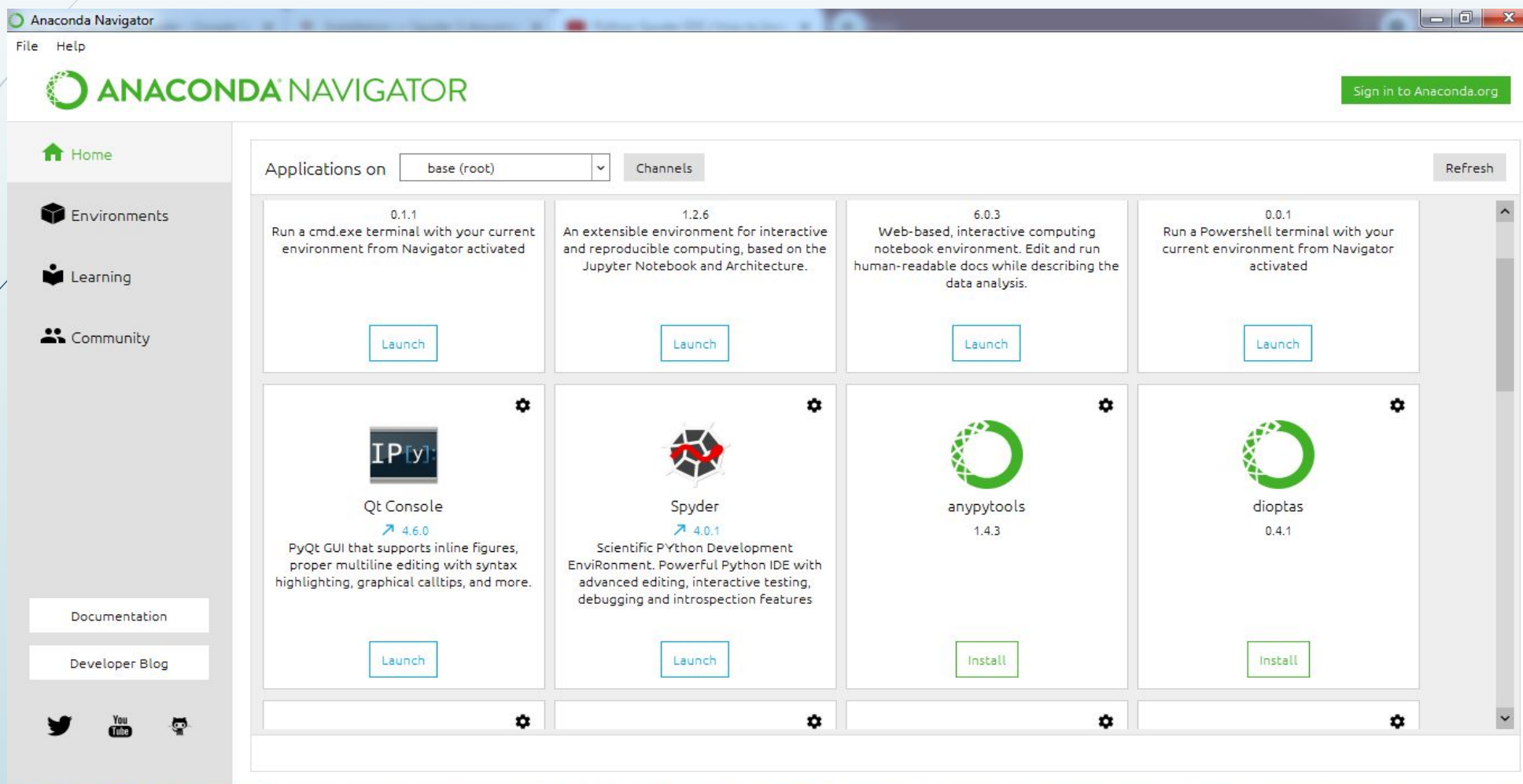
Launch Spyder from Anaconda

Follow steps

- <https://anaconda.com> and select products, click Individual Edition & Download



Launch Spyder from Anaconda Navigator





Create New File or Project

- ❑ Go to File Menu and select new File or
- ❑ Go to Projects Menu and select New Project- Give some Name to your Project and click Create.
- ❑ Now You can see The Python interface..
 - ❑ Lets See....

