

أساسيات لغة البرمجة بايثون

(Basics of Python Programming)

تعتبر لغة البرمجة بايثون (Python) أحد أسرع لغات البرمجة إنتشاراً، حيث كان أول اصدار لها عام 1990 وتعتبر من أسهل اللغات في الكتابة وأسرعها في التعلم.

يتم استخدام لغة البرمجة (Python) في أنظمة التشغيل المختلفة مثل (Windows) و نظام التشغيل (MacOS) ونظام التشغيل (Linux). ومن اهم مميزات لغة البرمجة (Python) أنها لغة مفتوحة المصدر، مما يعني أنه يمكن تحميل (Code Source) الخاص بها وتعديله واعادة استخدامه.



نشاط
تمهيدي

أبحث - بالتعاون مع أفراد مجموعتي - في المواقع الإلكترونية الموثوقة في شبكة الإنترنت عن نشأة لغة البرمجة بايثون (Python)، واستخداماتها، وأهم مزاياها، ثم أناقش أفراد المجموعات الأخرى ومُعَلِّمي / مُعَلِّمتي فيما أتوصّل إليه من نتائج.

الحل:

نشأة لغة البرمجة بايثون:

تطورت لغة البرمجة بايثون بواسطة جويدو فان روسم في أواخر الثمانينيات وأوائل التسعينيات في المعهد الوطني لأبحاث الرياضيات وعلوم الكمبيوتر في هولندا. تم إصدار أول نسخة من بايثون في عام 1991.

استخدامات بايثون:

تطوير الويب: و تطوير تطبيقات الويب، خاصةً باستخدام أطر عمل مثل Django و Flask.

تحليل البيانات: وتعلم الآلة، والذكاء الاصطناعي، بفضل مكتبات مثل NumPy و

Pandas و Scikit-learn.

البرمجة النصية: لأتمتة المهام، مثل معالجة الملفات، وإدارة النظام، واختبار البرامج.

تطوير الألعاب: خاصةً مع مكتبات مثل Pygame.

تطبيقات سطح المكتب: وتطويرها باستخدام مكتبات مثل Tkinter و PyQt.

أهم مزايا بايثون:

سهولة التعلم: بسبب بساطة قواعدها اللغوية وسهولة قراءتها

مفتوحة المصدر: لأنها مجانية للاستخدام والتعديل والتوزيع.

دعم واسع: تتمتع بايثون بمجتمع كبير ونشط من المطورين، مما يوفر دعمًا واسعًا للمستخدمين.

مكتبات واسعة: تحتوي على مجموعة ضخمة من المكتبات التي تغطي مجموعة واسعة من المجالات، مما يجعلها لغة متعددة الاستخدامات

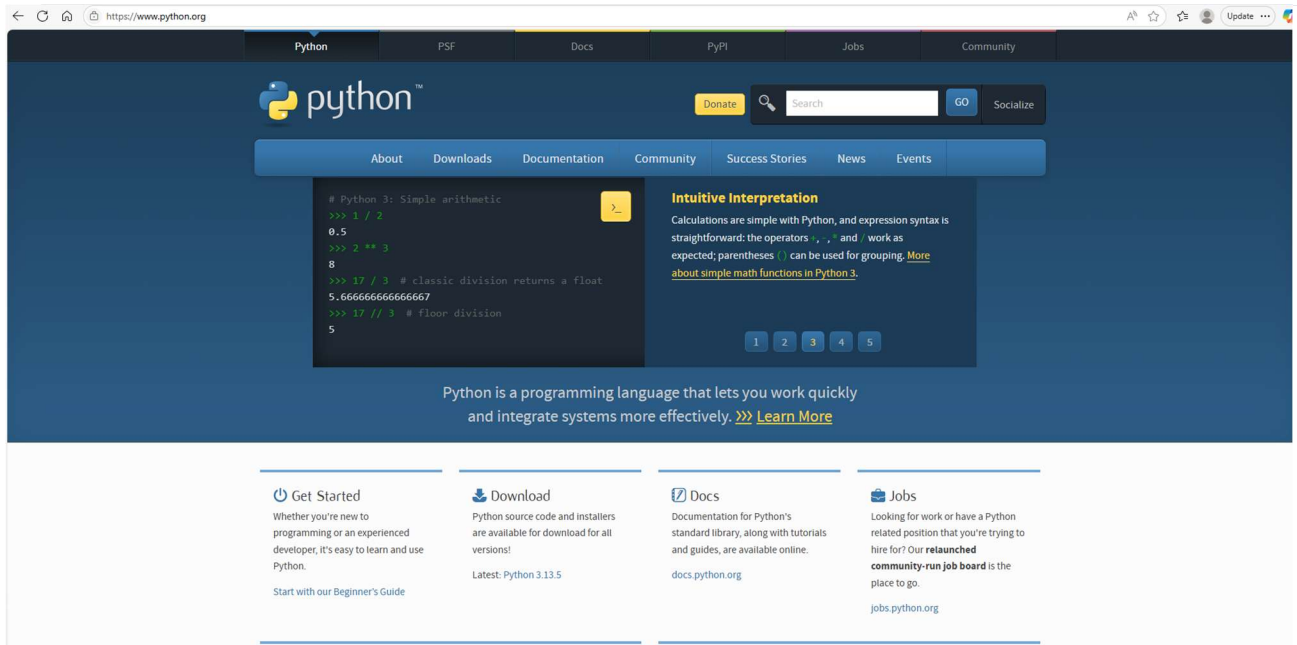
تثبيت لغة البرمجة بايثون (Python Setup)

لغايات تثبيت لغة البرمجة بايثون (Python) قم بالخطوات التالية:

أولاً: تحميل مُفسّر لغة البرمجة بايثون


1. زيارة الموقع الإلكتروني <https://www.python.org>

سوف تظهر لك هذه الصفحة:



2. قم بالضغط على رقم آخر اصدار متوافر للغة البرمجة بايثون كما يظهر في الصورة التي في الأسفل عند المنطقة المحاطة بمستطيل أحمر:

[Python](#)[PSF](#)[Docs](#)[PyPI](#)[Jobs](#)[Community](#)



[Donate](#)[GO](#)[Socialize](#)

[About](#)[Downloads](#)[Documentation](#)[Community](#)[Success Stories](#)[News](#)[Events](#)

```
# Python 3: List comprehensions
>>> fruits = ['Banana', 'Apple', 'Lime']
>>> loud_fruits = [fruit.upper() for fruit in fruits]
>>> print(loud_fruits)
['BANANA', 'APPLE', 'LIME']


# List and the enumerate function
>>> list(enumerate(fruits))
[(0, 'Banana'), (1, 'Apple'), (2, 'Lime')]
```

Compound Data Types

Lists (known as arrays in other languages) are one of the compound data types that Python understands. Lists can be indexed, sliced and manipulated with other built-in functions. [More about lists in Python 3](#)

[1](#)[2](#)[3](#)[4](#)[5](#)


Python is a programming language that lets you work quickly and integrate systems more effectively. [>>> Learn More](#)



Get Started

Whether you're new to programming or an experienced developer, it's easy to learn and use Python.


[Start with our Beginner's Guide](#)



Download

Python source code and installers are available for download for all versions!


Latest: Python 3.13.5



Docs

Documentation for Python's standard library, along with tutorials and guides, are available online.

[docs.python.org](#)



Jobs

Looking for work or have a Python related position that you're trying to hire for? Our **relaunched community-run job board** is the place to go.

[jobs.python.org](#)

حيث سوف يتم تحميل مفسر لغة البرمجة بايثون (Python) من خلال هذا الرابط. قم بحفظ الملف من خلال الضغط على زر (Save File).

ودوماً اعمل على أن تتأكد بأنك قمت بتحميل آخر إصدار متوفر من اللغة.

3. بعد الضغط سوف تظهر لك الشاشة التالية:

The screenshot shows the Python.org website's header with navigation links: Python, PSF, Docs, PyPI, Jobs, and Community. Below the header is a search bar with a 'GO' button and a 'Socialize' button. The main content area features a large heading 'Python 3.13.5' and a subheading 'This is the fifth maintenance release of Python 3.13'. The text explains that Python 3.13 is the newest major release and that 3.13.5 is the fifth maintenance release. It also mentions that 3.13.5 is an expedited release to fix a couple of significant issues with the 3.13.4 release. A list of issues is provided, including 'gh-135151: Building extension modules on Windows for the regular (non-free-threaded) build failed', 'gh-135171: Generator expressions stopped raising `TypeError` (when iterating over non-iterable objects) at creation time, delaying it to first use.', and 'gh-135326: Passing int-like objects (like `numpy.int64`) to `random.getrandbits()` failed, when it worked before.'

Python 3.13.5

Release Date: June 11, 2025

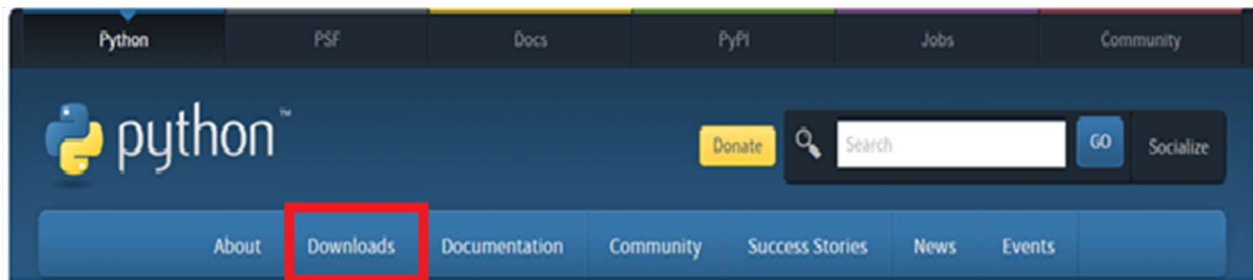
This is the fifth maintenance release of Python 3.13

Python 3.13 is the newest major release of the Python programming language, and it contains many new features and optimizations compared to Python 3.12. 3.13.5 is the fifth maintenance release of 3.13.

