ملخص Python

"صخرة الأساس"

BASIC ROCK

ملخص لغة Python

تأليف: احمد حسونة عام 2019

نبذه مختصرة

هذا الكتاب يعد للقارئ بمثابة صخرة قوية ثابته بفضل الله من الأساس في عالم بايثون، مع ملخص لغة بايثون الساحرة مع أهم تفاصيلها من البداية بفضل الله، حيث أن هناك كود وتنفيذ لتطبيق، ثم تتعلم أكثر، ثم تطبق أكثر، ثم تتأسس وتتعلم أكثر وأكثر، ثم تمارس أكثر وتستمر في التطبيق، ثم تصل الى الاحتراف بفضل الله

احمد حسونة

صخرة الأساس ملخص لغة_بايثون

فهرس الكتاب

المحتويات

مقدمة أساسية – هامة جداً

بسم الله الرحمن الرحيم والصلاة والسلام على أشرف المرسلين سيدنا محمد الذي أرسله الله رحمة للعالمين، وأشهد أن لا إله إلا الله وحده لا شريك له، الذي لم يتخذ ولدا ولم يكن له شريك في الملك وخلق كل شيء فقدره تقديرا ،،، ثم أما بعد...

الشيء الوحيد الذي أخبر الله رسوله محمد ﷺ أن يطلب فيه الزيادة هو العلم عندما قال الله سبحانه وتعالي في سورة طه (وَقُلْ رَبِّ زِدْنِي عِلْمًا) ، وأول ما نزل من القرآن الكريم (اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ (1) خَلَقَ الْإِنْسَانَ مِنْ عَلَقٍ (2) اقْرَأْ وَرَبُّكَ الْأَكْرَمُ (3) الَّذِي عَلَّمَ بِالْقَلَمِ (4) عَلَّمَ الْإِنْسَانَ مَا لَمْ يَعْلَمْ) ، ورفع الله من قدر وشأن العلماء حينما قال (يَرْفَعِ اللَّهُ الَّذِينَ آمَنُوا مِنكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ) ، وبين الله عز وجل أن العلماء لا يتساوون في القدر مع من لا يعلم فقال جل في علاه (قُلْ هَلْ يَسْتَوِي الَّذِينَ يَعْلَمُونَ وَالَّذِينَ لَا يَعْلَمُونَ إِنَّمَا يَتَذَكَّرُ أُولُو الْأَلْبَابِ) ، وأيضاً أخبرنا رب العزة سبحانه أن من يخشونه هم العلماء حين قال في كتابه العزيز (إِنَّمَا يَخْشَى اللَّهَ مِنْ عِبَادِهِ الْعُلَمَاءُ) ، وهو الذي علمنا عدد السنين والحساب حين قال (هُوَ الَّذِي جَعَلَ الشَّمْسَ ضِيَاءً وَالْقَمَرَ نُورًا وَقَدَّرَهُ مَنَازِلَ لِتَعْلَمُوا عَدَدَ السِّنِينَ وَالْحِسَابَ مَا خَلَقَ اللَّهُ ذَلِكَ إِلَّا بِالْحَقِّ يُفَصِّلُ الْآيَاتِ لِقَوْمٍ يَعْلَمُونَ) ، وكما نري أن العلم قدرة عالي ولا يطلبه إلا غالي أدرك علاه.

ولقد عزمت بإذن الله تعالي أن أقدم أفضل ما لدي من علوم في مجالات الحاسب الآلي (الكومبيوتر)، لأنه يفيد في جميع مناحي الحياة، فإذا تحدثنا عن شركة تجارية فهي تستخدم الكمبيوتر لتدوين وحساب جميع اعمالها من انتاج وتسويق ورواتب وغيرها، وإذا تحدثنا عن مدرسة أو كلية أو معهد، فجميعها تستخدم الكمبيوتر في أعمالها وتظهر نتائج الامتحانات والجداول، بل وبعض المؤسسات التعليمية الحديثة تمتحن الطالب ببرامج ذكية متطورة، وفي الهيئات الحكومية يتم استخدام الكمبيوتر أيضاً، وأصبح الحاسب يفعل كل شيء في حياتنا فلقد صعد الإنسان في الآفاق والفضاء بالحاسب، ونشاهد به الفيديو ونشغل به الصوت ونطبع به الوثائق والمستندات ونصور به الصور والأوراق وندير به كافة المجالات، ولا نقتصر على هذا فحسب، بل ولو قلت لك أن الحاسب قد يغير في حياة أسرة بأكملها وينجيها بفضل الله ، سوف تتعجب وتقول كيف ؟! – والجواب هو كم من رجل كان على حافة الانهيار المادي والمعنوي بسبب فقده لوظيفته، ففتح الله علية بتعلم الحاسب ورزقه الله بعمل جيد ورزق وفير، وكم من شاب كان على حافة الانحراف الأخلاقي بسبب الأخلاق السيئة المتداولة بين الشباب إلا من رحم ربي، فمن الله علية بأن بدء في تعلم مجالات الحاسب فأحبها وبدء يبني مستقبلة الحقيقي.