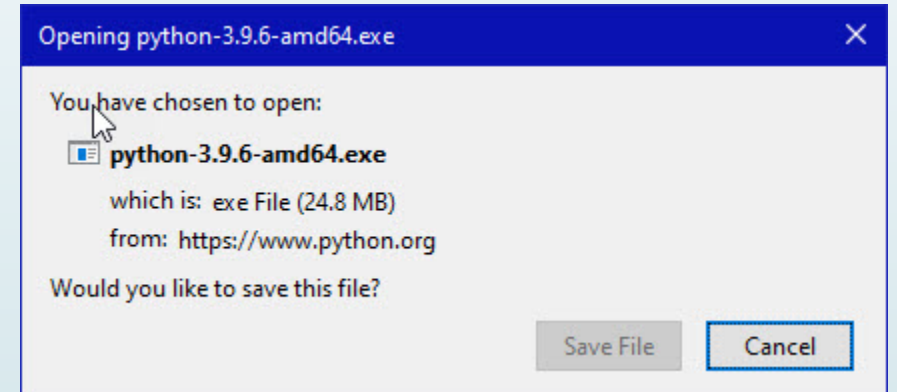


Download & Install Python: Version 3.9.6

- The Python download requires about 25 Mb of disk space; keep it on your machine, in case you need to re-install Python. When installed, Python requires about an additional 90 Mb of disk space.
- **Downloading**
- Click [Python Download](https://www.python.org/downloads/). (https://www.python.org/downloads/)
- The following page will appear in your browser.



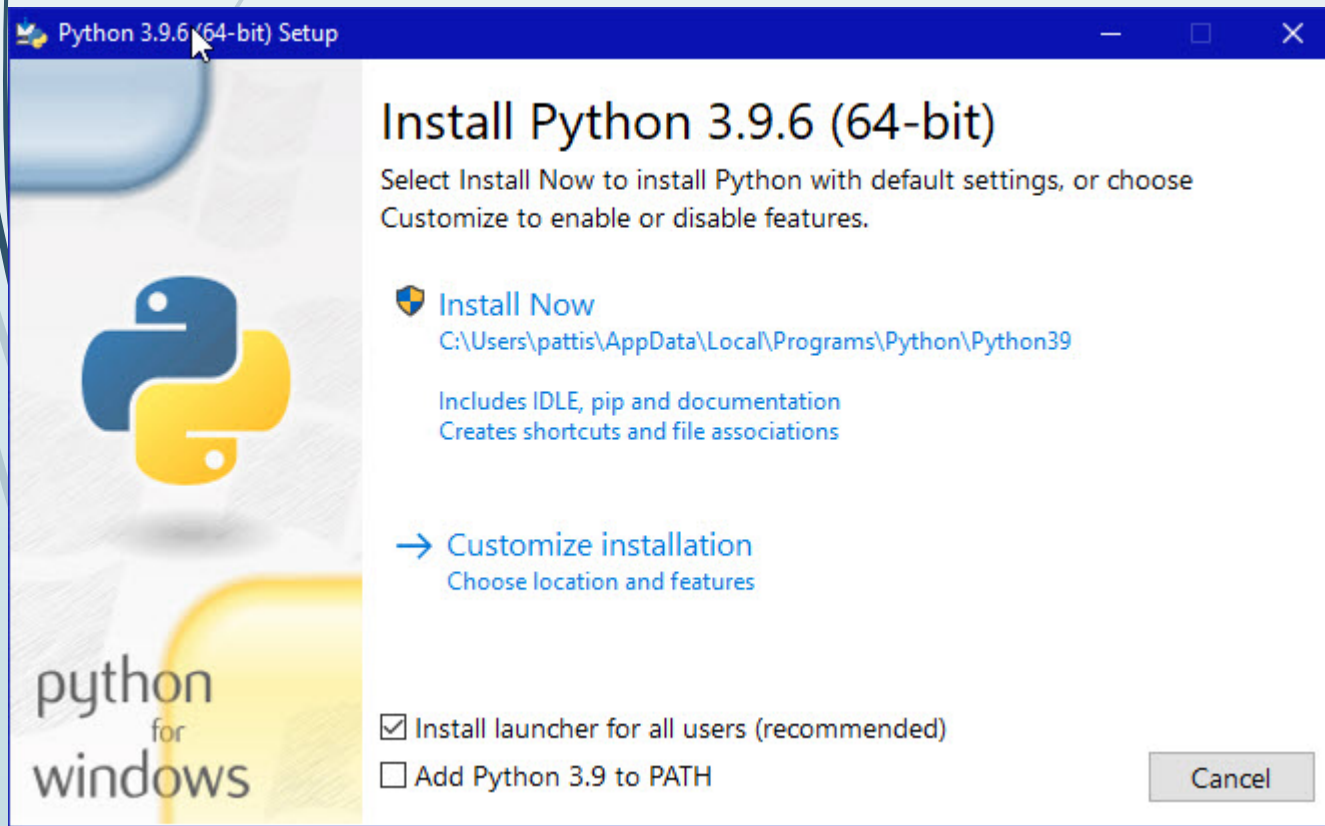
Click the **Download Python 3.9.6** button. The following pop-up window titled **Opening python-3.9.6-amd64.exe** will appear



