لغة برمجة ١

إشراف/ الإدارة العامة للمناهج

م/ ثامر عبدالله العتيبي

١٤٤١هـ

فعالية التعارف

١٠ دقائق

الهدف العام للحقيبة

تهدف هذه الحقيبة إلى إكساب المتدرب المعارف والمهارات الأساسية في برمجة تطبيقات سطح المكتب باستخدام لغة بايثون.

وصف الحقيبة

تقدم هذه الحقيبة مجموعة من المهارات والمعارف الأساسية لمفهوم البرمجة وأنواع لغاتها المختلفة وخطوات حل المسائل ومن ثم برمجة تطبيقات سطح المكتب باستخدام إحدى لغات البرمجة المشهورة (لغة بايثون Python) ويشمل ذلك أساسيات البرمجة، ومعرفة أنواع البيانات، وتعريف المتغيرات والتعامل مع العمليات (الحسابية والعلاقية والمنطقية) وجمل الاختيار الشرطية وتشمل: (جملة ifالبسيطة وجملة if-else والجملة الشرطية if-elif-else).

محتوى الحقيبة

الوحدة	عنوان الوحدة	زمن الوحدة (ساعة)
الأولى	مقدمة في البرمجة ولغاتها	١٠ ساعات تدريبية
الثانية	مقدمة إلى لغة بايثون	١٠ ساعات تدريبية
الثالثة	أنواع البيانات والمتغيرات	١٥ ساعة تدريبية
الرابعة	العمليات الحسابية	١٥ ساعة تدريبية
الخامسة	العمليات العلاقية و المنطقية	١٥ ساعة تدريبية
السادسة	الاختيار بالجمل الشرطية	١٥ ساعة تدريبية

الأهداف التفصيلية للحقيبة

١. يمثل الخوارزميات بخرائط التدفق في حل المسائل.

٢. يمثل المتغيرات في لغة بايثون.

٣. يستخدم المعاملات الحسابية في لغة بايثون.

٤. يستخدم الدوال الجاهزة في لغة بايثون.

٥. يكتب التعابير الشرطية والمنطقية في لغة بايثون.

٦. يستخدم الجمل الشرطية if، if-else، if-elif-else في لغة بايثون.

٧. يرسم المخططات الانسيابية للجمل الشرطية.

٨. يكتب برنامج بلغة بايثون.

مقدمة في البرمجة ولغاتها

الأهداف التفصيلية للوحدة

١. يحل المسائل باستخدام الخوارزميات.

٢. يمثل الخوارزميات بخرائط التدفق في المسائل.

٣. يتحكم في سير تنفيذ البرنامج.

الوقت المتوقع للتدريب على هذه الوحدة: ١٠ ساعات تدريبية.

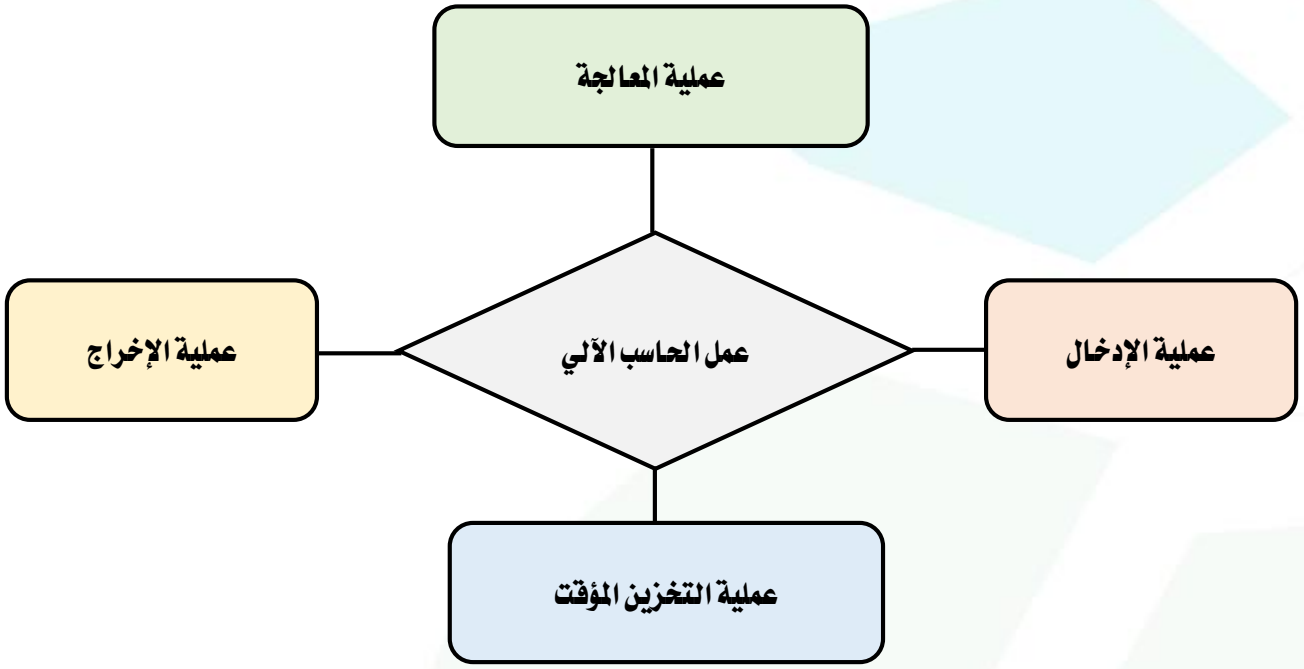
الوسائل التدريبية المساعدة:

- جهاز حاسب آلي.
- جهاز عرض (Data Show).

طريقة عمل جهاز الحاسب الآلي

يمر جهاز الحاسب الآلي بأربع عمليات رئيسية أثناء عمله وهي:

- عملية الإدخال.
- عملية المعالجة.
- عملية التخزين المؤقت.
- عملية الإخراج.



ماذا تعرف عن البرنامج؟

البرنامج عبارة عن مجموعة أو سلسلة من التعليمات والأوامر التي يحددها المبرمج بواسطة لغة برمجة معينة لتنفيذ مهمة أو حل مشكلة واستخراج النتائج في إطار زمني محدد.

تعليمات أساسية لابد أن توجد في أي برنامج

أي برنامج لابد ان يحتوي على الأساسيات التالية أو بعضها:

- المدخلات.
- المخرجات.
- الحساب.
- التنفيذ المشروط.
- التكرار.

لغات البرمجة

هي اللغات التي يتم كتابة البرامج من خلالها، وتنقسم لغات البرمجة إلى ثلاثة أقسام رئيسية هي:

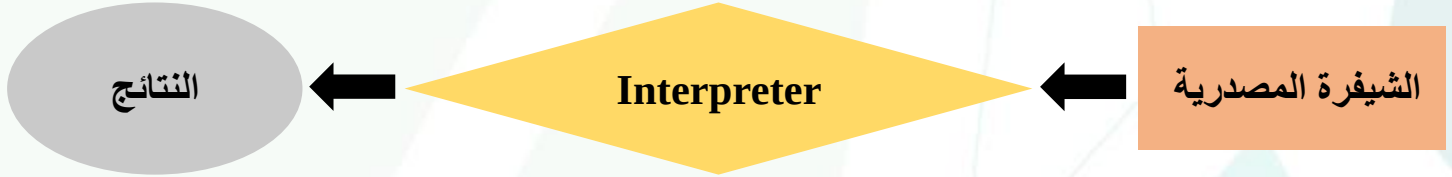
- لغة الآلة (Machine language).
- لغة التجميع (Assembly language).
- لغات عالية المستوى (High level language).

الفرق بين المترجم The compiler والمفسر The Interpreter

المترجم والمفسر يقومان بنفس العمل، وهو تحويل لغات عالية المستوى إلى لغة الآلة، لكن يختلفان في طريقة التنفيذ، من هنا سوف نبدأ في التعرف على كل واحد على حده؛ ليسهل عليك فهمهما بسهولة.

- المترجم The Compiler

- المفسر The Interpreter



الخوارزميات

هي عبارة عن مجموعة من الخطوات والتعليمات الرياضية والمنطقية المتسلسلة التي تساعد في حل مشكلة أو إنجاز مهمة معينة.

طريقة كتابة الخوارزمية:

- البداية.
- محتوى الخوارزمية.
- النهاية.

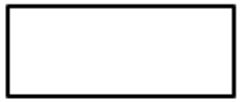
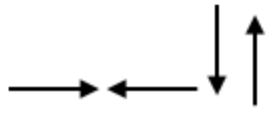
خرائط التدفق Flowchart

هي تمثيل بياني لتدفق البيانات يعتمد على رموز معروفة لتوضيح ترتيب وتسلسل الخطوات اللازمة لحل المشكلة.

- مميزات خرائط التدفق:
  - الاتصال.
  - تحليل فعال.
  - توثيق صحيح.

خرائط التدفق Flowchart

- عيوب خرائط التدفق:
- أسلوب معقد.
- إجراء تعديلات.
- رموز خرائط التدفق:

	**الأطراف** تستخدم في الأطراف حيث تبين (البداية والنهاية)
	**العمليات** يمثل جميع العمليات مثل: الضرب، الجمع، القسمة ... إلخ
	**المدخلات والمخرجات** يتم تمثيل المدخلات والمخرجات بهذا الشكل
	**القرارات** يتم تمثيل القرارات والشروط بهذا الشكل
	**الأسهم** تستخدم الأسهم لتوضيح تسلسل العمليات والانتقالات

حل المشكلات

عبارة عن عملية منظمة تتضمن سلسلة من الخطوات المتبعة لتحقيق الهدف أو إزالة المشكلة، ويمكن اتباع الخطوات التالية للحصول على حل للمشكلة:

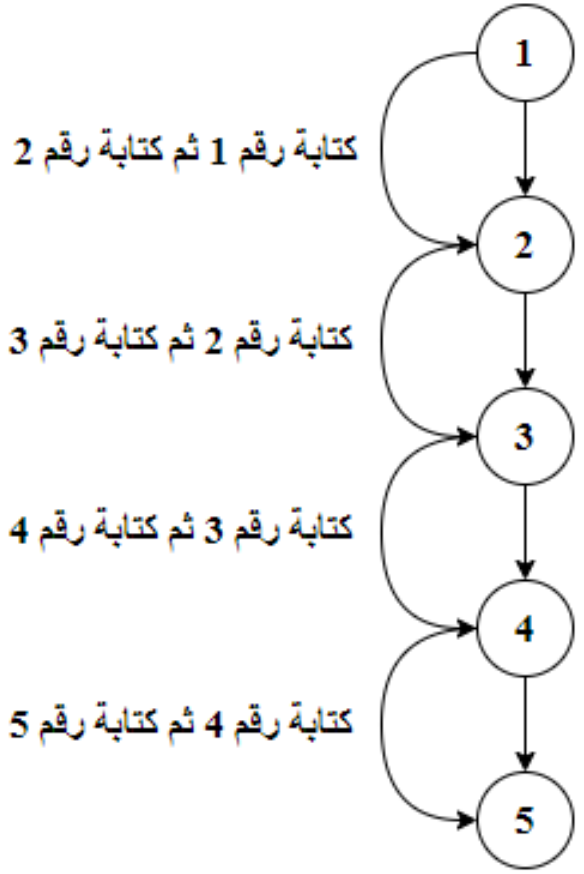
- تحديد المشكلة
- تمثيل المشكلة
- جمع المعلومات
- توليد الحل
- تنفيذ الحل
- تقويم الحل

جمل التحكم بتنفيذ البرامج

هناك عنصران أساسيان في برمجة برامج الحاسب الآلي هي: البيانات والهيكلة، وللتعامل مع البيانات لابد من فهم المتغيرات وأنواع البيانات، أما بالنسبة للهيكلة لابد من فهم صيغ وتركيبات التحكم. حيث يمكن التحكم في تنفيذ برامج الحاسب الآلي وفق ثلاث معايير أساسية:

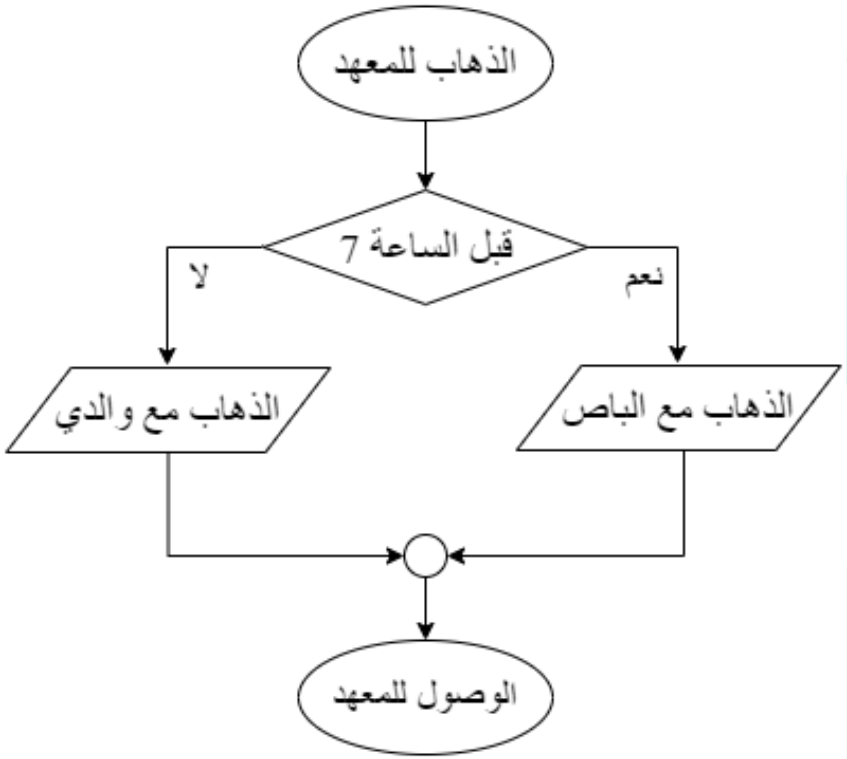
- التسلسل sequence.
- الاختيار selection.
- التكرار iteration.

التسلسل sequence



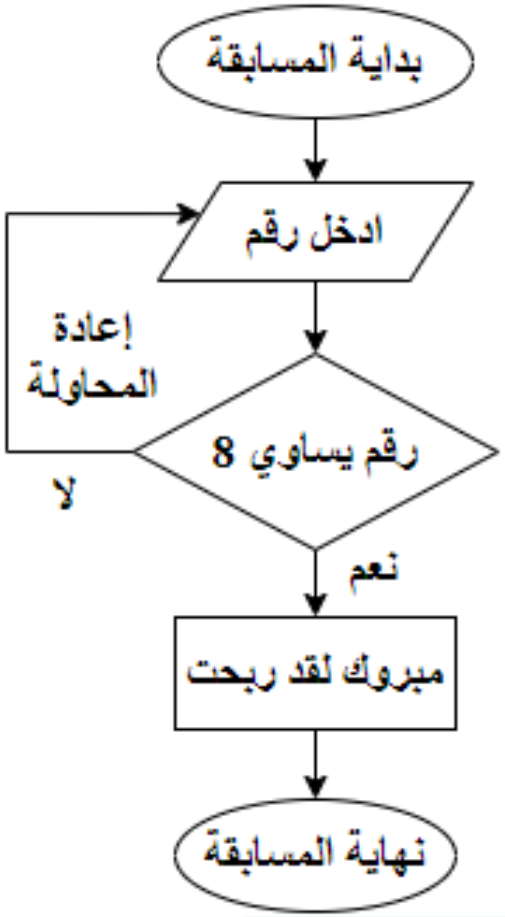
يتم تنفيذ البرنامج وفق مجموعة أوامر متسلسلة، حيث كل أمر يتبع الأمر الذي يليه ولا يمكن تنفيذ أي أمر قبل الأخر حسب التسلسل.

الاختيار selection



يتم تنفيذ البرنامج من خلال تحديد خيارين فقط ولا يتم إكمال البرنامج إلا بعد تحقق أحد الخيارين ثم يتم إكمال عمل البرنامج وفق التعليمات المدخلة.

التكرار iteration



هو تكرار عدد من الأوامر أو الأحداث المحددة من المبرمج لتنفيذ جزء أو كامل البرنامج، حيث لا يتم التنفيذ إلا بعد إكمال التكرار.

مقدمة إلى لغة بايثون

الأهداف التفصيلية للوحدة

١. يثبت لغة بايثون على جهاز الحاسب الآلي.

٢. يستخدم محرر الكتابة في لغة بايثون.

٣. يكتب برنامج باستخدام الدالة.

٤. يكتب التعليقات في لغة بايثون.

الوقت المتوقع للتدريب على هذه الوحدة: ١٠ ساعات تدريبية.

الوسائل التدريبية المساعدة:

- جهاز حاسب آلي.
- جهاز عرض (Data Show).
- اتصال إنترنت.
- محرر كتابة Python.

مدخل إلى لغة بايثون

- ماهي لغة بايثون Python؟
- مميزات لغة بايثون.
- استخدامات لغة بايثون.
- ماذا يقصد ببرمجة كائنية التوجه.
- أفضل محررات لغة بايثون.

تثبيت لغة بايثون على الجهاز

للتأكد من وجود بايثون على الجهاز:

- فتح موجه الأوامر cmd.
- كتابة الأمر التالي: pythonثم الضغط على مفتاح Enter.

الرسالة التالية تعني وجود بايثون على الجهاز.

```
C:\Users\Tam>python
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit
 (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
```

تثبيت لغة بايثون على الجهاز

في حالة عدم وجود بايثون ،يتم تثبيت بايثون من خلال أحد الطريقتين التالية:

أولًا: استخدام أداة الحزم في النظام ويندوز.

ثانيًا: تحميل محرر لغة بايثون من موقع :Python

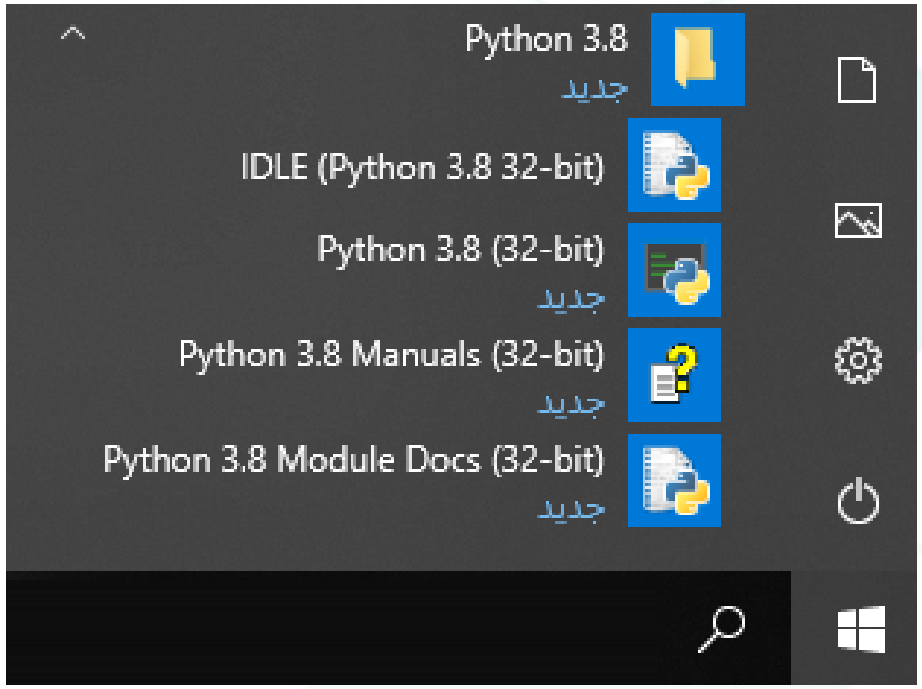
تثبيت لغة بايثون على الجهاز

يتم تثبيت محرر لغة بايثون على الجهاز بالخطوات التالية:

تثبيت لغة بايثون على الجهاز

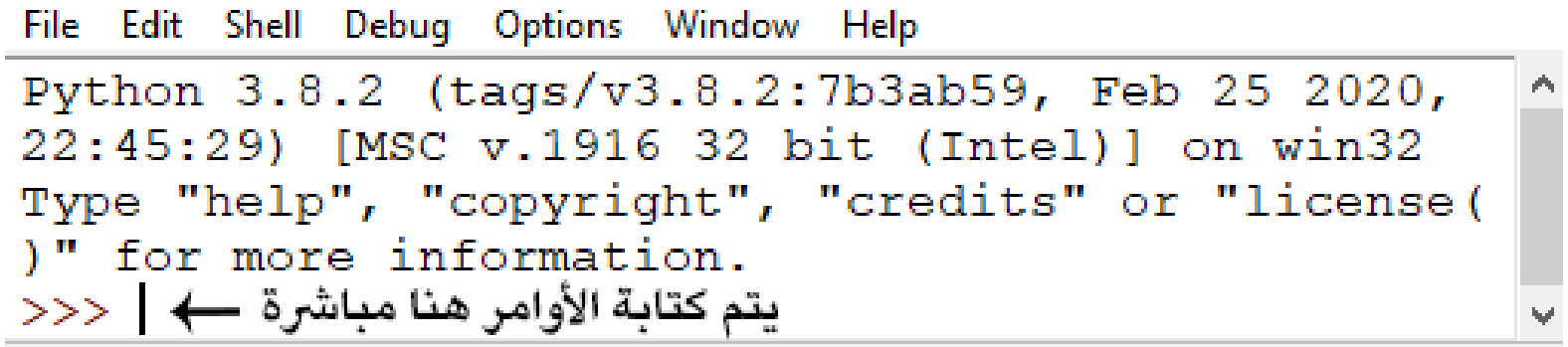
تشغيل محررکتابة بايثون Python IDLE:

- الضغط على قائمة ابدأ.
- البحث عن مجلد Python 3.8.
- اختيار IDLE Python 3.8.



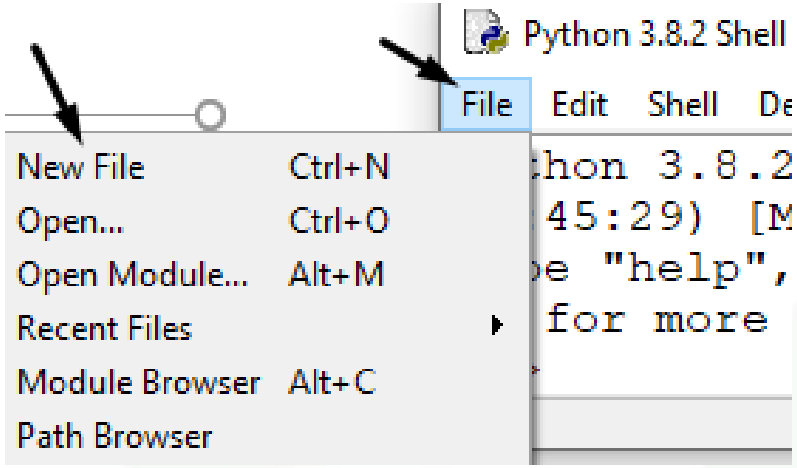
التعرف على بيئة محرركتابة بايثون Python IDLE

واجهة الـ Shellفي محرر IDLE-Python:

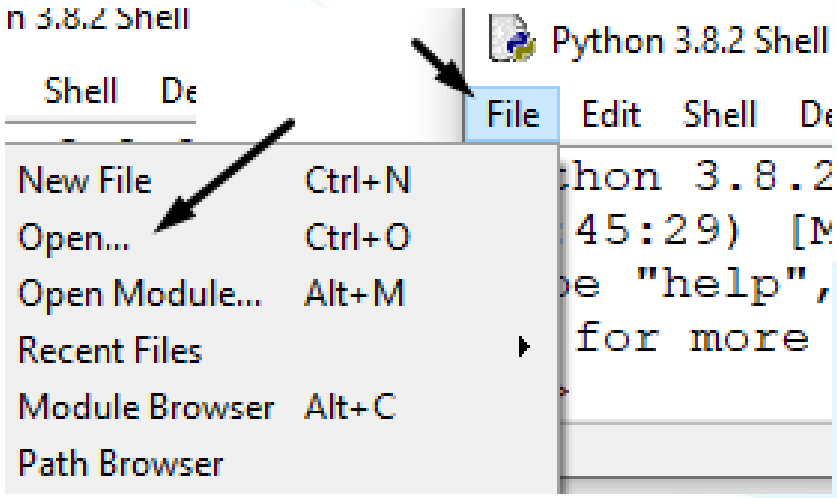


فتح ملف جديد:

- فتح قائمة File.
- اختيار New File.