المحتوي الذي سوف نتعلمه يتكون من مواضيع تتبع بعضها بعض، ومنظمة بطريقة تسلسلية بحيث تأخذ بيدك خطوة بخطوة من البداية إلى الاحتراف بفضل الله، وسوف أتبع نظام في شرح هذا المحتوي مثل صعود الطائرة، ففي البداية يتم التنبيه على الركاب ببعض الإرشادات بالتفصيل الدقيق، ثم

في بداية الإقلاع يكون الأمر صعباً ثم نطير إلى أعلي ونحلق في الفضاء، وسوف نبدأ بإذن الله بالتفصيل ثم نعلوا.. ونعلوا ، ثم نحلق ونحترف بإذن الله، فإنني أنصح القارئ أن يهتم بجميع الموضوعات مرتبة، فعسي كلمة أن تغير بفضل الله، والمحتوي تم تأليفه بكامل الحب والإخلاص لجميع القراء، ولن يأخذ نسخ ولصق، بل وحتي المحتويات التاريخية الموجودة داخل المحتوي تم صياغتها بأساليب مختلفة من أكثر من مكان معلوماتي، والمؤلف وضع كل خبرته بكل الحب لجميع أحبائه القارئين لهذا المحتوي، والكمال لله وحدة، ونحن لا نعلم إلا قليلاً كما قال الله في القرآن (وَيَسْأَلُونَكَ عَنِ الرُّوحِ ۖ قُلِ الرُّوحُ مِنْ أَمْرِ رَبِّي وَمَا أُوتِيتُم مِّنَ الْعِلْمِ إِلَّا قَلِيلًا).

وفي الختام أسأل الله تبارك وتعالي بأسمائه الحسني وصفاته العلا أن يفيد جميع القراء بهذا المحتوي، ويغير في حياتهم إلى الأفضل إن شاء الله.

كما أحب أن اشجع الجميع على العطاء في العلم، فإذا كنت تعلم أي معلومة لا تبخل على غيرك بها أبداً، وتأكد أن الله سوف يكرمك بذلك ويمن عليك بمعلومات أفضل بكثير من الذي لديك، ولقد أمرنا رسول الله كامل الأخلاق أن نحب الخير لإخواننا كما نحبه لأنفسنا حين قال (حب لأخيك ما تحب لنفسك)، وهذا الموضوع هام للغاية لأنه ينهض بالمجتمع ككل، فإذا تعلمت شيء وعلمته لأخيك أو زميلك فسوف يأخذ فكرتك ثم يبتكر عليها ابتكارا مختلفاً من ابتكارك، لأن الله خلق البشر بعقول ومميزات مختلفة، ثم يعلم زميلك معلومتك ومعلومته لزميل آخر، فيبتكر ابتكاراً ثالثاً مختلفاً عن الأول والثاني، وهكذا، ولو حدث هذا على مستوي المجتمع

نهض المجتمع وكثرت ابتكاراته واختراعاته، ولا أقصد مجال الحاسب الآلي فحسب، لا، بل أقصد جميع المجالات في جميع مناحي الحياة، وإذا قابلت شخص طلبت منه معلومة وبخل عليك، ثم بعد ذلك طلب منك معلومة لا تفعل مثلة، بل اعطية المعلومة على الفور ولمح له أن هذا هو الصحيح، وسوف يؤثر هذا علية ويغير حالة إلى العطاء، وهذا حدث معي شخصياً أكثر من مرة، وأثبتت لي التجربة أن هذا هو الصحيح.