Click the **Save File** button.

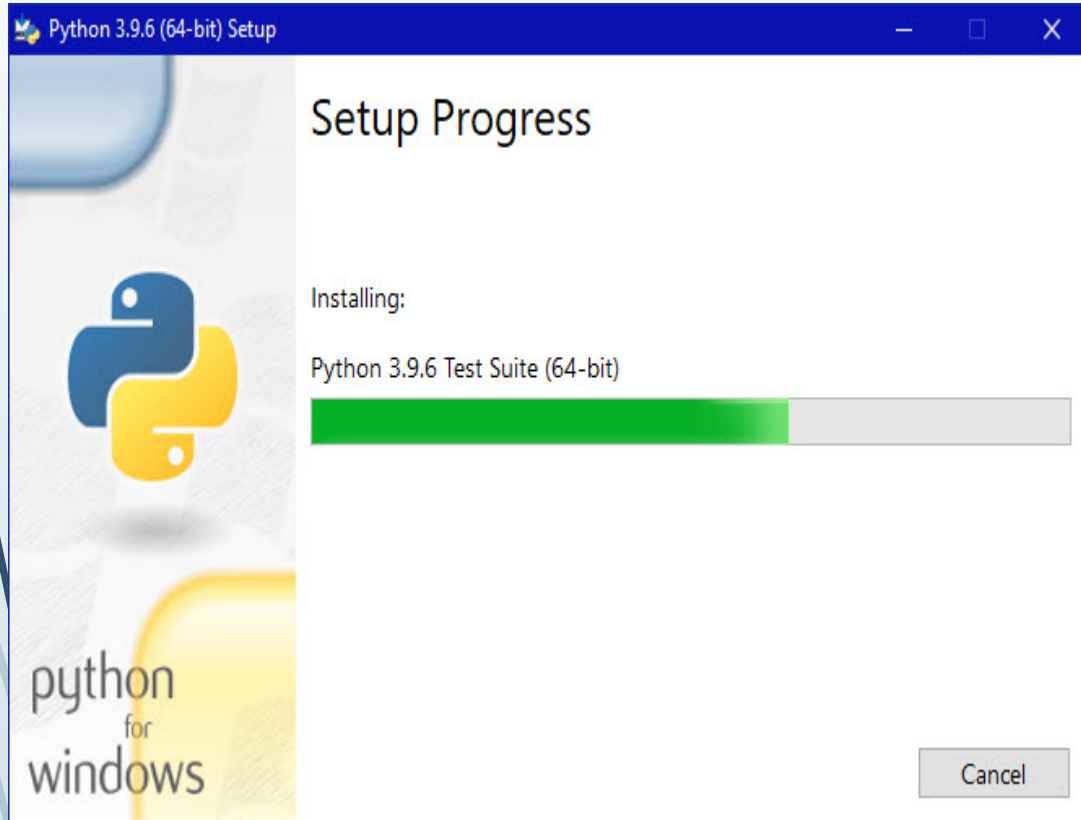
The file named **python-3.9.6-amd64.exe** should start downloading into your standard download folder. This file is about 25 Mb so it might take a while to download fully if you are on a slow internet connection.

- ❑ Move this file to a more permanent location, so that you can install Python (and reinstall it easily later, if necessary).
- ❑ Feel free to explore this webpage further; if you want to just continue the installation, you can terminate the tab browsing this webpage.
- ❑ Start the **Installing** instructions directly below.
- ❑ Double-click the icon labeling the file **python-3.9.6-amd64.exe**. A **Python 3.9.6 (64-bit) Setup** pop-up window will appear.