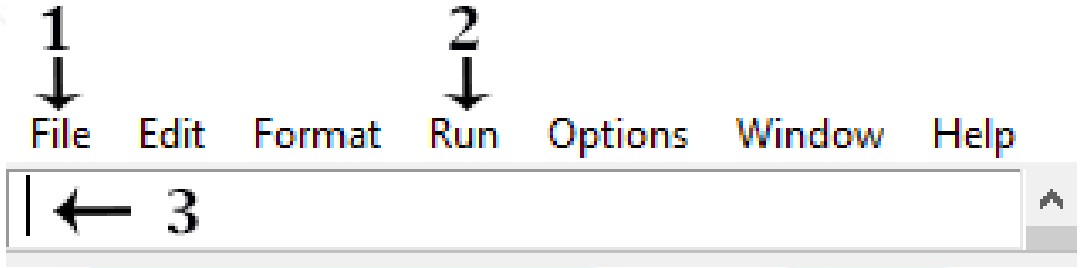
التعرف على بيئة محرركتابة بايثون Python IDLE



فتح ملف محفوظ:

- فتح قائمة File.
- اختيار Open.
- تحديد مكان الملف المحفوظ.

شرح أهم الخيارات في واجهة محررIDLE:

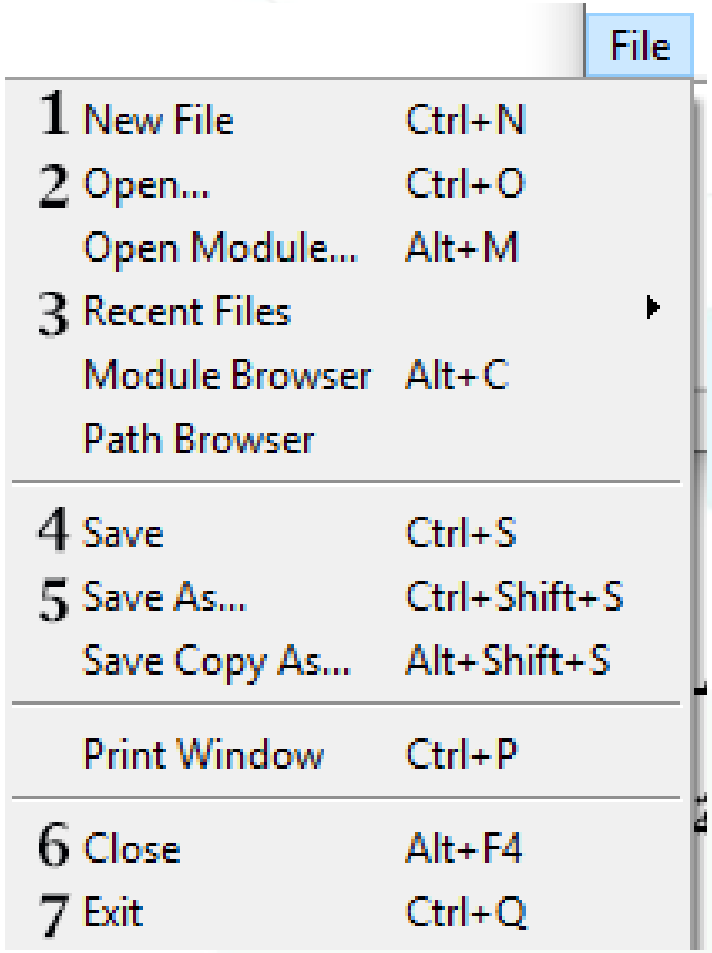


١. File: قائمة ملف، حيث يوجد فيها عدة خيارات منها: فتح، حفظ الملف.

٢. Run: تسمح بتجربة البرنامج بعد الانتهاء من كتابة الكود، بالإضافة لتشغيل Shell.

٣. Coding Place: مساحة العمل، حيث يتم من خلالها كتابة الشيفرة البرمجة.

التعرف على بيئة محرركتابة بايثون Python IDLE



شرح قائمة ملف:

١. إنشاء ملف جديد.

٢. فتح ملف محفوظ.

٣. فتح آخر الملفات المستخدمة.

٤. حفظ الملف.

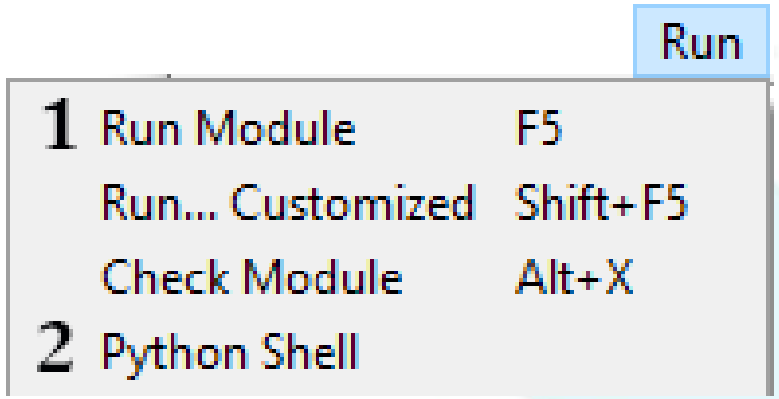
٥. حفظ الملف باسم.

٦. إغلاق الملف.

٧. إغلاق المحرر.

التعرف على بيئة محرركتابة بايثون Python IDLE

شرح قائمة تشغيل:



١. تنفيذ ملف البرمجة.

٢. فتح الـ Shell.

لاحقة حفظ الشيفرة البرمجية في لغة بايثون:

يتم حفظ ملف الشيفرة البرمجية في لغة بايثون باللاحقة (.py).

أساليب كتابة الشيفرة البرمجية:

- كتابة كل أمر برمجي في سطر مستقل.

```
print('TVTC') ←1
print(2002)   ←2
```

- كتابة عدة أوامر برمجية في سطر واحد وذلك من خلال الفصل بينهم بعلامة (;).

```
print('TVTC'); print(2002)
      ↑                ↑
      1                2
```

ماهي الدالة:

عبارة عن مجموعة من الأوامر تعمل مع بعض وفق تسلسل معين، وتنفذ عند استدعائها من المبرمج، وفي لغة بايثون الدوال تنقسم لقسمين: دوال جاهزة وسوف نتعرف عليها في المواضيع القادمة ودوال يتم إنشائها بواسطة المبرمج وسوف نتعرف عليها لاحقًا.

تعريف القيم النصية في لغة بايثون:

يتم تعريف القسم النصية باستخدام علامة التنصيص (الفردية أو المزدوجة).

```
'Hello, World!'
"Hello, World!"
'2020'
"1441"
```

الكلمات المحجوزة في بايثون:

هي كلمات لا يمكن استخدامها في لغة بايثون عند تعريف متغير أو كلاس أو دالة أو كائن

return	if	def	not	exec	and
finally	import	del	or	pass	assert
while	except	elif	try	raise	break
global	continue	else	is	from	class
yield	lambda	in	for	with	print

كتابة اول برنامج باستخدام دالة ()print:

دالة ()Print عبارة عن دالة جاهزة تقوم بطباعة رسالة معينة على الشاشة.

١. print: اسم الدالة.

٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).

٣. value: القيمة المطبوعة سواء: (نصية أو رقمية .... إلخ)

دالة ()print

معاملات تستخدم مع دالة ()print:

- طباعة أكثر من ناتج دالة ()print في سطر واحد.

```
print(value,end='separator')
```

1 2 3 4 5 6 7 8

١. print: اسم الدالة.

٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).

٣. value: القيمة المطبوعة سواء: (نصية أو رقمية .... إلخ)

٤. (,): علامة الفاصلة توضع بين القيمة وكلمة end.

٥. end: اسم المعامل المستخدم.

٦. (=): علامة يساوي توضع بين المعامل وعلامة تنيص الفاصل.

٧. (' '): علامة التنصيص يمكن استخدام المفردة أو المزدوجة؛ لتحديد الفصل المطلوب.

٨. separator: الفاصل المطلوب استخدام.

دالة ()print

- **تحديد فاصل محدد بين القيم المطبوعة في دالة ()print.**

```
print(value,sep="separator")
```

١. print: اسم الدالة.
٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).
٣. value: القيمة المطبوعة سواء: (نصية أو رقمية .... إلخ)
٤. (,): علامة الفاصلة توضع بين القيمة وكلمة end.
٥. sep: اسم المعامل المستخدم.
٦. (=): علامة يساوي توضع بين المعامل وعلامة تنيص الفاصل.
٧. (' '): علامة التنصيص يمكن استخدام المفردة أو المزدوجة؛ لتحديد الفصل المطلوب.
٨. separator: الفاصل المطلوب استخدامه.

مهارات في دالة ()print

- **استخدام علامة التنصيص مع النصوص.**
  - استخدام علامة التنصيص المزدوجة مع المفردة

```
print('Hello, "world"')
```

  - استخدام علامة التنصيص المفردة مع المزدوجة.

```
print("I'm useing Python")
```

  - استخدام علامة التنصيص المفردة مع المفردة.

```
print('I\'m useing Python')
```

مهارات في دالة ()print

- استخدام علامة (+) في دالة ()print.
  - مع النصوص.

```
print('Hi'+'Python')
```

  - مع الأرقام.

```
print(5+5)
```

مهارات في دالة ()print

- استخدم علامة (*) في دالة ()print.
  - مع النصوص.

```
print('Hi'*3)
```

  - مع الأرقام.

```
print(3*3)
```

مهارات في دالة ()print

- **طباعة أكثر من سطر في دالة** ()print.

باستخدام علامة التنصيص الثلاثية ("") تستطيع طباعة أكثر.

١. print: اسم الدالة.

٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).

٣. ("): علامات التنصيص الثلاثة، قبل وبعد الأسطر، (يستخدم مفردة أو مزدوجة)

٤. line: مكان كتابة الأسطر.

مهارات في دالة ()print

- **وضع مسافة Tab(٥ مسافات).**

يمكن إضافة مسافة Tabبين القيم الموجودة في دالة ()print، وذلك باستخدام (\t)

```
print('value1\tvalue2')
```

١. print: اسم الدالة.

٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).

٣. (' '): علامة التنصيص يمكن استخدام المفردة أو المزدوجة؛ لتحديد الفصل المطلوب.

٤. Value1: القيمة المطبوعة سواء: (نصية أو رقمية .... إلخ)

٥. (\t): الشرطة المعكوسة مع حرف (t) بين القيمتين المطلوب وضع المسافة بينهما.