3.13.5 is an expedited release to fix a couple of significant issues with the 3.13.4 release:

- [gh-135151](#): Building extension modules on Windows for the regular (non-free-threaded) build failed.
- [gh-135171](#): Generator expressions stopped raising `TypeError` (when iterating over non-iterable objects) at creation time, delaying it to first use.
- [gh-135326](#): Passing int-like objects (like `numpy.int64`) to `random.getrandbits()` failed, when it worked before.

الآن قم بالضغط على قائمة (Downloads) المشار لها بالمستطيل الأحمر



Python 3.13.5

Release Date: June 11, 2025

This is the fifth maintenance release of Python 3.13


Python 3.13 is the newest major release of the Python programming language, and it contains many new features and optimizations compared to Python 3.12. 3.13.5 is the fifth maintenance release of 3.13.

3.13.5 is an expedited release to fix a couple of significant issues with the 3.13.4 release:

- [gh-135151](#): Building extension modules on Windows for the regular (non-free-threaded) build failed.
- [gh-135171](#): Generator expressions stopped raising `TypeError` (when iterating over non-iterable objects) at creation time, delaying it to first use.
- [gh-135326](#): Passing int-like objects (like `numpy.int64`) to `random.getrandbits()` failed, when it worked before.

عندها سوف تظهر لك الشاشة التالية:

[Python](#)[PSF](#)[Docs](#)[PyPI](#)[Jobs](#)[Community](#)

[Donate](#)[GO](#)[Socialize](#)

[About](#)[Downloads](#)[Documentation](#)[Community](#)[Success Stories](#)[News](#)[Events](#)

Python 3.13.5

Release Date: June 11, 2025

This is the fifth main

Python 3.13 is the newest major release. 3.13.5 is the fifth maintenance release of 3.13.

3.13.5 is an expedited release to fix a couple of significant issues with the 3.13.4 release:

- [gh-135151](#): Building extension modules on Windows for the regular (non-free-threaded) build failed.
- [gh-135171](#): Generator expressions stopped raising `TypeError` (when iterating over non-iterable objects) at creation time, delaying it to first use.
- [gh-135326](#): Passing int-like objects (like `numpy.int64`) to `random.getrandbits()` failed, when it worked before.

[All releases](#)[Source code](#)[Windows](#)[macOS](#)[Other Platforms](#)[License](#)[Alternative Implementations](#)

Download for Windows

Python 3.13.5

Note that Python 3.9+ cannot be used on Windows 7 or earlier.

Not the OS you are looking for? Python can be used on many operating systems and environments.
[View the full list of downloads.](#)

Python 3.12. 3.13.5 is the fifth maintenance release of 3.13.

الظاهرة بالرسم الذي

Python 3.13.5

قم بالضغط على كلمة

بالأعلى لغايات تحميل مفسر لغة البرمجة بايثون ،حيث سوف يتم تخزين الملف الخاص بالمفسر على جهاز الحاسوب الخاص بك على هذا الشكل:

 python-3.13.5-amd64

والان أصبح الملف موجودا لديك وجاهز لتثبيته على الجهاز.

ثانياً: تثبيت مُفسّر لغة البرمجة بايثون

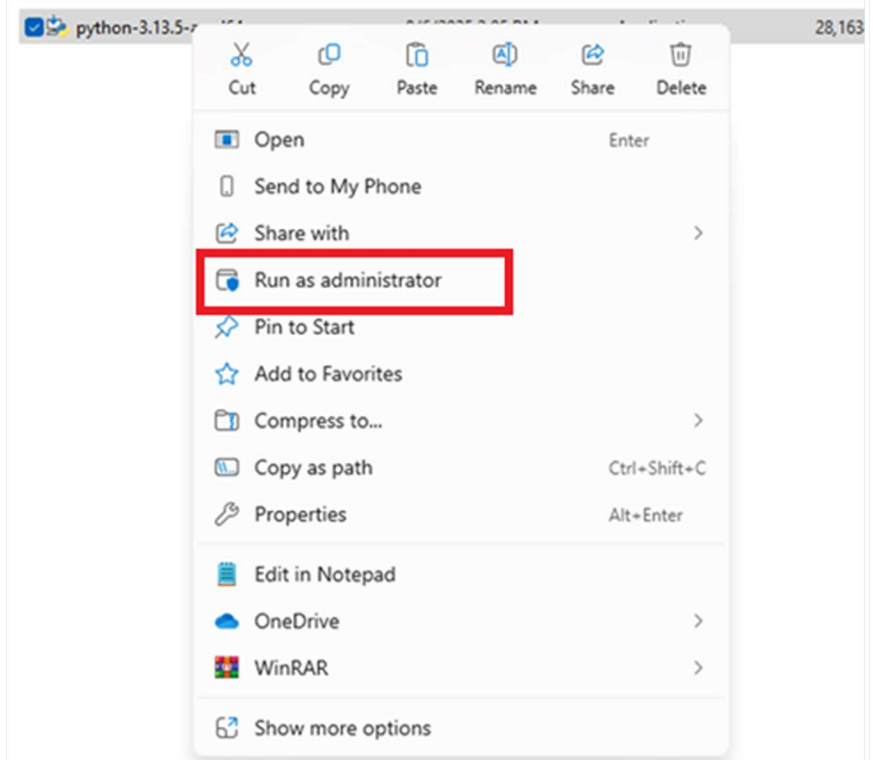
لغايات تثبيت مفسر لغة البرمجة بايثون، كل ما عليك هو :

1. الذهاب الى المكان الذي قمت فيه بتخزين الملف الذي يحمل الاسم (Python-3.13.5-amd64) والذي يوجد على هذا

python-3.13.5-amd64

الشكل

2. لنقم بالتأشير الى الملف من خلال الفأرة، ثم الضغط على الزر الأيمن للفأرة، سوف تظهر لك قائمة منسدلة كما في الصورة:



3. قم بالضغط على خيار (Run as administrator) الموجود بالقائمة المنسدلة المحاط بالمستطيل الأحمر، عندها سوف تظهر لنا الشاشة التالية:



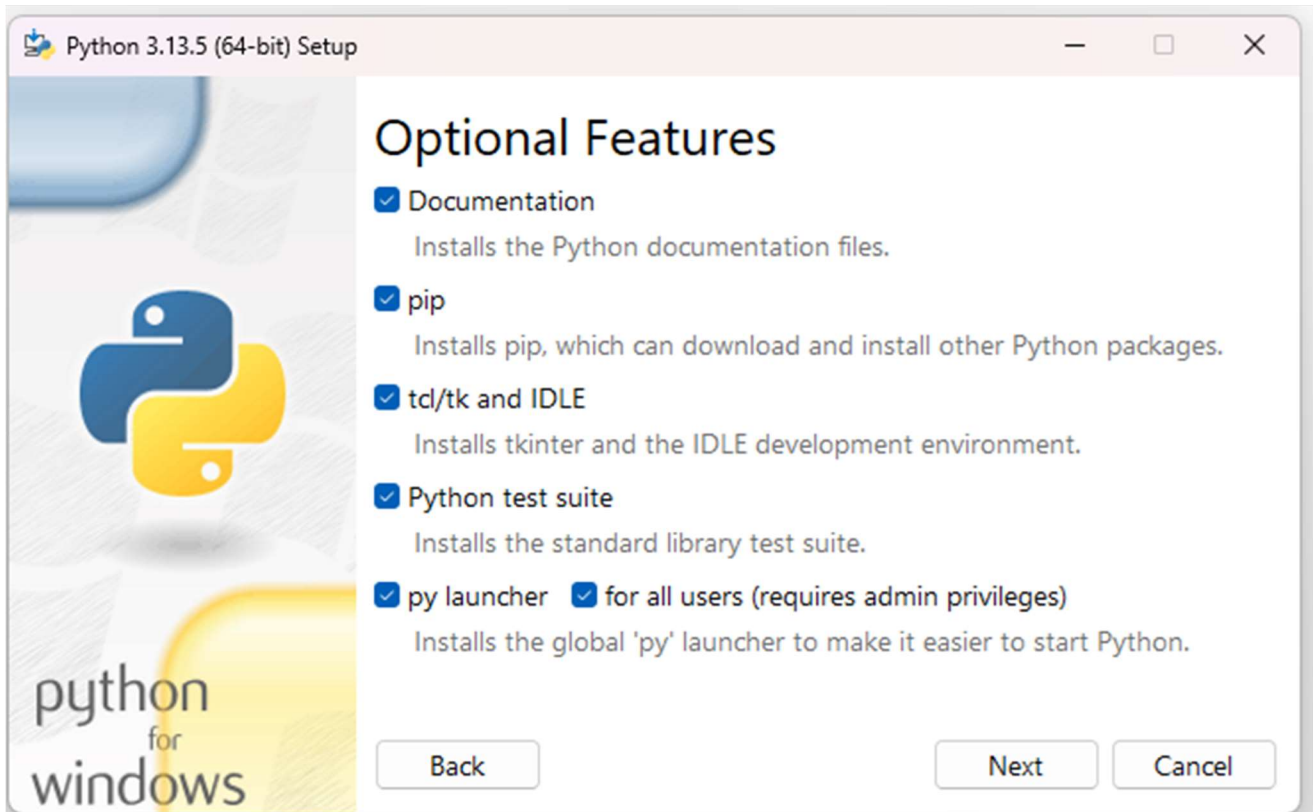
4. والآن، دعنا نقوم ب تفعيل الخيارات الموجودة داخل المستطيل الأحمر:

Use admin privileges when installing py.exe

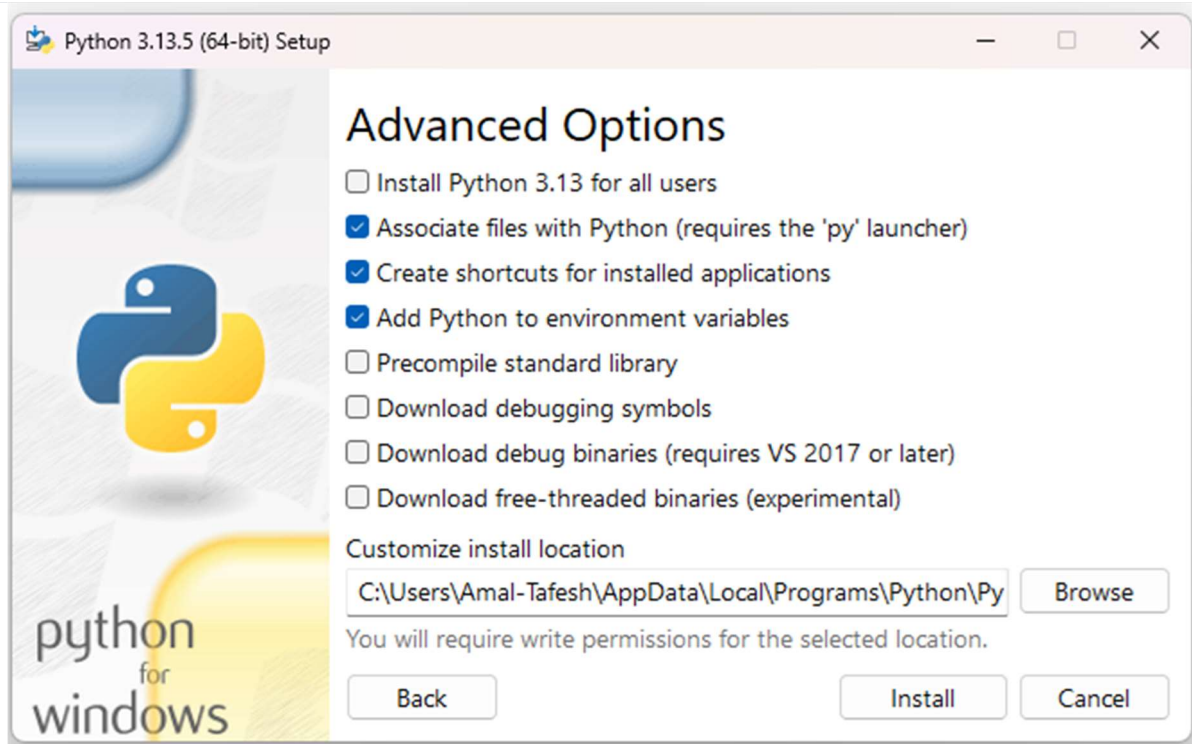
Add path to python.exe

من خلال الضغط على المربع بجانب كل امر وظهور علامة الصح، اذ إن عدم تفعيل الخيار الأول من هذه الخيارات يؤدي الى عدم إظهار رقم إصدار لغة البرمجة (Python) عند التحقق من تثبيت مفسر لغة البرمجة بايثون.

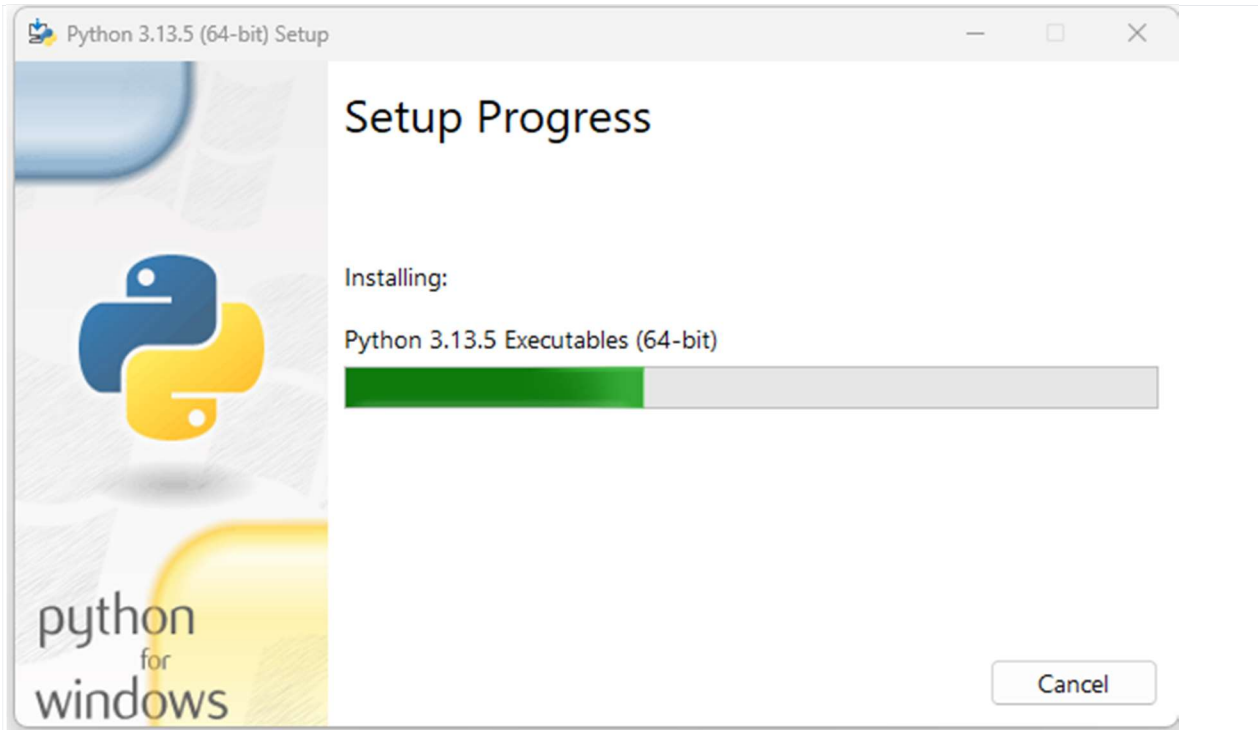
5. ثم سوف تظهر لنا الشاشة التالية:



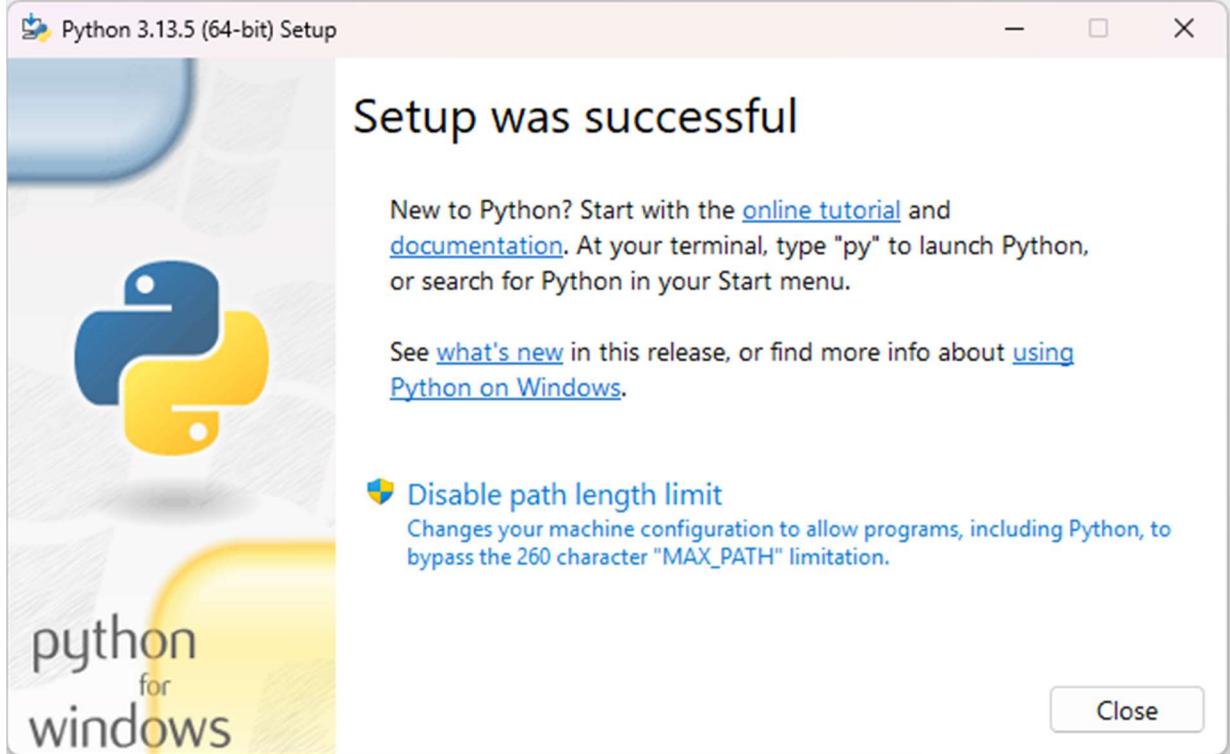
نقوم بالضغط على خيار NEXT لتظهر الشاشة التالية:



(حتى تبدأ عملية Install. الآن لنقم بالضغط على زر التنصيب (6 التنبيت كما يظهر بالشكل التالي:



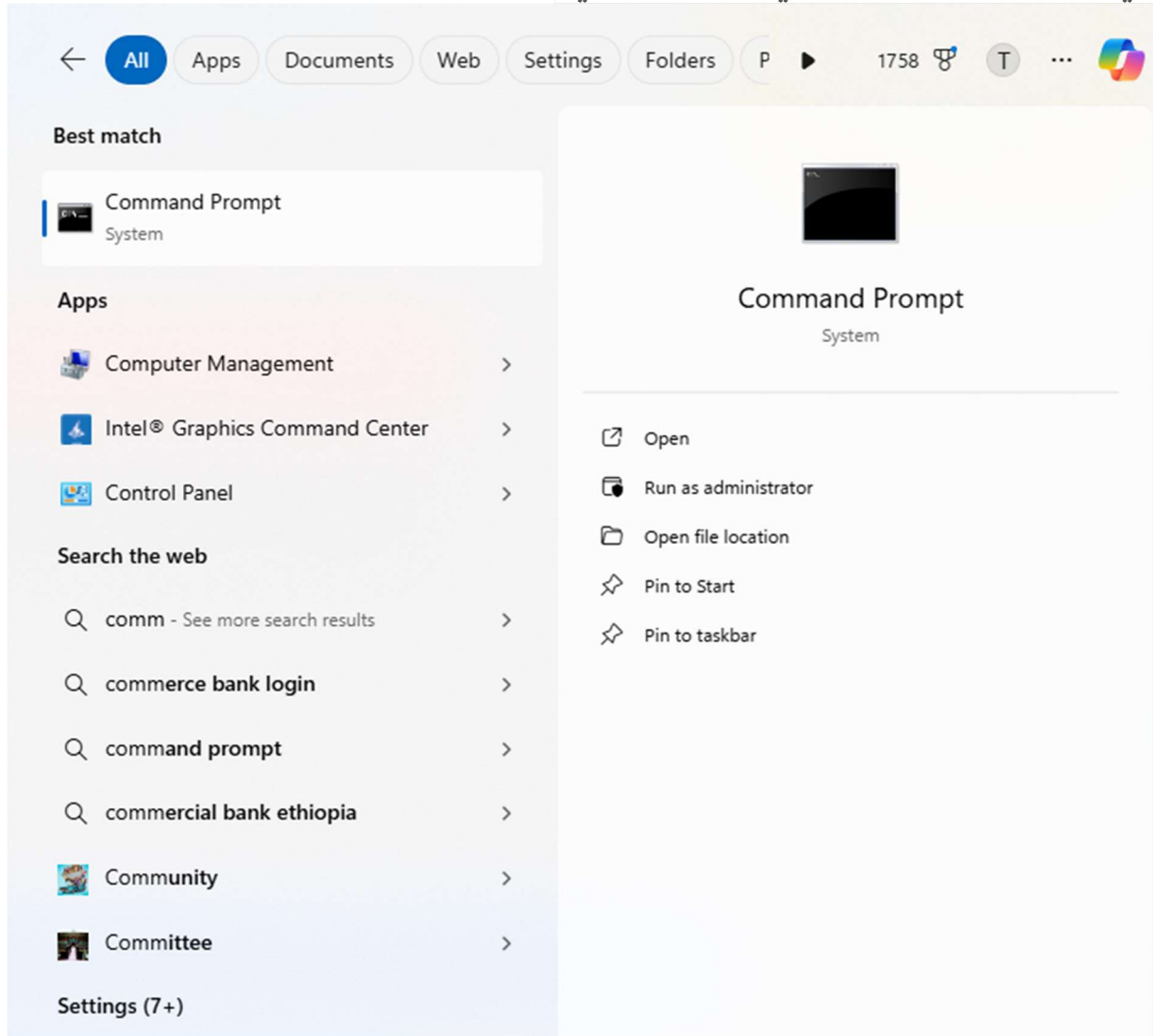
7. والآن وبعد الانتهاء من عملية التثبيت بنجاح (Setup Was Successful)، قم بالضغط على زر الإغلاق (Close) كما يظهر في الشكل التالي:



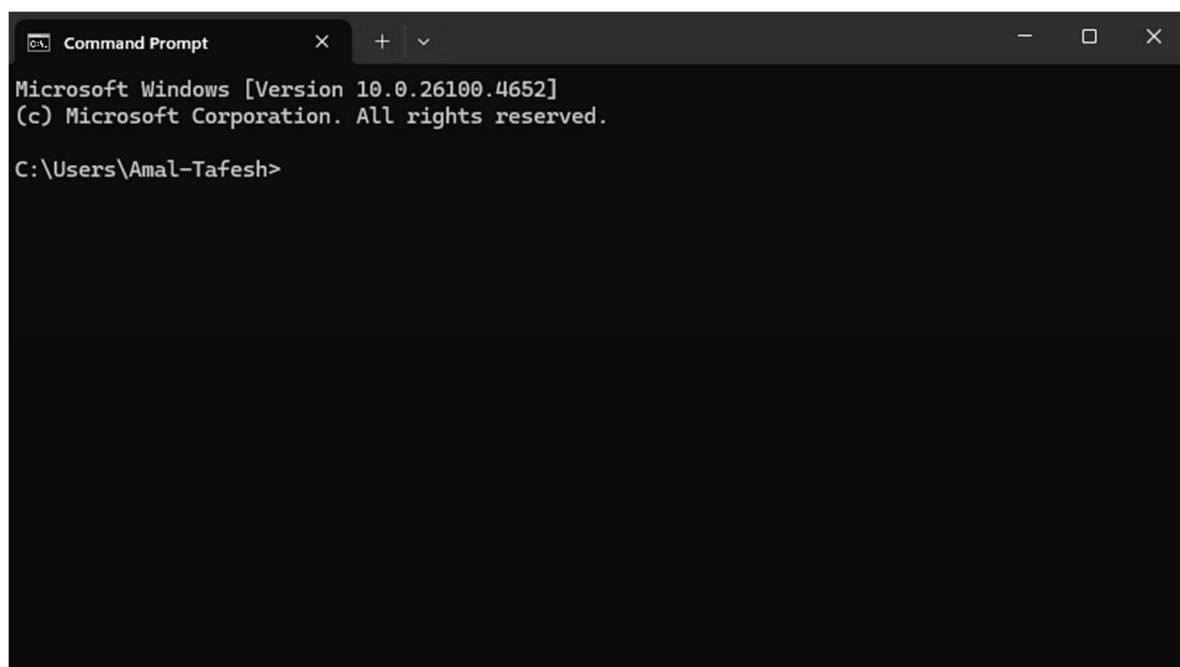
ثالثاً: التحقق من تثبيت مفسر لغة البرمجة بايثون (Python)
حتى نتأكد من نجاح عملية تثبيت مفسر لغة البرمجة بايثون (Python)
يجب علينا القيام بما يلي:

1. الذهاب الى القائمة الرئيسية في نظام التشغيل (windows) على جهاز الحاسوب

2. القيام بالبحث عن الأمر (Command Prompt) من خلال كتابة اسمه في شريط البحث كما في الشكل التالي:



3. عند ظهور الأمر (Command Prompt) ، وبعد عملية الضغط عليه تظهر لنا الشاشة التالية:

A screenshot of a Windows Command Prompt window. The title bar at the top reads "Command Prompt" with standard window controls (minimize, maximize, close). The main area of the window is black with white text. The text displayed is: "Microsoft Windows [Version 10.0.26100.4652]" followed by "(c) Microsoft Corporation. All rights reserved." and the current directory path "C:\Users\Amal-Tafesh>".

```
Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amal-Tafesh>
```

4.الان دعنا نكتب كلمة (Python) ثم الضغط على زر الإدخال Enter ،
فتظهر لدينا الشاشة التالية:

```
Command Prompt - Python
Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amal-Tafesh>Python
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

حيث سوف يظهر لنا رقم الاصدار المثبت الخاص بمفسر لغة البرمجة
بايثون (Python) وهنا هو 3.13.5

،(Python 3.13.5)والان بعد أن تحققنا من تثبيت مفسر لغة البرمجة بايثون)

(بعد الإشارة >>> ثم Exit دعنا نخرج من خلال كتابة كلمة الخروج)
(كما يظهر لنا في الشكل التالي: Enter: الضغط على زر الإدخال)

```
Command Prompt
Microsoft Windows [Version 10.0.26100.4652]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Amal-Tafesh>Python
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit

C:\Users\Amal-Tafesh>
```

6. والآن لنخرج من شاشة موجه الأوامر (Command Prompt) والعودة الى سطح المكتب من خلال كتابة الامر (exit)

مُحرّرات النصوص وبيئات التطوير المتكاملة

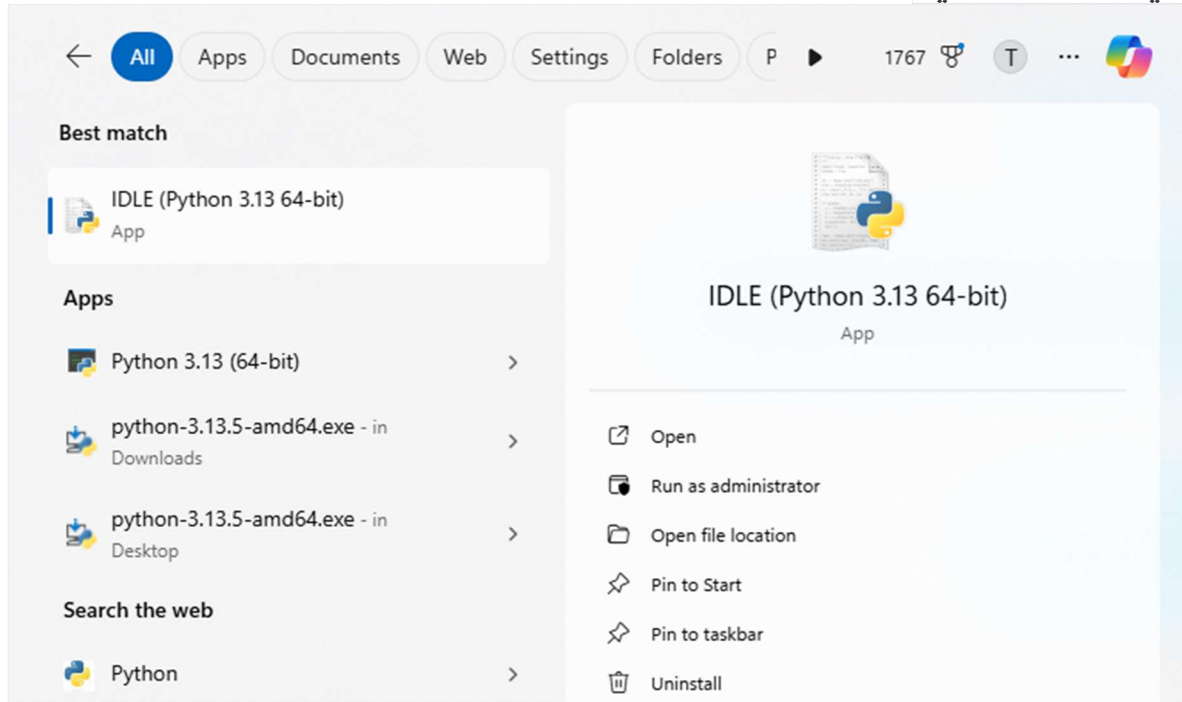
(Text Editors and Integrated development Environment)

إن إعداد البرامج و تطوير التطبيقات بالاعتماد على لغة البرمجة بايثون يكون عادة عبر محررات وبيئات تطوير متكاملة وهي كثيرة للغاية، إذ يفضل البعض استخدام محرر نصي بسيط مثل إيماكس [Emacs](#) أو [Vim](#) أو Gedit ، وجميع هذه المحررات قابلة لتوسيع إمكانياتها مثل ميزة إبراز صيغة الجملة syntax highlighting والإكمال التلقائي (Code completion)، أما بالنسبة للمستخدم المحترف الذي يعمل في مشاريع كبيرة معقدة فيفضل استخدام بيئة تطوير متكاملة IDE.

فما هي بيئة التطوير المتكاملة (IDE)

تعتبر بيئة التطوير المتكاملة عبارة عن محرر نصوص مدعم بميزات إضافية وتتخصص أحياناً بلغة برمجة واحدة أو لغتين فقط وتساعد المبرمج على تتبع المشروع ككل حيث تقوم معظم هذه البيئات بالاحتفاظ بذاكرة تخزين مؤقت لأسماء الأصناف والتوابع والمتغيرات بحيث يمكن إكمالها تلقائياً بسرعة وتمتلك بعضها أدوات اكتشاف الأخطاء وتصحيحها وتكون مدعمة بتصحيح المنطق الفوري instant logic الذي يساهم في تسريع أداء المواقع والتطبيقات وزيادة الأمان.

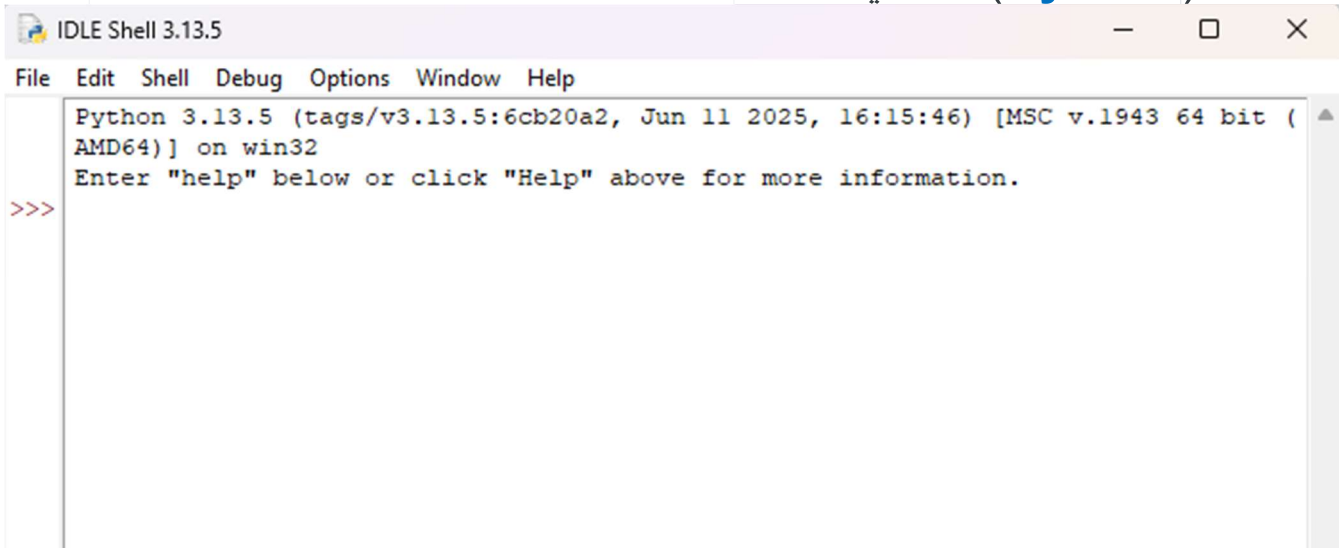
وهنا في لغة البرمجة الخاصة بنا (Python)، وبعد تثبيت المفسر الخاص بها، يتم تحميل بيئة التطوير الخاصة بلغة بايثون حيث تظهر لدينا في قائمة البرامج عند استعراض القائمة الرئيسية لنظام التشغيل ويندوز كما يظهر معنا في الشكل التالي:



كتابة برنامج بلغة البرمجة بايثون Python وحفظه:

والآن دعونا نكتب برنامجًا بسيطًا بلغة بايثون (Python) من خلال تنفيذ الخطوات التالية:

1. الذهاب الى القائمة الرئيسية لنظام التشغيل ويندوز وفتح بيئة التطوير والتعلم المتكاملة للغة البرمجة (Python) كما تعلمنا سابقاً.
2. سوف تظهر معنا الشاشة الخاصة ببيئة التطوير والتعلم المتكاملة للغة البرمجة (Python) كما في الشكل:

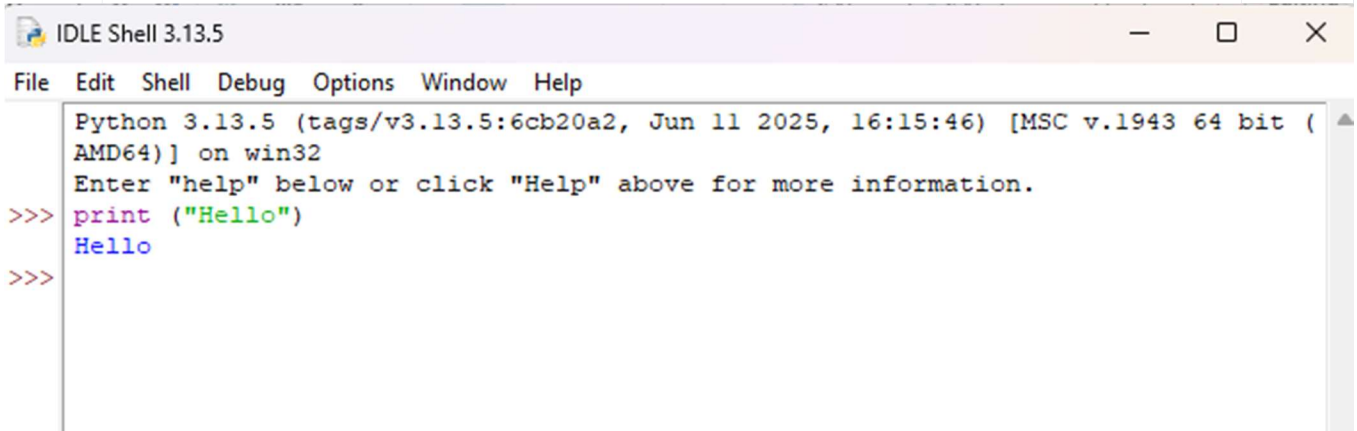