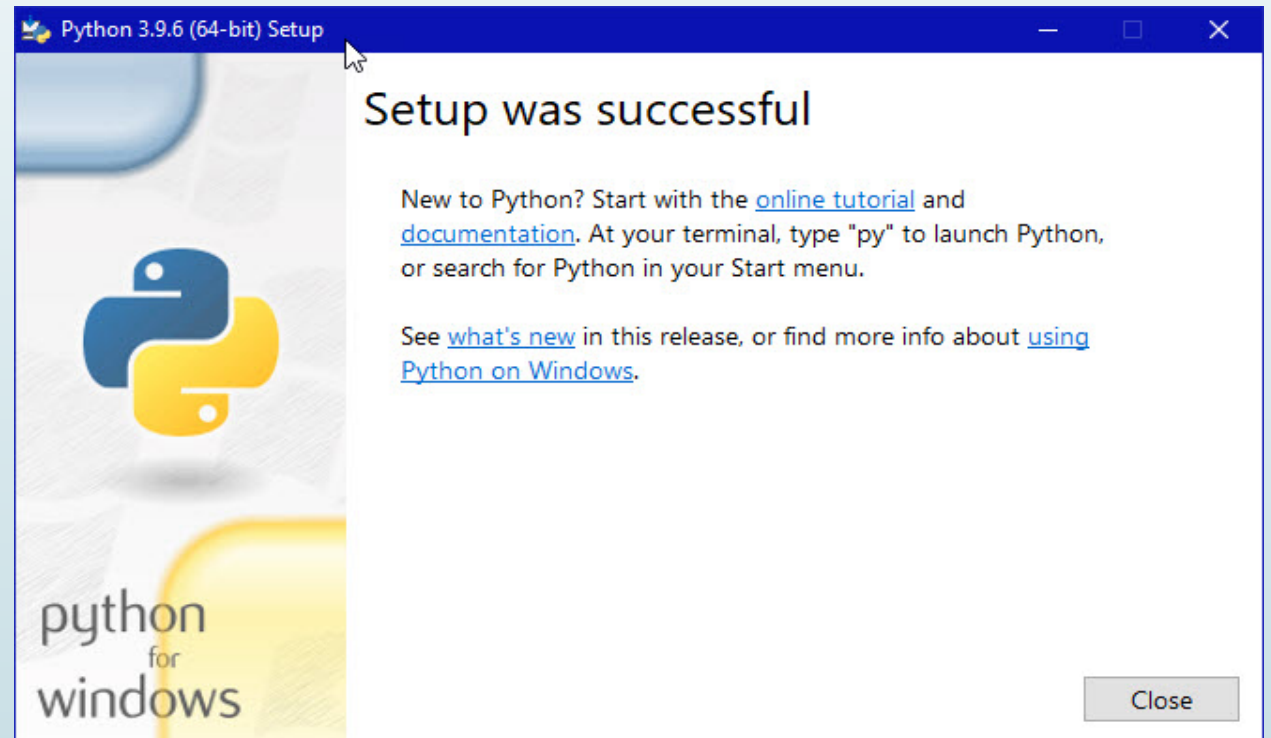


Ensure that **both** the **Install launcher for all users (recommended)** and the **Add Python 3.9 to PATH** checkboxes at the bottom are checked: typically only first is checked by default. If the Python Installer finds an earlier version of Python installed on your computer, the **Install Now** message may instead appear as **Upgrade Now** (and the checkboxes will not appear). Highlight the **Install Now** (or **Upgrade Now**) message, and then click it.

- A new **Python 3.9.6 (64-bit) Setup** pop-up window will appear with a **Setup Progress** message and a progress bar.



During installation, it will show the various components it is installing and move the progress bar towards completion. Soon, a new **Python 3.9.6 (64-bit) Setup** pop-up window will appear with a **Setup was successfully** message



Click the **Close** button. Python should now be installed.

Download & Install Sublime Text3

- ❑ Go to website <https://www.sublimetext.com/3>
- ❑ Select your OS and download.
- ❑ Go to the download location and double click installation file.
- ❑ Once installed, open sublime Text3, you are ready to write your code.
- ❑ Write the code and save the file using .py extension.
- ❑ Go to Tools menu select Build sytem, it should be on Automatic (confirm)
- ❑ Now Again go to Tools and select build. Your code should run and a console window with ouput should appear.
- ❑ For taking input install packagecontrol (press ctrl + shift +p) and type install, click and it should be installed and then press ctrl + shift +p and select packagecontrol:Install package and it will list the packeges which you can install, type sublimeREPL and install. Lets see...