٦. Value2: القيمة المطبوعة سواء: (نصية أو رقمية .... إلخ)

مهارات في دالة ()print

- **طباعة أكثر من قيمة في دالة** ()print.

يتم طباعة اكثر من قيمة في دالة ()print من خلال الفصل بين القيم بعلامة (،).

```
print(value1,value2,value3...etc)
  ↑   ↑   ↑   ↑   ↑
  1   2   3   4   5
```

١. print: اسم الدالة.

٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).

٣. value1: القيمة الأولى.

٤. (،): علامة الفاصلة توضع بين القيم.

٥. value2: القيمة الثانية، ويتم الاستمرار بوضع الفاصلة بين كل قيمة.

مهارات في دالة ()print

- **النزول سطر جديد في دالة** ()print.

يمكن طباعة القيم الموجودة في دالة ()printفي أكثر من سطر وذلك باستخدام (\n).

```
print('value1\nvalue2')
```

1 2 3 4 5 6

- داخل النصوص

١. print: اسم الدالة.
٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).
٣. (' '): علامة التنصيص يمكن استخدام المفردة أو المزدوجة، لتحديد الفصل المطلوب.
٤. Value1: القيمة المطبوعة سواء: (نصية أو رقمية .... إلخ)
٥. (\n): الشرطة المعكوسة مع حرف (n) بين القيمتين المطلوب وضع سطر بينهما.
٦. Value2: القيمة المطبوعة سواء (نصية أو رقمية .... إلخ)

مهارات في دالة ()print

- **النزول سطر جديد في دالة** ()print.

يمكن طباعة القيم الموجودة في دالة ()printفي أكثر من سطر وذلك باستخدام (\n).

```
print(value1,'\n',value2)
      ↑ ↑ ↑ ↑ ↑   ↑
      1 2 3 4 5   6
```

- خارج النصوص

١. print: اسم الدالة.
٢. (): قوس الدالة، (يفتح في البداية ويغلق في النهاية).
٣. Value1: القيمة المطبوعة سواء: (نصية أو رقمية .... إلخ)
٤. (,): علامة الفاصلة توضع بين القيمة وعلامة ('\n').
٥. ('\n'): الشرطة المعكوسة مع حرف (n) بين علامتي تنصيص بين القيمتين المطلوب وضع سطر بينهما.
٦. Value2: القيمة المطبوعة سواء (نصية أو رقمية .... إلخ)

كتابة التعليقات في لغة بايثون

تسمح لغة بايثون بإضافة التعليقات للشيفرة البرمجية، حيث يستخدم المبرمجون التعليقات في لغات البرمجة لشرح آلية عمل شيفرة معين أو توضيح شيفرة برمجية، حيث يتم عرض التعليق في كود المصدر دون تنفيذه عند تنفيذ البرنامج.

- **كتابة تعليق من سطرواحد.**

```
#Comments
↑  ↑
1  2
```

١. (#): رمز الشباك، يتم إضافته قبل كتابة التعليق، (بداية سطر التعليقات).

٢. Comments: التعليقات، حيث يتم كتابة التعليق المطلوب.

كتابة التعليقات في لغة بايثون

- كتابة تعليق من عدة أسطر.

```
''' ← 1
Comments ← 2
'''
```

١. (""): علامة التنصيص الثلاثية، تدرج في بداية ونهاية التعليق.

٢. Comments: التعليقات، يتم كتابتها بين علامتي التنصيص الثلاثية.

أنواع البيانات والمتغيرات

الأهداف التفصيلية للوحدة

١. يوضح أنواع البيانات في لغة بايثون.

٢. يستخدم الدوال في لغة بايثون.

٣. يمثل المتغيرات في لغة بايثون.

الوقت المتوقع للتدريب على هذه الوحدة: ١٥ ساعة تدريبية

الوسائل التدريبية المساعدة:

- جهاز حاسب آلي.
- جهاز عرض (Data Show).
- محرر كتابة Python

البيانات في لغة بايثون

البيانات عبارة عن مدخلات يطلبها البرنامج من المستخدم أو المبرمج للعمل بشكل صحيح أو تنفيذ مهمة معينة.

أنواع البيانات الأساسية في بايثون:

- الأرقام Numbers.
- النوع المنطقي Boolean type.
- متسلسلات حرفية Strings.
- القوائم Lists.
- المصفوفات Tuples.
- المجموعات Sets.
- القاموس Dictionary.

الأرقام Numbers

هي الرموز المستخدمة للتعبير عن الأعداد الصحيحة أو العشرية، مثل: (١ ، ٢ ، ٣، ١,٥ ، ٣,١٤).

الأرقام الصحيحة Integers:

هي الأعداد التي تكتب بدون استخدام الكسور أو الفواصل مثل (١ ، ٢ ، ٣)، في لغة بايثون يرمز لها بالاختصار (int)، وتمثل بدالة ()int.

الأرقام العشرية Floats:

هي الأعداد التي تكتب باستخدام الكسور أو الفواصل مثل: (٠.٥ ، ٢.٥ ، ٣.١٤)، وفي لغة بايثون يرمز لها بالاختصار (float)، وتمثل بدالة ()float.

النوع المنطقي و النصوص

النوع المنطقي: عبارة عن بيانات قد تحمل إحدى قيمتين (صح True) أو (خطأ False) فقط، ويمكن تمثيلها بالأرقام (1 للصح) و (0 للخطأ). ويرمز له بالاختصار (bool)، وتمثل بدالة ()bool.

```
print(5>8)
```

النتيجة:

```
False
```

النصوص: ترمز النصوص في لغة البرمجة للحروف والكلمات والجمل، فهي عبارة عن سلسلة من الحروف أو الكلمات التي ليس لها حجم محدد، وفي لغة بايثون يرمز لها بـ (str)، وتمثل بدالته ()str.

```
print('Python')
```

المتغيرات في لغة بايثون Variables

عبارة عن حاوية يتم إسناد القيمة لها من قبل المبرمج أو المستخدم، ويكون اسمها مختلف عن القيمة المسندة لها، وفي لغة بايثون يتم إسناد القيمة بطريقتين إما مباشرة من قبل المبرمج باستخدام (=) أو عن طريقة المستخدم بداله ()input.

تنبيه: في حالة عدم تحديد نوع المتغير عند الإسناد المباشر باستخدام (=)، فإن بايثون يقوم بتحديد النوع تلقائيًا حسب قيمته.

قواعد تسمية المتغير

في لغة بايثون يوجد عدة شروط لاختيار اسم المتغير:

- يجب إلا يحتوي اسم المتغير على مسافة.
- يمكن استخدام العلامة (_) للفصل بين الكلمات في اسم المتغير.
- يمكن التسمية بالعلامة (_) فقط.
- يمنع استخدام الكلمات المحجوزة في تسمية المتغير مثل: import.
- لغة بايثون تعتبر الحروف الصغيرة والكبيرة حروف مستقلة، ولذلك يمكن استخدام الحرفين A وa، كأسماء لمتغيرين مختلفين.
- اسم المتغير لابد أن يبدأ بحرف أو (_) ويمكن إضافة أرقام بعدهما مثل: A3، ولا يمكن بداية اسم المتغير برقم.

تعيين قيم المتغيرات في لغة بايثون

يتم تعيين قيم المتغيرات في لغة بايثون من خلال طريقتين:

اولًا: إسناد القيم بواسطة (=).

```
variable = value
   ↑       ↑     ↑
   1       2     3
```

١. variable: اسم المتغير

٢. (=): هي رمز الإسناد، بدونها لا يمكن إسناد القيم للمتغير.

٣. value: قيمة المتغير.

تعيين قيم المتغيرات في لغة بايثون

ثانيًا: إسناد القيم بواسطة المستخدم بدالة ()input.

```
Variable = input(description)
    ↑     ↑   ↑   ↑     ↑
    1     2   3   4     5
```

١. variable: اسم المتغير.

٢. (=): علامة الإسناد.

٣. input: دالة طلب المدخلات.

٤. (): علامة قوس دالة طلب المدخلات.

٥. description: موقع كتابة وصف المدخلات المطلوبة، (جزء اختياراي ليس ضروري كتابة وصف لعمل الدالة).

طباعة محتوى المتغيرات

يتم طباعة محتوى المتغيرات من خلال استخدام دالة print() واستدعاء اسم المتغير.

```
print(Variables)
```

1 2

١. دالة print() للطباعة.

٢. اسم المتغير المطلوب طباعة محتواه.

إسناد أكثر من قيمة بواسطة دالة input() مع دالة split()


```
Variable=input(Description).split('Separator')
```

1 2 3 4 5 6 7 8 9

variable: اسم المتغير.

(=): علامة الإسناد.

input: دالة طلب المدخلات.

(): علامة قوس دالة طلب المدخلات.

description: موقع كتابة وصف المدخلات المطلوبة، (الدالة تعمل بدون وصف).

(.): نقطة الفصل بين اسم المتغير والدالة.

split: دالة فصل المدخلات.

(): علامة قوس دالة طلب المدخلات.

separator: الفاصل المطلوب استخدامه، عند تركة فارغ سوف يكون الفاصل مسافة، عند إضافة فاصلة لابد من وضعه بين علامة تنصيص.

تحديد نوع المتغيرات قبل إسنادها

```
Variable = type(input(description))
```

1 2 3 4 5 6 7

١. variable: اسم المتغير.
٢. (=): علامة الإسناد.
٣. type: تحديد نوع بيانات المدخلات مثل: int، float.
٤. (): علامة قوس نوع البيانات.
٥. input: دالة طلب المدخلات من المستخدم.
٦. (): علامة قوس دالة طلب المدخلات.
٧. description: موقع كتابة وصف المدخلات المطلوبة.

معرفة أنوع البيانات في لغة بايثون

في لغة بايثون يمكن عرض نوع البيانات المدخلة من خلال دالة type().

```
type(variable)
```

1 2 3

١. type: اسم الدالة.

٢. (): قوس الدالة.

٣. variable: اسم المتغير.

تخزين النصوص في لغة بايثون

يتم تخزين النصوص في لغة بايثون في مصفوفة ذات بعد واحد من الحروف، حيث يتم حجز رقم في الذاكرة لكل حرف من حروف النص، وتبدأ المصفوفة من الرقم [٠]، حيث يكون في البرمجة بداية العد من الرقم [٠] وليس الرقم [١]، بناء على ذلك تستطيع طباعة أحرف معينة من النصوص معتمدًا على مكانها في المصفوفة.

تخزين كلمة TVTC في المصفوفة

	T	V	T	C	
للطباعة من البداية إلى النهاية	0	1	2	3	
	-4	-3	-2	-1	للطباعة من النهاية إلى البداية

طباعة حرف معين من النص

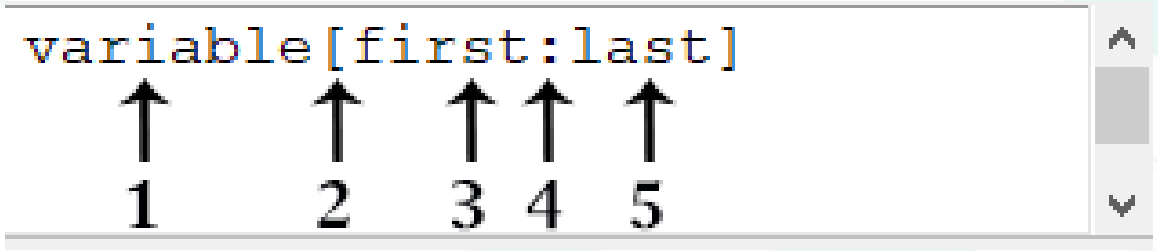
```
variable[location]
   ↑      ↑      ↑
   1      2      3
```

١. variable: اسم المتغير.