```
IDLE Shell 3.13.5
File Edit Shell Debug Options Window Help
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
```

3. لنقم بكتابة الجملة التالية بعد الإشارة >>>

>>> Print ("Hello")

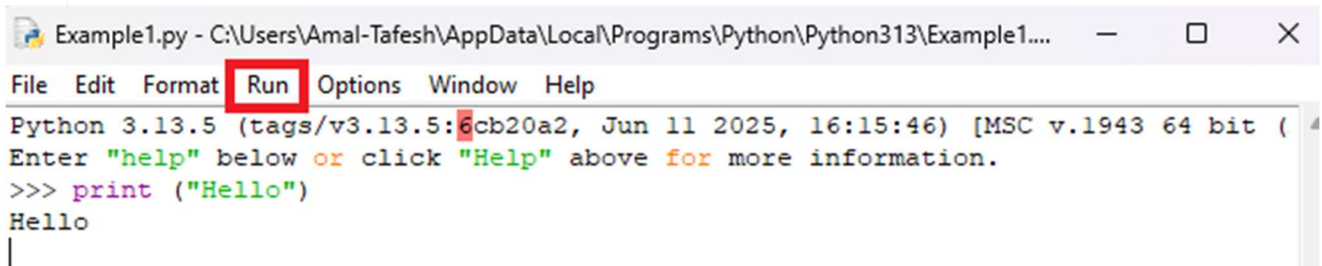
وبعدها نضغط مفتاح الإدخال (Enter)، عندها ستظهر لنا الشاشة التالية:



```
IDLE Shell 3.13.5
File Edit Shell Debug Options Window Help
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>> print ("Hello")
Hello
>>>
```

4. يمكننا الآن حفظ الملف من خلال الامر (Save as) الموجود في قائمة (File) بالاسم المناسب لطبيعة عمل البرنامج.

5. لغايات تنفيذ أي برنامج لا بد من حفظه أولاً حتى نستطيع أن نرى الامر (Run) ضمن قائمة الاوامر في شاشة بيئة التطوير والتعلم المتكاملة للغة البرمجة (Python) كما في الشاشة التالية:



```
Example1.py - C:\Users\Amal-Tafesh\AppData\Local\Programs\Python\Python313\Example1...
File Edit Format Run Options Window Help
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>> print ("Hello")
Hello
|
```

حيث تحتوي قائمة هذا الملف على الأمر (Run Module) الذي يمكن من خلاله تشغيل البرنامج.

جملة الإدخال Input()

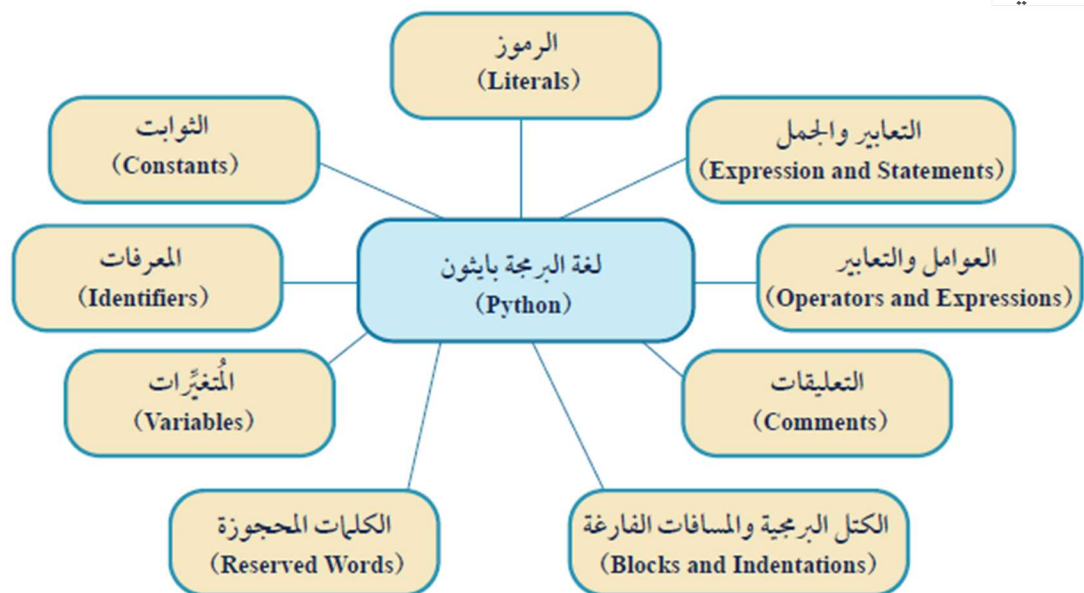
حتى يستطيع المستخدم إدخال البيانات أثناء تنفيذ البرنامج، لا بدّ من استخدام الدالة التي تحمل اسم **input()** إذ ان وجودها يعطي اوامر لمفسر بايثون بأن يكون في وضع الإستعداد و ينتظر من المستخدم القيام بغدخال البيانات من خلال لوحة المفاتيح ومن ثم الضغط على مفتاح الإدخال (Enter) .

بعد الضغط على مفتاح الإدخال (Enter) يقوم مفسر بايثون على بإدخال ما تم إدخاله على الشكل الذي تم إدخاله به إمّا على شكل نص او على شكل عدد.

عناصر لغة البرمجة بايثون (Python)

حتى نستطيع كتابة برنامج كامل بلغة البرمجة بايثون (Python) ، لا بد لنا من التعرف الى العناصر والمكونات الأساسية لهذه اللغة.

إن الناظر الى أي برنامج مكتوب بلغة البرمجة بايثون (Python) لا بدّ من أن يرى العناصر التسعة الرئيسة للغة البرمجة بايثون والتي تتمثل في الشكل التالي:



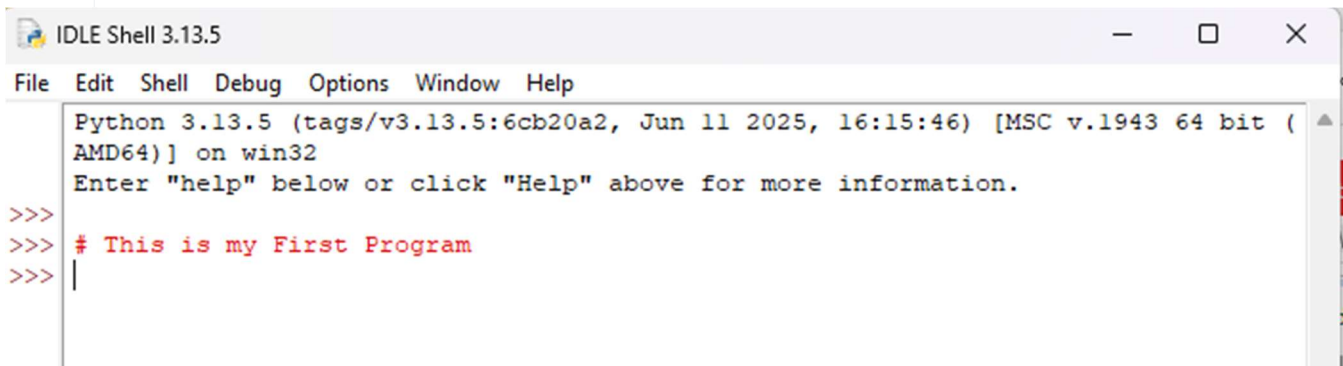
وفيما يلي توضيح حول كل عنصر من هذه العناصر :

1. التعليقات (Comments)

تعتبر من العناصر الاختيارية عند كتابة البرنامج، إذ أن وجودها لا يؤثر أثناء تنفيذ البرنامج، إلا أن وجودها في البرنامج مفيد لعدة أسباب:

أ. توثيق البرنامج : وجود التعليقات في البرنامج تساعد في فهم الهدف الذي من أجله تم كتابة هذه الأسطر من الأوامر، وبالتالي يساهم في عمليات التعديل والتطوير للبرنامج ويسهل عملية مراجعة البرنامج بعد فترة زمنية طويلة من كتابته.

ب. تحسين مقروئية البرنامج : إن وجود التعليقات في البرنامج يساعد المبرمج أو غيره من المطورين إلى فهم البرنامج أثناء قراءته بشكل أوضح مما يسهل عملية التعديل والتطوير عليه. وحتى تستطيع كتابة أي تعليق في برنامج مكتوب بلغة البرمجة بايثون، كل ما عليك فعله هو كتابة الرمز # ومن ثم التعليق الذي تريد كما يظهر في الشكل التالي:



```
IDLE Shell 3.13.5
File Edit Shell Debug Options Window Help
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
>>> # This is my First Program
>>> |
```

2. المُعرِّفات (Identifiers)

ويطلق عليها أيضا مصطلح (المسميات) وهي عبارة عن الأسماء التي تُستخدم لتعريف كل ما هو موجود في البرنامج من عناصر نستعملها داخل البرنامج

من

متغيرات (Variables) وثوابت (Constants) وكائنات (Objects)

وغيرها من العناصر التي نتعامل معها.

وبحسب لغة البرمجة بايثون، هناك مجموعة من القواعد التي يجب الالتزام بها عند كتابة المعرفات هي :

1. أن يحتوي اسم المُعرّف على أحد الحروف الصغيرة (a-z) أو أحد الحروف الكبيرة (A-Z) أو الأعداد (0-9) أو الشرطة السفلية (_) التي تسمى underscore

2. أن يبدأ اسم المعرف بأحد الحروف الكبيرة (A - Z) أو أحد الحروف الصغيرة (a - z) أو الشرطة السفلية (_)

3. عدم بدء اسم المعرف بعدد

4. منع استخدام أي كلمة من الكلمات المحجوزة (سوف نتعرف عليها لاحقاً) عند وضع اسم المعرف.

وفيما يلي مجموعة من الأمثلة على المُعرّفات المقبولة وغير المقبولة في

| المُعرّفات المقبولة | المُعرّفات غير المقبولة |
|---------------------|-------------------------|
| Name | @Example |
| number | -name |
| Grade_ | return |
| _test1 | 2ER |

برنامج بايثون:

3. الكلمات المحجوزة (Reserved words)

هي مجموعة من الكلمات التي تم تحديدها مسبقاً في لغة البرمجة بايثون بحيث لا يمكن استخدامها كمعرفات أو أسماء متغيرات. وقد يتم تغيير الكلمات المحجوزة من إصدار الى إصدار آخر من لغة البرمجة، حيث يتم اضافة أو حذف بعض الكلمات.

ويمكنك الحصول على قائمة الكلمات المحجوزة من لغة البرمجة بايثون بالنسخة المثبتة عندك من خلال كتابة الأمر `help("keywords")` >>> كما في الشكل التالي:

```
IDLE Shell 3.13.5
File Edit Shell Debug Options Window Help
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
>>> # This is my First Program
>>>
>>> help("keywords")

Here is a list of the Python keywords. Enter any keyword to get more help.

False      class      from       or
None       continue  global     pass
True       def        if          raise
and        del        import     return
as         elif       in          try
assert     else       is          while
async      except     lambda     with
await      finally   nonlocal   yield
break      for       not

>>>
```


4. الثوابت (Constants)

هي عبارة عن قيم تبقى قيمتها ثابتة ولا تتغير أثناء تنفيذ البرنامج. وتنقسم الى نوعين:

أ. الثوابت الرمزية النصية

أي مجموعة من الحروف يتم كتابتها داخل علامتي الاقتباس مثل: "Jordan", "Hello", "البترء",

ب. الثوابت العددية

مجموعة من الأعداد التي تبدأ بالعدد (0) وتنتهي بالعدد (9) ويمكن أن يتم إضافة إشارة (+) للدلالة على أن العدد موجب أو الإشارة (-) للدلالة على أن الرقم سالب. مثل:

+1234 ، -20 ، 4357

والاختلاف بين الثابت العددي والثابت الرمزي أن الثابت الرمزي لا يمكن استخدامه في العمليات الحسابية.

5. المتغيرات (Variables)

هي عبارة عن رموز تدل على مكان يتم حجزه في الذاكرة (RAM) بهدف تخزين قيمة معينة فيها أثناء تشغيل البرنامج. وتعتبر القيمة التي نضعها في المتغير قيمة مؤقتة حيث أنه يتم حذفها من الذاكرة بمجرد أن يتوقف البرنامج. ويستخدم اسم المتغير للإشارة الى تلك القيمة.

مثال:

حتى نقوم بتعريف متغير يحتوي على قيمة 99 نحتاج الى إعطاء اسم للمتغير، على سبيل المثال X ، وتكتب على الشكل التالي: **X=99**

وهناك عدد من أنواع المتغيرات في لغة البرمجة بايثون منها:

- المتغيرات العددية (Numbers) وتنقسم الى أعداد صحيحة (Int) وأعداد عشرية (Float)
- المتغيرات النصية (Strings) ويمكن تعريفها من خلال علامات الاقتباس الفردية (') او المزدوجة (") أو الثلاثية (""")

- المتغيرات المنطقية (Booleans) وهي التي تحتل إما قيمة صواب (True) أو قيمة خطأ (False)
- المصفوفات ذات الحجم الثابت (Lists)
- المصفوفات ذات الحجم غير الثابت ولا تحتوي على قيم مكررة (Sets)
- الجداول التي تخزن البيانات بصورة مفاتيح (Keys) وقيم (Values) وتسمى القواميس (Dictionaries)



مثال:

Age = 16

المتغير Age نوعه متغير عددي

Name = "Ali"

المتغير Name نوعه متغير نصي

6. الرموز (Literals)

هي مجموعة الرموز التي يتم استخدامها في لغة البرمجة بايثون (Python) مثل:

أ.النصوص (String Literals)

أي مجموعة من الحروف والكلمات الموجودة داخل علامات الاقتباس مثل:

"I am good student"

"مرحبا"

ب.الحروف (Character Literals)

أي حرف من حروف اللغات، مثل:

"A"

"م"

ج.الأرقام (Numeric Literals)

مجموعة الأرقام المتعارف عليها

10 ، 33.00 ، 120 ، ...

د.الرموز المنطقية (Boolean Literals)

ويقصد بها الرموز التي تحمل القيم (صواب أو خطأ)

مثل

True x=

Y= False

7. التعابير والجمل (Expression and Statements)

التعبير هو عبارة عن سلسلة تتكون من واحد أو أكثر من القيم، المتغيرات، العوامل، والدوال والتي من خلال اجتماعها معاً تنتج قيمة معينة.

مثال: 2+3

بينما الجملة هي أصغر جزء من البرنامج يكون قابلاً للتنفيذ

مثال : z=2+3

8. الكتل البرمجية والمسافات الفارغة (Blocks and Indentations)

هي مجموعة من الأوامر التي يتم تنفيذها معًا كجزء من بنية تحكم معينة مثل الحلقات (for ,while) أو الشرط (If) . ممكن أن تكون الكتلة البرمجية جملة واحدة فقط أو أكثر

في بايثون، يجب أن تكون جميع الأوامر التي تنتمي إلى كتلة برمجية معينة لها نفس المسافة البادئة، أي تبدأ بنفس عدد المسافات (عادة 4) من بداية السطر.

مثال