New Build system for sublime

```
{  
  "cmd":["C:/Users/<user>/AppData/Local/Programs/Python/Python37-32/python.exe", "-u", "$file"],  
  "file_regex": "^[ ]File \"(...?)\\", line ([0-9]*)",  
  "selector": "source.python"  
}
```



First Program -Hello World

- `print("Hello World")`
- Or
- `print('Hello World')`
- Ex: Hello.py

Two ways of execution

❑ Interactive Mode

- ❑ Open command Prompt, type python or python3 (depending on python installation) and then hit enter. Interface is open in interactive mode.
- ❑ The standard prompt for the interactive mode is >>>, so as soon as you see these characters, you'll know you are in. Now, you can write and run Python code as you wish, with the only drawback being that when you close the session, your code will be gone. when you work interactively, every expression and statement you type in is evaluated and executed immediately:

❑ Script Mode

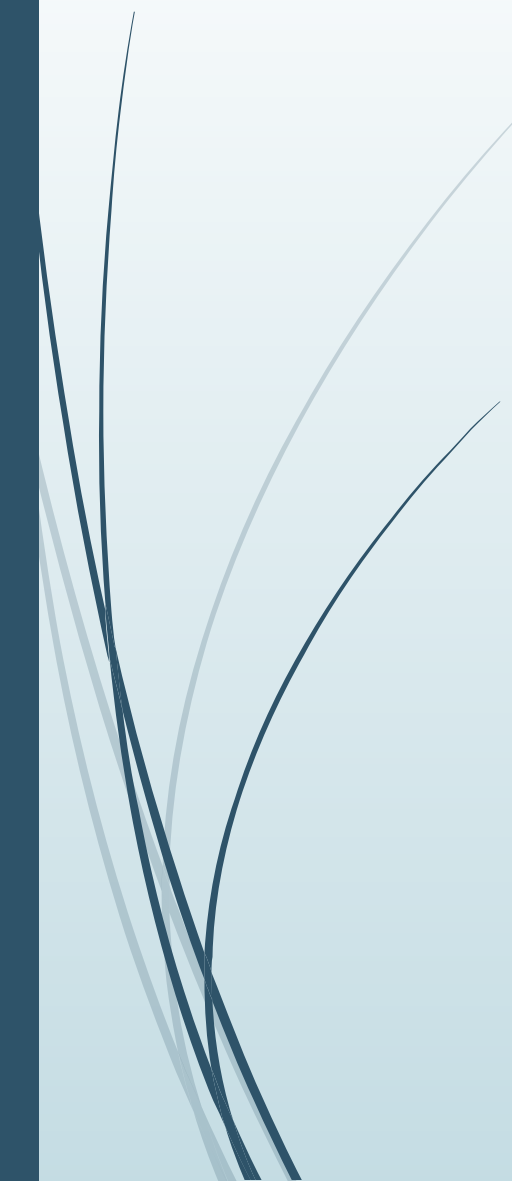
- ❑ A plain text file containing Python code that is intended to be directly executed by the user is usually called **script**.
- ❑ **modules are meant to be imported**, while **scripts are made to be directly executed**.
- ❑ Script mode can be run from command prompt & IDE
- ❑ Output can be saved using `c:\> python Hello.py > out.txt`

First program with modules

- Create the file and put `print("Hello World")` in the file. Name the file `HWM.py`.
- On python shell `>>> import sys`
- `>>> sys.path`
- `>>> sys.path.append(r' C:\Users\DIRECTORY_FOR_MODULE')`
- `>>> import HWM`
- `>>> Hello World` -----Output



Agenda for Today

- Identifiers
 - Variables
 - String formatters
 - Python Statements
- 

Identifiers

- Names we give to identity variables, functions, variables and classes etc.
- **Rules:**
 - Only begin with A-Z, a -z or underscore
 - 0 or more letters, digits and underscore may follow.
 - It is case sensitive; name is not same as Name
 - Reserved words cannot be used as identifiers. (approx. 35)

Best Practice

Upper case Initials for class name, lowercase for all other.

Name of private identifier with a leading underscore

Name of strongly private identifier with two leading underscore

Special identifiers by python begin and end with two underscores.