٢. [ ]: قوس المصفوفة

٣. location: رقم الحرف المطلوب في الذاكرة.

اقتطاع جزء معين من النص



١. variable: اسم المتغير.

٢. [ ]: قوس المصفوفة

٣. first: رقم أول حرف مطلوب اقتطاعه.

٤. (:): نقطتان راسيتان للفصل بين البداية والنهاية.

٥. last: رقم آخر حرف مطلوب اقتطاعه، (الرقم غير مشمول في الاقتطاع).

دوال تتعامل مع النصوص في لغة بايثون

دالة ()strip:

عبارة عن دالة جاهزة تقوم بحذف المسافات الموجودة في بداية ونهاية القيم المدخلة.

```
variable.strip()
   ↑     ↑  ↑  ↑
   1     2  3  4
```

١. variable: اسم المتغير.

٢. (.): نقطة الفصل بين اسم المتغير والدالة.

٣. strip: اسم الدالة.

٤. (): قوس الدالة.

دوال تتعامل مع النصوص في لغة بايثون

دالة capitalize():

عبارة عن دالة جاهزة تقوم بتحويل الحرف الأول إلى حرف كبير. هذه الدالة تعمل مع اللغة الإنجليزية.

```
variable.capitalize()
```

1 2 3 4

١. variable: اسم المتغير.

٢. (.): نقطة الفصل بين اسم المتغير والدالة.

٣. capitalize: اسم الدالة.

٤. (): قوس الدالة.

دوال تتعامل مع النصوص في لغة بايثون

دالة upper():

عبارة عن دالة جاهزة تقوم بتحويل حروف النص المدخل من حروف صغيرة إلى كبيرة.

```
variable.upper()
   ↑     ↑  ↑  ↑
   1     2  3  4
```

١. variable: اسم المتغير.

٢. (.): نقطة الفصل بين اسم المتغير والدالة.

٣. upper: اسم الدالة.

٤. (): قوس الدالة.

دوال تتعامل مع النصوص في لغة بايثون

دالة lower():

عبارة عن دالة جاهزة تقوم بتحويل حروف النص المدخل من حروف كبيرة إلى صغيرة.

```
variable.lower()
   ↑     ↑  ↑  ↑
   1     2  3  4
```

١. variable: اسم المتغير.

٢. (.): نقطة الفصل بين اسم المتغير والدالة.

٣. lower: اسم الدالة.

٤. (): قوس الدالة.

دوال تتعامل مع النصوص في لغة بايثون

دالة title():

عبارة عن دالة جاهزة تقوم بتحويل أول حرف من كل كلمة إلى حرف كبير.

```
variable.title()
   ↑     ↑ ↑  ↑
   1     2 3  4
```

١. variable: اسم المتغير.

٢. (.): نقطة الفصل بين اسم المتغير والدالة.

٣. title: اسم الدالة.

٤. (): قوس الدالة.

تحويل نوع المدخلات في لغة بايثون

دالة eval():

عبارة عن دالة جاهزة في بايثون تستخدم لحساب العمليات الحسابية.

```
variable=eval(input(description))
   ↑     ↑ ↑ ↑ ↑ ↑   ↑
   1     2 3 4 5 6   7
```

١. variable: اسم المتغير.
٢. (=): علامة الإسناد.
٣. eval: اسم الدالة.
٤. (): قوس دالة.
٥. input: دالة طلب المدخلات.
٦. (): علامة قوس دالة طلب المدخلات.
٧. description: موقع كتابة وصف المدخلات المطلوبة.

العمليات الحسابية

الأهداف التفصيلية للوحدة

١. ينفذ التعابير الرياضية في لغة بايثون.

٢. يستعمل الدوال الرياضية في لغة بايثون.

٣. يستدعي المكتبة مع دوالها في لغة بايثون.

٤. يطبق معاملات الإسناد الرياضية في لغة بايثون.

الوقت المتوقع للتدريب على هذه الوحدة: ١٥ ساعة تدريبية.

الوسائل التدريبية المساعدة:

- جهاز حاسب آلي.
- جهاز عرض (Data Show).
- محرر كتابة Python.

المعاملات الحسابية

تستطيع لغة بايثون حل العمليات الرياضية وتستخدم الترتيب المتفق عليه من قبل الرياضيين لحل تكل العمليات مع اختلاف بسيط في استخدام بعض الرموز وذلك ما سنوضحه في هذه الوحدة

يتم حل العمليات الحسابية في لغة بايثون بطرقتين:

- عن طريق Shellمباشرة.
- عن طريق كتابة الكود البرمجي بالمتغيرات والشروط أو استخدام دالة ()eval.

رموز المعاملات الحسابية

الرمز	الوظيفة	مثال حسابي
الجمع (+)	يقوم بجمع العناصر	3+2=5
الطرح (-)	يقوم بطرح العناصر	3-5=2
الضرب (*)	يقوم بضرب العناصر	2*3=6
القسمة (/)	يقوم بقسمة العناصر	3/6=2

الجمع (+)

طريقة تعامل لغة بايثون مع الجمع في Shell:

```
>>> 3+2
5
>>> 2+2
4
>>>
```

طريقة تعامل لغة بايثون مع الجمع في البرمجة:

```
1 N1=int(input('1st Num(Int): '))
2 N2=float(input('2nd Num(Float): '))
3 x=N1+N2
4 print(x)
5 print(N1+N2)
```

الطرح (-)

طريقة تعامل لغة بايثون مع الجمع في Shell:

```
>>> 3-5
-2
>>> 2.-1.5
0.5
>>>
```

طريقة تعامل لغة بايثون مع الجمع في البرمجة:

```
1 x=int(input('1st Num(Int): '))
2 y=float(input('2nd Num(Float): '))
3 r=x-y
4 print(r)
5 print(x-y)
```

الضرب (*)

طريقة تعامل لغة بايثون مع الجمع في Shell:

```
>>> 5*5
25
>>> 2.5*2.5
6.25
>>>
```

طريقة تعامل لغة بايثون مع الجمع في البرمجة:

```
1 N1=int(input('1st Num(Int): '))
2 N2=float(input('2nd Num(Float): '))
3 x=N1*N2
4 print(x)
5 print(N1*N2)
```

القسمة (/)

طريقة تعامل لغة بايثون مع الجمع في Shell:

```
>>> 6/2
3.0
>>> 3/3
1.0
```

طريقة تعامل لغة بايثون مع الجمع في البرمجة:

```
1 N1=int(input('1st Num(Int): '))
2 N2=float(input('2nd Num(Float): '))
3 x=N1/N2
4 print(x)
5 print(N1/N2)
```

باقي القسمة (%)

طريقة تعامل لغة بايثون مع الجمع في Shell:

```
>>> 5%2
1
>>> 5%1.5
0.5
>>>
```

طريقة تعامل لغة بايثون مع الجمع في البرمجة:

```
1 N1=int(input('1st Num(Int): '))
2 N2=float(input('2nd Num(Float): '))
3 x=N1%N2
4 print(x)
5 print(N1%N2)
```

معامل الأس (**)

طريقة تعامل لغة بايثون مع الجمع في Shell:

```
>>> 2**3
8
>>> 3**2
9
>>>
```

طريقة تعامل لغة بايثون مع الجمع في البرمجة:

```
1 N1=float(input('1st Num: '))
2 N2=float(input('2nd Num(Expo): '))
3 x=N1**N2
4 print(x)
5 print(N1**N2)
```

أولويات المعاملات الحسابية

الترتيب	العملية
1	العمليات داخل الأقواس
2	رفع الأس
3	الضرب والقسمة
4	الجمع والطرح

كتابة التعابير الحسابية

يقصد بالتعابير الحسابية هنا المعادلات الرياضية، في بايثون يتم دعم كتابة المعادلات الرياضية وتنفيذها وذلك من خلال تحويل وترتيب العمليات.

$$\frac{3 + 4x}{5} - \frac{10(y - 5)(a + b + c)}{x} + 9\left(\frac{4}{x} + \frac{9 + x}{y}\right)$$

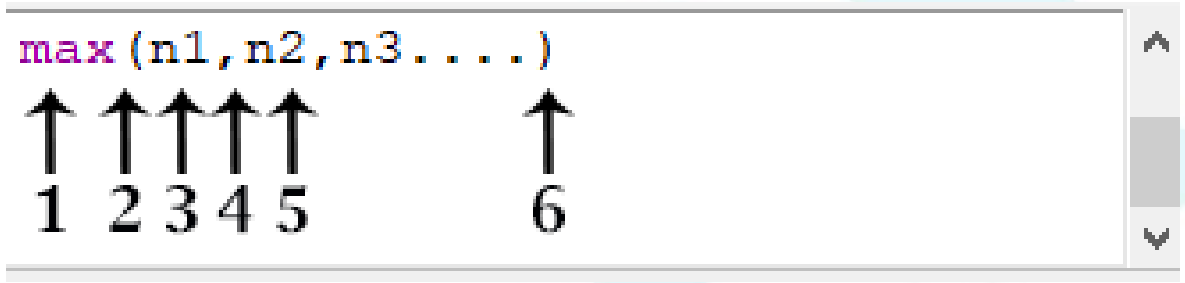
```
(3 + 4 * x) / 5 – 10 * (y - 5) * (a + b + c) / x + 9 * (4 / x + (9 + x) / y)
```

كما تلاحظ تم تحويل الكسور إلى أقواس وكتابة المقام والبسط بتعبير رياضي مبسط مبني على رموز العمليات الرياضية في بايثون.

دوال رياضية جاهزة في لغة بايثون

دالة ()max:

عبارة عن دالة تقوم باختيار القيمة الكبرى لمجموعة من القيم، سواء نصية أو رقيمة.

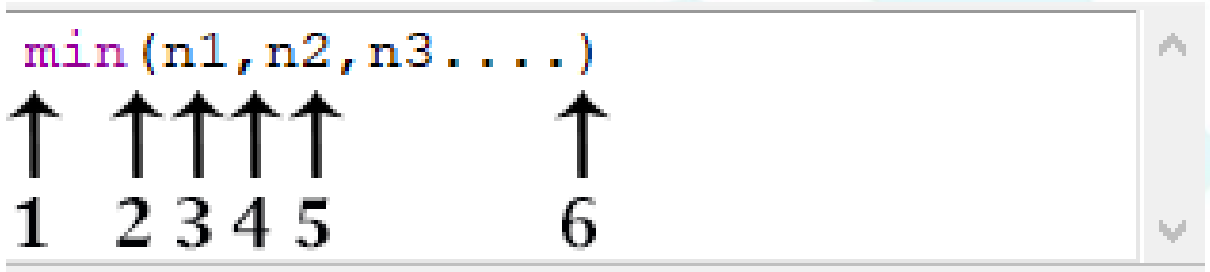


١. يتم كتابة max.
٢. فتح القوس.
٣. وضع القيمة الأولى أو المتغير.
٤. وضع علامة فاصلة.
٥. إضافة القيمة الثانية أو المتغير (الاستمرار بوضع الفاصلة والقيمة حسب المطلوب).
٦. إغلاق القوس.

دوال رياضية جاهزة في لغة بايثون

دالة ()min:

عبارة عن دالة تقوم باختيار القسمة الصغرى لمجموعة من القيم، سواء نصية أو رقمية.