```
x=70
if x > 50 :
    print ("Pass")
    |
```

لاحظ هنا، انه قبل الدالة Print() يوجد مسافة فارغة ، وهذه ليست من اختيار المبرمج عند كتابة الامر، وانما هي شيء اجباري عند كتابة أي كتلة برمجية. وليس هناك شرط على عدد معين من الفراغات (Space) في البرنامج كاملاً، انما يشترط ان تكون عدد المسافات نفسها في الكتلة البرمجية الواحدة.

وهناك مجموعة من القواعد التي يجب مراعاتها عند كتابة الجملة البرمجية:

1. حالة الاحرف : إذ أن لغة البرمجة بايثون تعتبر (Case sensitive) بالنسبة لحالة الحرف (احرف كبيرة أم صغيرة) أذ ان كلمة (Days) تختلف عن كلمة (days) فهما لا يشيران الى نفس المتغير أو نفس الدالة.

2. آلية اختيار الأسماء : يجب مراعاة القواعد التالية عند اختيار اسم كل من المتغيرات، الدوال ، الكتابة على الأسطر :-

أ. اسم المتغير : يجب ان يكون اسم المتغير بالأحرف الصغيرة مثل (salary) وإذا كان المتغير أكثر من كلمة، يجب ان يفصل بين الكلمات إشارة الشرطة السفلية (underscore) مثل : (first_name)

Salary =650

First_name = "Ali"

ب. اسم الدالة : تماماً كما هو الحال مع اسم المتغير ، يجب ان يكون اسم الدالة بالأحرف الصغيرة مثل (print) وإذا كان الدالة أكثر من كلمة، يجب ان يفصل بين الكلمات إشارة الشرطة السفلية (underscore)

ج. كتابة أكثر من جملة على السطر نفسه: في الوضع الاعتيادي، يتم كتابة كل جملة على سطر في لغة البرمجة بايثون، ولكن في حال أردنا كتابة أكثر من جملة على نفس السطر، يجب وضع فاصلة منقوطة بين كل جملة والتي تليها.
مثال:

X=10

Y =20

يمكن كتابتهم على نفس السطر على الشكل التالي:

X=10 ; y =20

د. كتابة أمر واحد على أكثر من سطر : في حال أردنا تقسيم الامر نفسه على أكثر من سطر، يجب وضع اشارة | في نهاية كل سطر حتي يفهم مفسر بايثون أن الأمر مكتوب على أكثر من سطر
مثال : اذا كان الدخل لكل من شهر آذار، نيسان وآيار كما يلي:

March_income =563

April_income =324

May_income =230

لإيجاد مجموع الدخل خلال الشهور الثلاثة ممكن من خلال كتابة الأمر على سطر واحد كما يلي:

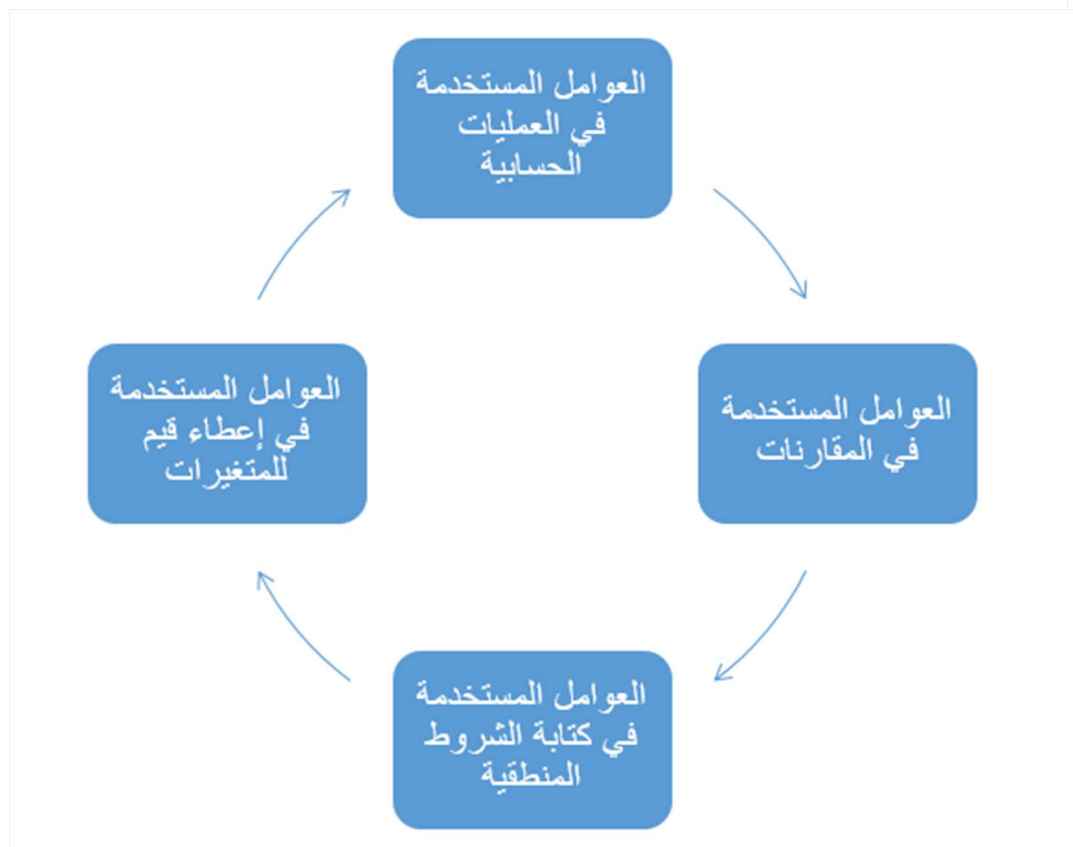
Total = March_income + April_income + May_income

او من خلال كتابة الامر على أكثر من سطر كما يلي:

**Total = March_income + **
**April_income + **
May_income

9.العوامل والتعابير (Operators and Expressions)

في لغة البرمجة بايثون ،يقصد بالعوامل (Operators) الرموز التي تستخدم لتنفيذ عمليات على القيم (مثل العمليات الحسابية، المقارنة، المنطقية، إلخ). ويتم تصنيف العوامل الى سبع مجموعات رئيسية من أهمها الأربعة التالية:



أما التعابير (Expressions) فهي أي جزء من البرنامج ينتج عنه قيمة، سواء كانت بسيطة كقيمة عددية أو معقدة تتضمن عوامل وعمليات.

وفيما يلي مجموعة من الجداول توضح كل مجموعة من هذه العوامل حسب التصنيف السابق:

أ. العوامل المُستخدمة في العمليات الحسابية (Arithmetic Operators)

| اسم العامل | الرمز | مثال توضيحي | الشرح |
|---------------------------------|-------|-------------|--|
| إضافة (Addition) | + | $x+y$ | إضافة قيمة y إلى قيمة x . |
| الطرح (Subtraction) | - | $x-y$ | طرح قيمة y من قيمة x . |
| الضرب (Multiplication) | * | $x*y$ | ضرب قيمة x في قيمة y . |
| القسمة (Division) | / | x/y | قسمة قيمة x على قيمة y . |
| باقي القسمة (Modulus) | % | $x\%y$ | إرجاع باقي قسمة قيمة x على قيمة y . |
| القوة (Exponentiation) | ** | $x**y$ | رفع قيمة x إلى أس بقيمة y . |
| القسمة التحتية (Floor Division) | // | $x//y$ | قسمة قيمة x على قيمة y ، وإرجاع أقرب عدد صحيح إلى الناتج (أقل من الناتج، أو يساوي الناتج). |

ب. العوامل المُستخدمة في المقارنات (Comparison Operators)

| اسم العامل | الرمز | مثال توضيحي | الشرح |
|-------------------------|-------|-------------|----------------------------|
| يساوي (Equal to) | == | $x==y$ | هل قيمة x تساوي قيمة y؟ |
| لا يساوي (Not equal to) | != | $x!=y$ | هل قيمة x لا تساوي قيمة y؟ |
| أكبر من (Greater than) | > | $x>y$ | هل قيمة x أكبر من قيمة y؟ |
| أصغر من (Less than) | < | $x<y$ | هل قيمة x أصغر من قيمة y؟ |

| | | | |
|---|----|--------|--------------------------------------|
| أكبر من أو تساوي (Greater than or equal to) | >= | $x>=y$ | هل قيمة x أكبر من قيمة y أو تساويها؟ |
| أصغر من أو تساوي (Less than or equal to) | <= | $x<=y$ | هل قيمة x أصغر من قيمة y أو تساويها؟ |

ج. العوامل المُستخدمة في كتابة الشروط المنطقية (Logical Operators)

| اسم العامل | الرمز | مثال توضيحي | الشرح |
|---------------|-------|--------------------|--|
| (Logical AND) | and | $x \text{ and } y$ | - إرجاع (True) فقط إذا كانت قيمتي x و y هي True - إرجاع (False) إذا كانت قيمة x أو قيمة y أو قيمة كلٍّ منهما (False). |
| (Logical OR) | or | $x \text{ or } y$ | - إرجاع (False) فقط إذا كانت قيمة x و قيمة y هي False - إرجاع (True) إذا كانت قيمة x أو قيمة y أو قيمة كلٍّ منهما (True). |
| (Logical NOT) | not | $\text{not } x$ | - إرجاع (True) إذا كانت قيمة x هي (False) وإرجاع (False) إذا كانت قيمة x هي (True). |

د.العوامل المُستخدمة في إعطاء قيم للمتغيرات (Assignment Operators)

| اسم العامل | الرمز | مثال توضيحي | الشرح |
|--|-------|-------------|---|
| الإسناد الأساسي (Basic Assignment) | = | $x=y$ | إعطاء x قيمة y . |
| الإضافة والإسناد (Addition Assignment) | += | $x+=y$ | زيادة قيمة x بمقدار قيمة y . |
| الطرح والإسناد (Subtraction Assignment) | -= | $x-=y$ | إنقاص قيمة x بمقدار قيمة y . |
| الضرب والإسناد (Multiplication Assignment) | *= | $x*=y$ | مضاعفة قيمة x بقيمة y من المرات. |
| القسمة والإسناد (Division Assignment) | /= | $x/=y$ | تخزين ناتج قسمة قيمة x على قيمة y في x . |
| باقي القسمة والإسناد (Modulus Assignment) | %= | $x%=y$ | تخزين باقي قسمة قيمة x على قيمة y في x . |
| القوة والإسناد (Exponentiation Assignment) | **= | $x**=y$ | تخزين قيمة x مرفوعة إلى أس بقيمة y في x . |
| القسمة والإسناد (Floor Division Assignment) | //= | $x//=y$ | تخزين ناتج x / y في x . |

أولوية العوامل وترابطها Operators Precedence and Associativity

كما تعلمنا في الرياضيات، هناك أولويات أثناء تنفيذ العمليات الحسابية، وذات الأمر يتم التعامل به أثناء تنفيذ البرنامج في لغة البرمجة بايثون، إذ يوجد هناك ترتيب للعوامل الحسابية والمنطقية التي يتم استخدامها. وفيما يلي جدولاً يمثل الأولويات في تنفيذ العوامل الحسابية والمنطقية وعوامل المقارنات من الأعلى إلى الأقل أولوية.

| الأولوية | اسم العامل | الرمز | الترباط |
|----------|--|-------|-----------------------|
| 1 | القوة (exponentiation). | ** | من اليمين إلى اليسار. |
| 2 | الجمع والاسناد (Unary plus). | + | من اليمين إلى اليسار. |
| | الطرح والاسناد (Unary minus). | - | |
| 3 | الضرب (Multiplication). | * | من اليسار إلى اليمين. |
| | القسمة (Division). | / | |
| | باقي القسمة (Modulus). | % | |
| | القسمة التحتية (Floor division). | // | |
| 4 | الجمع (Addition). | + | من اليسار إلى اليمين. |
| | الطرح (Subtraction). | - | |
| 5 | أكبر من (Greater than). | > | من اليسار إلى اليمين. |
| | أصغر من (Less than). | < | |
| | أكبر من أو يساوي (Greater than or Equal to). | => | |
| | أصغر من أو يساوي (Less than or Equal to). | <= | |
| 6 | يساوي (Equal to). | == | من اليسار إلى اليمين. |
| | لا يساوي (Not equal to). | != | |
| 7 | (Logical NOT). | not | من اليمين إلى اليسار. |
| 8 | (Logical AND). | and | من اليسار إلى اليمين. |
| 9 | (Logical OR). | or | من اليسار إلى اليمين. |