Variables

- Dynamic typing
 - No need to define datatype before assigning.
 - `A = 10` (python knows it is integer)
 - `B = 1.2` (float)
 - `C = 1+3j` (complex)
 - `D = "Hello" or 'Hello'` (string)
 - We can find the datatype of variable by using type method
 - `type(A)` – int
 - `Type(B)` - float
 - `type(C)` – complex
 - `type(D)` – str

Variables – Little More

Python
a = 10

Puts value 10 in memory and a refers to the address of that memory

Java
Int a = 10

Assigns 4 byte of memory for a and put value 10 in the assigned space

What happens when we change a = 12

Python
a = 12

Puts value 12 in memory and a refers to the address of that memory. Reference to memory address of the value 10 is lost.

Java
a = 12

use same 4 byte of memory for a and put new value 12 in the assigned space, earlier where 10 was written

Checking the concept learned

- `A = 10`
- `Id(A)` – some unique number associated with storage suppose x1
- `A = 11`
- `Id(A)` -some unique number associated with storage suppose x2
- The two outputs from call to function `id()` will be different.
- Lets check
- Limit to storage
 - In java 4 bytes are allocated for the integer so the range of values it can store is fixed whereas in python there is no limit to the range of the values it can store, the only restriction is how much space your program has been allocated.

Assign Multiple values

□ Many Values to Multiple Variables

```
□ x, y, z = "Orange", "Banana", "Cherry"  
  print(x)  
  print(y)  
  print(z)
```

□ One Value to Multiple Variables

```
□ x = y = z = "Orange"  
  print(x)  
  print(y)  
  print(z)
```

Write a program to swap two values

```
num1, num2 = 7,8
```

String formatters

- ❑ `X=10, printer="HP"`
- ❑ `print(" I just printed %s pages from the printer %s"%(X,printer) -%`
- ❑ `print(" I just printed {} pages from the printer {1}").format(X,printer)-format`
- ❑ `print(" I just printed {X} pages from the printer {printer}").format(X=11,printer='Epson) – format with assignment`
- ❑ `Print(f" I just printed {x} pages from the printer {printer}") -f-strings`

Examples

```
# Python3 program to demonstrate  
# the str.format() method
```

```
# using format option in a simple string  
print("{}, A computer science Engineering in ."  
      .format("Faculty of Engineering & Technolgy"))
```

```
# using format option for a  
# value stored in a variable  
str = "This article is written in {}"  
print(str.format("Python"))
```

```
# formatting a string using a numeric constant  
print("Hello, I am {} years old !".format(18))
```

```
# Python program using multiple place  
# holders to demonstrate str.format() method
```

```
# different datatypes can be used in formatting  
print("Hi ! My name is {} and I am {} years old"  
      .format("User", 19))
```

```
# The values passed as parameters  
# are replaced in order of their entry  
print("This is {} {} {} {}"  
      .format("one", "two", "three", "four"))
```

To demonstrate the use of formatters
with positional key arguments.

```
# Positional arguments  
# are placed in order  
print("{0} love {1}!!".format("Fetians", "FET"))
```

```
# Reverse the index numbers with the  
# parameters of the placeholders  
print("{1} love {0}!!".format("Fetians", "FET"))
```

```
print("Every {} should know the use of {} {} programming and {}"  
      .format("programmer", "Open", "Source",  
              "Operating Systems"))
```

```
# Use the index numbers of the  
# values to change the order that  
# they appear in the string  
print("Every {3} should know the use of {2} {1} programming and {0}"  
      .format("programmer", "Open", "Source", "Operating Systems"))
```

```
# Keyword arguments are called  
# by their keyword name  
print("{gfg} is a {0} science portal for {1}"  
      .format("computer", "geeks", gfg="GeeksforGeeks"))
```


Python Statement

- Instructions that a Python interpreter can execute are called statements.
- `a = 1` is an assignment statement.
- `if` statement, `for` statement, `while` statement, etc. are other kinds of statements

Multi-line statement

In Python, the end of a statement is marked by a newline character. But we can make a statement extend over multiple lines with the line continuation character (`\`). For example:

```
a = 1 + 2 + 3 + \
    4 + 5 + 6 + \
    7 + 8 + 9
```

This is an explicit line continuation. In Python, line continuation is implied inside parentheses `()`, brackets `[]`, and braces `{ }`. For instance, we can implement the above multi-line statement as:

```
a = (1 + 2 + 3 +
    4 + 5 + 6 +
    7 + 8 + 9)
```

We can also put multi line statements in a single statement using `;`
`A=1;b=2;c=3;`

For multi line string statements we can use triple quotes.