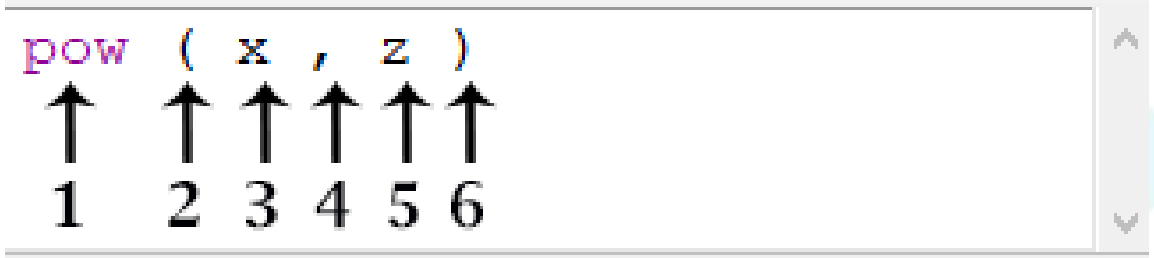


١. يتم كتابة min.
٢. فتح القوس.
٣. وضع القيمة الأولى أو المتغير.
٤. وضع علامة فاصلة.
٥. إضافة القيمة الثانية أو المتغير (الاستمرار بوضع الفاصلة والقيمة حسب المطلوب).
٦. إغلاق القوس.

دوال رياضية جاهزة في لغة بايثون

دالة ()pow:

عبارة عن دالة تقوم بحساب نتيجة رفع الأساس إلى قوة الأس.



كتابة اسم الدالة pow.

فتح القوس بعد اسم الدالة.

(x) مكان إضافة الرقم.

وضع الفاصلة.

(z) مكان إضافة رقم الأس.

إغلاق القوس بعد رقم الأس.

دالة ()round:

عبارة عن دالة تقوم بتقريب العدد العشري إلى أقرب عدد صحيح أو إلى أقرب خانة عشرية مطلوبة.

اولًا: التقريب إلى أقرب عدد صحيح.

١. كتابة اسم الدالة round.
٢. فتح علامة القوس مباشرة بعد اسم الدالة.
٣. كتابة الرقم أو المتغير المطلوب تقريبه.
٤. إغلاق القوس بعد الرقم.

دوال رياضية جاهزة في لغة بايثون

ثانيًا: التقريب إلى أقرب عدد عشري.

```
round ( x , z )
```

١. كتابة اسم الدالة round.
٢. فتح علامة القوس مباشرة بعد اسم الدالة.
٣. كتابة الرقم أو المتغير المطلوب تقريبه.
٤. إضافة فاصلة.
٥. كتابة عدد الأرقام العشرية المطلوب التقريب لها، علماً بأن الرقم يجب أن يكون رقمًا صحيحاً .
٦. إغلاق القوس بعد الرقم.

دوال رياضية جاهزة في لغة بايثون

دالة ()sqrt:

عبارة عن دالة تقوم بحساب الجذر التربيعي للرقم.

```
sqrt( x )
```

1 2 3

١. كتابة اسم الدالة sqrt.
٢. فتح القوس بعد اسم الدالة وإغلاقه بعد كتابة الرقم.
٣. (x) مكان إضافة الرقم.

المكتبات في لغة بايثون Modules

المكتبات في البرمجة عبارة مجموعة من المتغيرات، والدوال، يمكن تضمينها ضمن الشيفرة البرمجي الخاص، حيث قام مجموعة من المطورين بكتابتها ونشرها لتصبح متاحة للجميع والهدف منها تسهيل تنفيذ أمور معينة.

طرق استدعاء المكتبات في لغة بايثون:

- استدعاء المكتبة بدون دوال (import x).
- استدعاء المكتبة مع الدوال (from x import *).
- استدعاء دالة معينة من المكتبة (from x import name).

استدعاء المكتبة بدون دوال (import x)

في هذه الطريقة يتم استدعاء المكتبة بدون أي دالة.

```
import x
```

1 2

١. في بداية الشيفرة البرمجية كتابة كلمة importفي البداية.

٢. إضافة مسافة ثم كتابة اسم المكتبة مكان (x).

استدعاء المكتبة مع الدوال (from x import *)

في هذه الطريقة يتم استدعاء المكتبة مع جميع الدوال.

```
from x import *
```

1 2 3 4

١. في بداية الشيفرة البرمجية كتابة كلمة fromفي البداية.

٢. إضافة مسافة ثم كتابة اسم المكتبة مكان (x).

٣. إضافة مسافة ثم كتابة كلمة import.

٤. إضافة مسافة ثم إضافة علامة الضرب (*).

استدعاء دالة معينة من المكتبة (from x import name)

في هذه الطريقة يتم استدعاء دالة معين من المكتبة.

```
from x import name
  ↑   ↑    ↑     ↑
  1   2    3     4
```

١. في بداية الشيفرة البرمجية كتابة كلمة fromفي البداية.

٢. إضافة مسافة ثم كتابة اسم المكتبة مكان (x).

٣. إضافة مسافة ثم كتابة كلمة import.

٤. إضافة مسافة ثم كتابة اسم الدالة مكان (name).

معاملات الإسناد الرياضية

في لغة بايثون يوجد خمس معاملات إسناد رياضية.

المعامل	الوظيفة
=+	إضافة قيمة محددة لقيمة المتغير السابقة.
=-	إنقاص قيمة محددة من قيمة المتغير السابقة.
=*	ضرب قيمة محددة في قيمة المتغير السابقة.
=/	قسمة قيمة محددة على قيمة المتغير السابقة.
=%	حساب باقي القسمة لقيمة محددة من قيمة المتغير السابقة.

معامل (+=)

يقوم المعامل (+=) بإضافة قيمة محددة إلى قيمة متغير سابقة، حيث لا يقوم بحذف أو استبدال القيمة المسندة إلى المتغير مسبقًا.

```
a=float(input('Enter Num: '))
x=3
print(x)
x += a
print(x)
```

النتيجة:

```
Enter Num: 4.5
x:  3
New x:  7.5
```

معامل (=-)

يقوم المعامل (=-) بإنقاص قيمة محددة إلى قيمة متغير سابقة، حيث لا يقوم بحذف أو استبدال القيمة المسندة إلى المتغير مسبقًا.

```
a=float(input('Enter Num: '))
x=5
print(x)
x -= a
print(x)
```

النتيجة:

```
Enter Num: 2
x:  5
New x:  3.0
```

العمليات العلاقية والنطقية

الأهداف التفصيلية للوحدة

١. يكتب التعابير الشرطية والمنطقية في لغة بايثون.

٢. يستخدم دوال الأرقام العشوائية في لغة بايثون.

٣. يشرح أخطاء البرمجة في لغة بايثون.

الوقت المتوقع للتدريب على هذه الوحدة: ١٥ ساعة تدريبية.

الوسائل التدريبية المساعدة:

- جهاز حاسب آلي.
- جهاز عرض (Data Show).
- محرر كتابة Python.

معاملات المقارنة

عبارة عن معاملات تستخدم لمقارنة القيم مع بعضها، حيث تقوم بمقارنة القيم من حيث الأكبر أو الأصغر أو التساوي.

في لغة بايثون هنالك أربع معاملات للمقارنة:

١. معامل أصغر من (<).
٢. معامل أصغر من أو يساوي (=<).
٣. معامل أكبر من (>).
٤. معامل أكبر من أو يساوي (=>).

معامل أصغر من (<)

هو رمز رياضي يدل على عدم المساواة بين قيمتين. ويستخدم للدلالة على أن الطرف الأيسر في المتباينة أصغر من الطرف الأيمن.

١. كتابة قيمة الطرف الايسر مكان (A).

٢. إضافة مسافة قبل وبعد علامة المقارنة (<).

٣. كتابة قيمة الطرف الأيمن مكان (B).

معامل أصغر من أو يساوي (<=)

هو رمز رياضي يدل على عدم المساواة أو المساواة بين قيمتين. ويستخدم للدلالة على أن الطرف الأيسر في المتباينة أصغر من الطرف الأيمن أو يساويه.

١. كتابة قيمة الطرف الايسر مكان (A).

٢. إضافة مسافة قبل وبعد علامة المقارنة (<=).

٣. كتابة قيمة الطرف الأيمن مكان (B).

معامل أكبر من (>)

هو رمز رياضي يدل على عدم المساواة بين قيمتين. ويستخدم للدلالة على أن الطرف الأيسر في المتباينة أكبر من الطرف الأيمن.

١. كتابة قيمة الطرف الايسر مكان (A).

٢. إضافة مسافة قبل وبعد علامة المقارنة (>).

٣. كتابة قيمة الطرف الأيمن مكان (B).

معامل أكبر من أو يساوي (=>)

هو رمز رياضي يدل على عدم المساواة أو المساواة بين قيمتين. ويستخدم للدلالة على أن الطرف الأيسر في المتباينة أكبر من الطرف الأيمن أو يساويه.

١. كتابة قيمة الطرف الايسر مكان (A).

٢. إضافة مسافة قبل وبعد علامة المقارنة (=>).

٣. كتابة قيمة الطرف الأيمن مكان (B).

معاملات المساواة

عبارة عن معاملات مقارنة حيث تقوم بمقارنة القيم من حيث التساوي أو عدم التساوي.

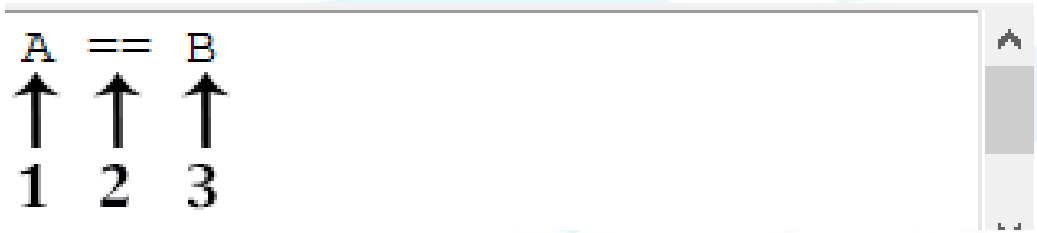
في لغة بايثون يوجد معاملين للمساواة:

١. **معامل المساواة (==)**

٢. **معامل عدم المساواة (=!)**

معامل المساواة (==)

عبارة عن معامل يقوم بمقارنة القيمة هل تساوي القيمة الأخرى، حيث يكون الجواب صح أو خطأ فقط.



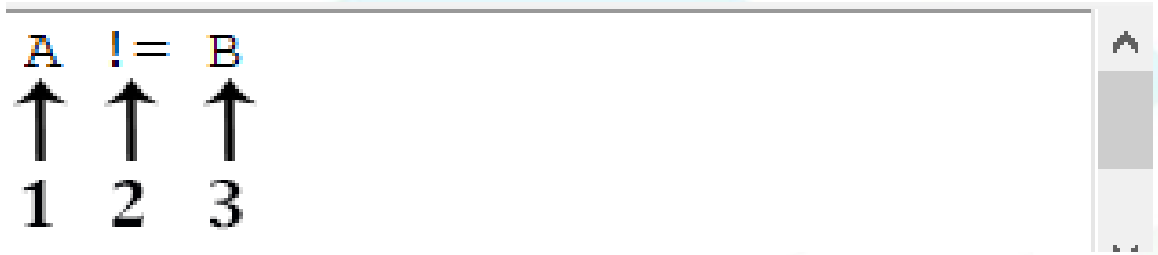
١. القيمة الأول تكون مكان (A).

٢. إضافة مسافة ثم علامة (==) ثم مسافة.

٣. القيمة الثانية تكون مكان (B).

معامل عدم المساواة (=!)

عبارة عن معامل يقوم بمقارنة القيمة هل هي مساوية للقيمة الأخرى، حيث يكون الجواب صح أو خطأ فقط.



١. القيمة الأول تكون مكان (A).

٢. إضافة مسافة ثم علامة (=!) ثم مسافة.

٣. القيمة الثانية تكون مكان (B).

المعاملات المنطقية في لغة بايثون

القيم المنطقية ترمز إلى الأشياء التي لا تحتمل أكثر من احتمالين وهما إما صح وإما خطأ، حيث تساعدنا في صناعة الشروط والقيود على شيء معين وبالتالي تمنحنا تحكماً أكبر في الشيفرة البرمجية.

المعاملات المنطقية في لغة بايثون تنقسم إلى ثلاث اقسام هي:

- **المعامل** and.
- **المعامل** or.
- **المعامل** not.

المعامل and

تعني الجمع بين شرطين.

```
A and B
↑   ↑   ↑
1   2   3
```

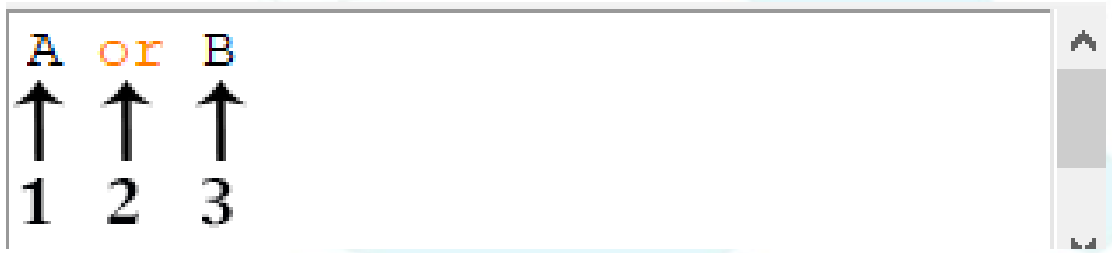
١. يتم كتابة الشرط الأول مكان (A).

٢. يتم إضافة مسافة قبل وبعد معامل (and).

٣. يتم كتابة الشرط الثاني مكان (B).

المعامل or

يعني اختيار أحد الشرطين.



١. يتم كتابة الشرط الأول مكان (A).

٢. يتم إضافة مسافة قبل وبعد معامل (or).

٣. يتم كتابة الشرط الثاني مكان (B).

كتابة التعابير الشرطية والمنطقية

في لغة بايثون يمكن كتابة التعابير الشرطية أو المنطقية مستقل أو الجمع بينهم، حيث يمكن كتابة أكثر من تعبير شرطي وربطهم بتعبير منطقي.

- استخدام تعبير شرطي واحد.

```
x > z
z < x
o == x
z != o
```

- استخدام تعبير منطقي واحد.

```
a or b
a and b
```

- استخدام تعبيران شرطيان.

```
x >= z
z =< x
```

- استخدام أكثر من تعبير شرطي وتعبير منطقي.

```
x > o and z < o
x == o or z != o
```

- استخدام أكثر من تعبيران شرطيان وأكثر من تعبير منطقية.

```
x == z and s != z and w > z
x == z and s != z or w > z
x == z or s != z and w > z
x == z or s != z or w > z
```

توليد الأرقام العشوائية

الأرقام العشوائية عبارة عن متتالية من الأعداد تفتقر إلى أي نظام أو ترتيب. وبتعبير آخر هي أعداد عشوائية

في لغة البرمجة بايثون يتم توليد الأرقام العشوائية من خلال بعض الدوال، في هذه الحقيبة سوف نتعلم دالتين هما:

- دالة randint().
- دالة random().

دالة ()randint

هي عبارة عن دالة جاهزة في لغة بايثون تقوم بتوليد أرقام عشوائية صحيحة وذلك ضمن نطاق يحدد من قبل المبرمج أو المستخدم.

```
from random import randint ←1
randint(start, stop)
   ↑    ↑  ↑  ↑   ↑
   2    3  4  5   6
```

١. استدعاء دالة randint من مكتبة random.
٢. كتابة اسم الدالة في البداية.
٣. فتح القوس ثم اغلاقه بعد الانتهاء.
٤. كتابة رقم بداية التوليد.
٥. وضع علامة فاصلة.
٦. كتابة رقم نهائية التوليد.

دالة random()

هي عبارة عن دالة جاهزة في لغة بايثون تقوم بتوليد أرقام عشوائية عشرية عشوائية دون تحديد نطاق التوليد.

```
from random import random ←1
random()
  ↑   ↑
  2   3
```

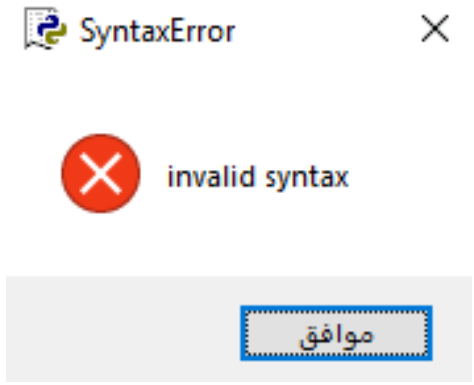
١. استدعاء دالة random من مكتبة random.

٢. كتابة اسم الدالة في البداية.

٣. فتح علامة القوس واغلاقها مباشرة.

الأخطاء في البرمجة

الخطأ مصطلح يستخدم لوصف مشكلة برمجية تظهر بشكل غير متوقع وتتسبب في توقف عمل الشيفرة البرمجية، في لغات البرمجة تصنف الأخطاء إلى ثلاث أنواع رئيسية هي:



أخطاء لغوية (Syntax Errors).

أخطاء منطقية (Logic Errors).

```
1st Num: 3.5
Traceback (most recent call last
):
  File "C:/Users/Tam/AppData/Loc
al/Programs/Python/Python38-32/1
23.py", line 1, in <module>
    a=int(input('1st Num: '))
ValueError: invalid literal for
int() with base 10: '3.5'
```

أخطاء وقت التشغيل (Run-Time Errors).

الاختيار بالجمل الشرطية

الأهداف التفصيلية للوحدة

١. يستخدم الجمل الشرطية if، if-else، if-elif-else في لغة بايثون.

٢. يرسم المخططات الانسيابية للجمل الشرطية.

٣. ينفذ الجمل الشرطية باستخدام دوال المقارنة.

الوقت المتوقع للتدريب على هذه الوحدة: ١٥ ساعة تدريبية.

الوسائل التدريبية المساعدة:

- جهاز حاسب آلي.
- جهاز عرض (Data Show).
- محرر كتابة Python.
- برنامج رسم مخططات انسيابية مثل Microsoft Visio.

الجمل الشرطية في لغة بايثون

عبارة عن أدوات تستخدم لتحديد طريقة عمل البرنامج، حيث يمكن وضع أكثر من شرط في نفس البرنامج أو أكثر من شرط داخل الشرط الأساسي.

وتنقسم الجمل الشرطية في لغة بايثون إلى ثلاث أقسام رئيسية هي:

١. الجملة الشرطية if.
٢. الجملة الشرطية if-else.
٣. الجملة الشرطية if-elif-else.

الجمل الشرطية في لغة بايثون

المخطط الانسيابي:

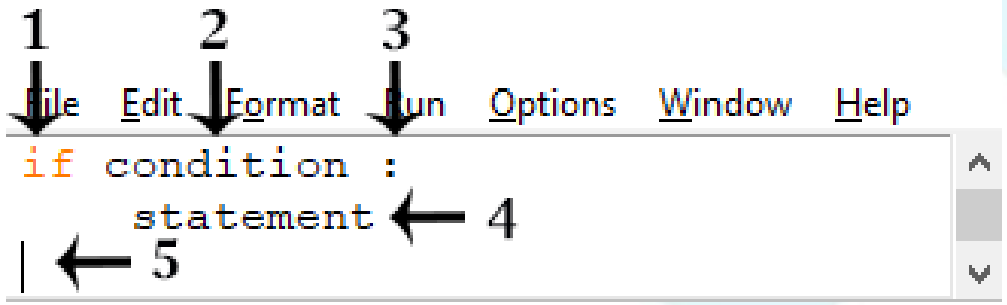
هو عبارة عن شكل توضيحي باستخدام رموز معينة، يتم رسمه من قبل المبرمج ليوضح طريقة عمل البرنامج كاملاً أو جزء منة.

عوامل المقارنة التي تستخدم في الجمل الشرطية:

هي عبارة عن عوامل رياضية تستخدم لصياغة الشروط حيث تقوم بمقارنة المعطيات للتأكد من تحقيق الشروط حسب المطلوب.

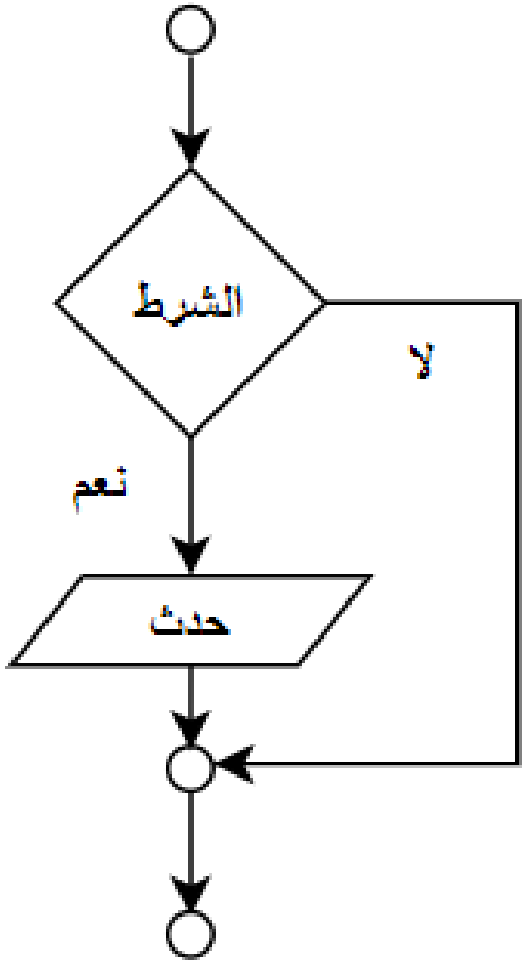
الجملة الشرطية if

بمعنى (إذا) وهي جملة تستخدم عندما يكون الشرط مرتبط بحدث واحد فقط.



١. يتم كتابة if في بداية السطر.
٢. يتم إضافة مسافة بعد if ثم كتابة الشرط.
٣. بعد الانتهاء من الشرط يجب وضع نقطتان رأسيتان ثم الضغط على مفتاح Enter.
٤. هنا يتم كتابة الأحداث الخاصة في الشرط if.
٥. يتم إعادة المؤشر لبداية السطر للخروج من الشرط if.

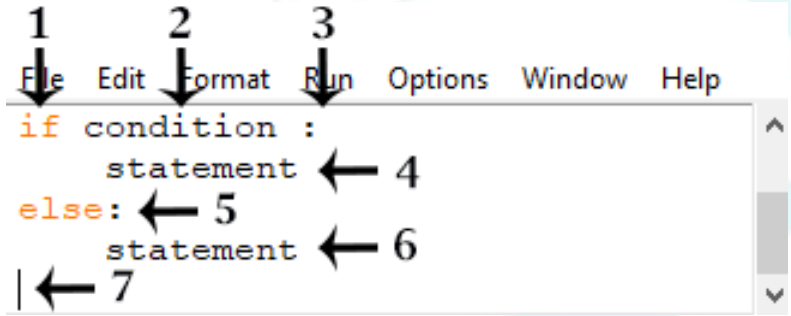
المخطط الانسيابي للشرط if



- الشرط: يتم إضافة الشرط المطلوب التقيد به، وغالباً يكون باستخدام العمليات العلاقية والمنطقية.
- الحدث: إذا تحقق الشرط (نعم) يتم تنفيذ الحدث المحدد.
- إذا لم يتحقق الشرط (لا) يتم تجاوز الحدث.

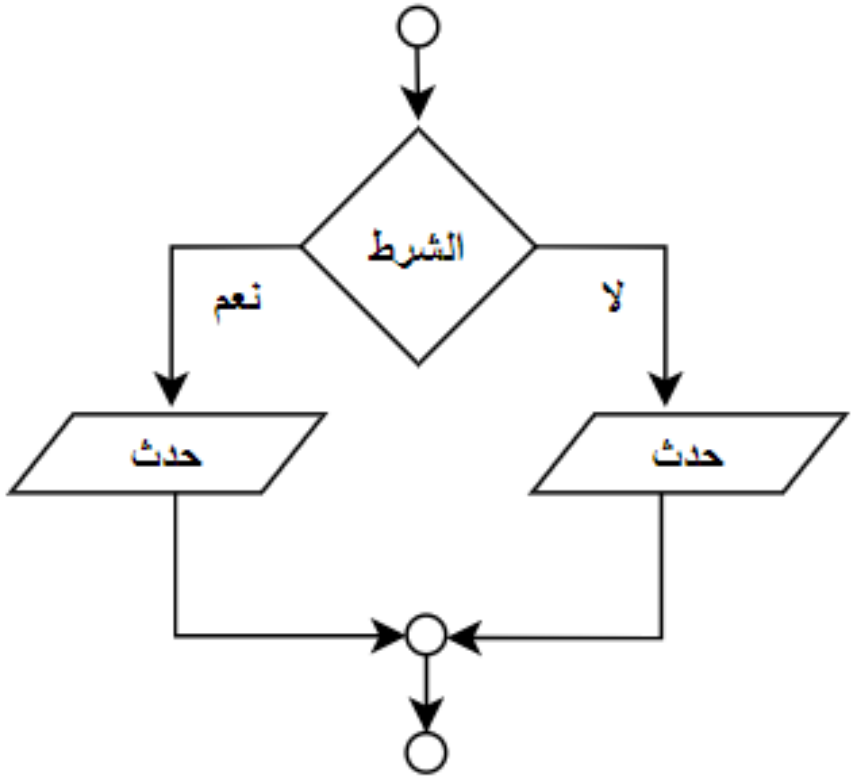
الجملة الشرطية if-else

بمعنى (إما - أو) وهي جملة تستخدم عندما يكون الشرط مرتبط بحدثين مختلفين.



١. يتم كتابة if في بداية السطر.
٢. يتم إضافة مسافة بعد if ثم كتابة الشرط.
٣. بعد الانتهاء من الشرط يجب وضع نقطتان رأسيتان ثم الضغط على مفتاح Enter.
٤. هنا يتم كتابة الأحداث الخاصة في الشرط if.
٥. يتم كتابة else في بداية السطر مع نقطتان رأسيتان بعد أحداث if.
٦. هنا يتم كتابة الأحداث الخاصة في else.
٧. يتم إعادة المؤشر لبداية السطر للخروج من الشرط else.

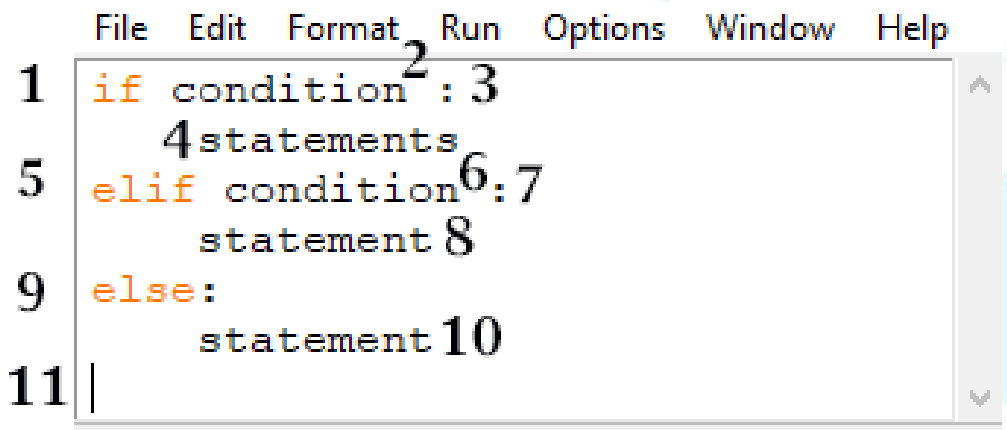
المخطط الانسيابي للشرط if-else



- الشرط: يتم إضافة الشرط المطلوب التقيد به، وغالباً يكون باستخدام العمليات العلاقية والمنطقية.
- الحدث (نعم): إذا تحقق الشرط يتم تنفيذ الحدث المحدد بـ نعم.
- الحدث (لا): إذا لم يحقق الشرط يتم تنفيذ الحدث المحدد بـ لا.

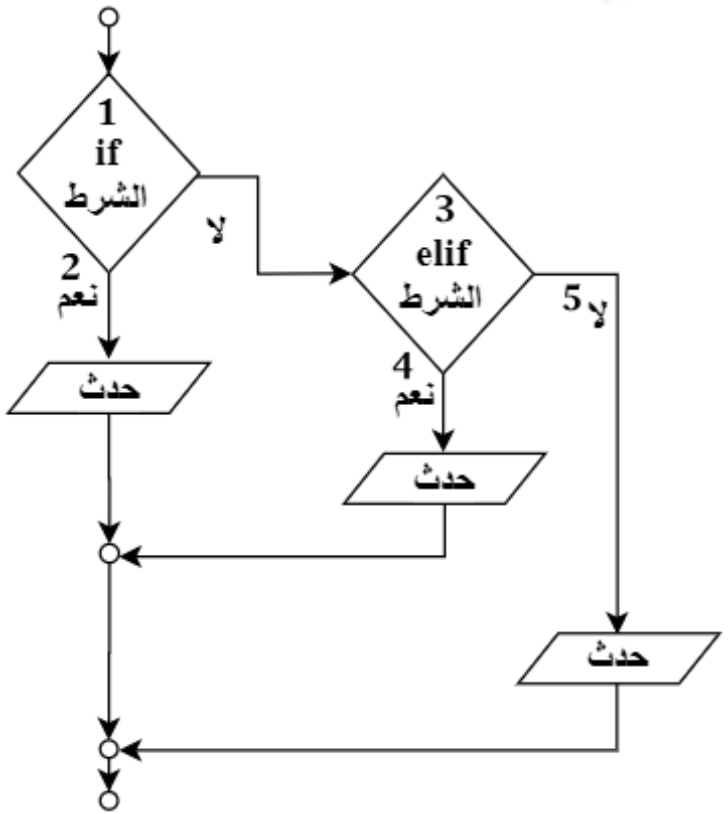
الجملة الشرطية if-elif-else

هي جملة تستخدم عندما يكون هنالك شرطين أو أكثر مرتبطين بثلاث أحداث أو أكثر.



١. يتم كتابة if في بداية السطر.
٢. يتم إضافة مسافة بعد if ثم كتابة الشرط.
٣. بعد الانتهاء من الشرط يجب وضع نقطتان رأسيتان.
٤. هنا يتم كتابة الأحداث الخاصة في شرط if.
٥. يتم كتابة elif في بداية السطر بعد أحداث if.
٦. يتم إضافة مسافة بعد elif ثم كتابة الشرط.
٧. بعد الانتهاء من الشرط يجب وضع نقطتان رأسيتان.
٨. هنا يتم كتابة الأحداث الخاصة في شرط elif.
٩. يتم كتابة else في بداية السطر مع نقطتان رأسيتان بعد أحداث elif.
١٠. هنا يتم كتابة الأحداث الخاصة في else.
١١. يتم إعادة المؤشر لبداية السطر للخروج من الشرط else.

المخطط الانسيابي للشرط if-elif-else



١. الشرط (if): يتم إضافة الشرط المطلوب التقيد به، وغالباً يكون باستخدام العمليات العلاقية والمنطقية.

٢. الحدث (نعم): إذا تحقق الشرط يتم تنفيذ الحدث المحدد بـ (نعم)، إذا لم يتحقق ينتقل إلى الشرط الثاني.

٣. الشرط (elif): يتم إضافة الشرط المطلوب التقيد به، وغالباً يكون باستخدام العمليات العلاقية والمنطقية.

٤. الحدث (نعم): إذا تحقق الشرط يتم تنفيذ الحدث المحدد بـ (نعم).

٥. الحدث: إذا لم يتحقق شرط elifينتقل إلى الحدث الأخير(else).

الأخطاء الشائعة عند كتابة الشروط في لغة بايثون

عدم إضافة مسافة ذلك يعني خروج الحدث من الشرط if.

```
x = 5
if x == 5:
print("تحقق الشرط")
```

عدم تساوي المسافة البادئة للأحداث.

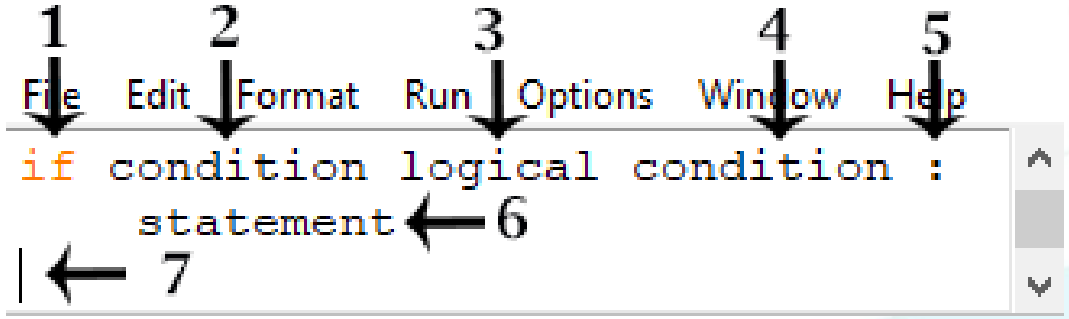
```
x = 5
if x == 5:
    print("تحقق الشرط")
   print("Python")
```

المعاملات المنطقية التي يمكن استخدامها في الشروط

هي عبارة عن معاملات منطقية تستخدم لربط عدة شروط بواسطة التعابير المنطقية، وهما كتالي:

- **المعامل** and: وتعني (و) يعني إضافة، تقوم بإضافة أكثر من تعبير لتحقيق الشرط، عند استخدامها يعني لابد من تحقق جميع التعابير المطلوبة.
- **المعامل** or: تعني (أو) يعني اختيار، تقوم بإضافة أكثر من تعبير لتحقيق الشرط، عند استخدامها يكتفى بتحقق تعبير واحد من التعابير المطلوبة.

صيغة كتابة المعاملات المنطقية في الشروط



١. يتم كتابة if في بداية السطر.

٢. يتم إضافة مسافة بعد if ثم كتابة الشرط.

٣. يتم إضافة مسافة ثم كتابة المعامل المنطقي and أو or.

٤. يتم إضافة مسافة بعد المعامل ثم كتابة الشرط.

٥. بعد الانتهاء من الشروط يجب وضع نقطتان رأسيتان ثم الضغط على مفتاح Enter.

٦. هنا يتم كتابة الأحداث الخاصة في الشرط if.

٧. يتم إعادة المؤشر لبداية السطر للخروج من الشرط.

المراجع

م	المرجع
١	Introduction to Programming Using Python -Y. Daniel Liang
٢	Lean Python Learn Just Enough Python to Build Useful Tools - Paul Gerrard
٣	تعلم البرمجة مع بايثون ٣ – ترجمة: هشام رزق الله
٤	فكر بايثون كيف تفكر كعالم حاسوب - ألن داوني
٥	https://www.w3schools.com/python/default.asp
٦	https://www.programiz.com/python-programming
٧	https://www.youtube.com/playlist?list=PLMYF6NkLrdN98I0nEXOuR_gK8b4w-NJcN
٨	https://www.youtube.com/playlist?list=PLF8OvnCBlEY1j4hxoqXqJk08ASU7D_W87
٩	حقيبة برمجة الحاسب الالي (دعم فني)

تم بحمد الله