كما أحب أن انوه أنه غير مسموح لأي فرد أو شركة أو هيئة أو منظمة أو مؤسسة أو قطاع من أي نوع أن يتربح أو يتاجر بهذا المحتوي بأي نوع أي كان، أو يعيد كتابته، أو ينسخه بغير حق ليغير الاسم، أو أي عملية تحويل كتاب pdf الى أي نوع أو التعديل عليه، ولن اسامح أي شخص تسبب في ذلك، ولن أتركه يوم القيامة، وعند الله تجتمع الخصوم.

وسوف تجد شرح فيديو لهذا المحتوي بالتفصيل مجاناً للمشاهدة فقط على أكاديمية حسونة في الموقع أو اليوتيوب على الرابطين التاليين:

www.hassouna-academy.com

www.youtube.com/user/hassounaacademy

والسلام عليكم ورحمة الله وبركاته.

المؤلف: احمد حسونة

هام جداً Very Important قبل الكتاب

أمانة - ثمن الكتاب، قراءة هذه الصفحة جيداً وصفحة نشكرك على قراءتك وصفحة مقدمة أساسية وتلبيه ما فيهم 😊 فضلاً منكم وليس أمراً.

موقع حسونة أكاديمي ← https://www.hassouna-academy.com

قـناة حسونة أكاديمي ← https://www.youtube.com/user/HassounaAcademy

بفضل الله، القناة عليها شرح كمبيوتر، ووينـدوز، ولينكس، وماك، وورد، واكسل، وأكسس، وباور بوينت، ولغات برمجة كثيرة، وويب بتقنيات كثيرة، وأمثلة كثيرة، وتطبيقات كثيرة، وموبايل، وأندرويد، وقواعد بيانات، ونظم، وتصميم، وتحليل بيانات، وتحليل نظم، ومناهج دراسية، ووسائط متعددة، ودردشة برمجة، وبرنامج مبيعات، وبرنامج مطاعم، ومعلومات متعددة، وقصص نجاح، وبرامج جاهزة، وفوتوشوب، وشروحات كثيرة بفضل الله.

رجاء! لا تنسي أن تشترك بالقناة ↗ وفعل زر الجرس في القناة 🔔 وسجل حساب على موقع حسونة أكاديمي 👤 الآن.

https://www.hassouna-academy.com/

شرح مرئي للكتاب

يوجد شرح مرئي على قناة أكاديمية حسونة لهذا الكتاب من الرابط البسيط التالي:

https://youtu.be/TOOGkMTZHgM

وهو مقطع فيديو حماسي لتعليم البايثون. لا يفوتك!

قواعد

- اعلم أن لغة البايثون تفرق بين الحروف الكبيرة والصغيرة مثل A وa أو B وb ويجب مراعاة ذلك
- لغة بايثون تجعل نوع المتغير بناء على قيمته التي اعطيتها إياه
- في لغة البايثون يكون الكود متفرع من جملة تكرار او if او غيرة باستخدام الـ tab، ومعني ذلك ان لغة البايثون تحاسب على المسافات
- لابد من التركيز جيداً في الكود وفي تنفيذ الكود لأن بهم بعض التفسير
- اعلم أن هذا الكتاب بمثابة مراجعة سريعة على لغة البايثون بدون تعليقات كثيرة على الأكواد، حيث أن كود لغة البايثون من أسهل الأكواد التي يمكنك كتابتها وفهم الكود فقط من قراءة عنوان الدرس في الكتاب
- إذا كنت مبتدئ لا تقرأ هذا الكتاب الآن، ولكن شاهد كل شرح البايثون الموجود على قناة حسونة أكاديمي أولاً، لتعرف اللغة وتتعلم جيداً قبل مراجعتها من هذا الكتاب

الطباعة – الدالة print

لاحظ! أنه فيما يلي تم طباعة قيم مختلفة

الكود

```
print(33)
print(77.99)
print(True)
print(False)
print("Hello")
print('Hello')
print("Hello 'Amr'")
print('Hello "Amr"')
print(None)
```

التنفيذ

```
33
77.99
True
False
Hello
Hello
Hello 'Amr'
Hello "Amr"
None
>>>
```

لاحظ! أنه فيما يلي تم طباعة متعددة بجملة print واحدة

الكود

```
print( "Hi" , 'Ok' , 7 , True , 5.5 )
```

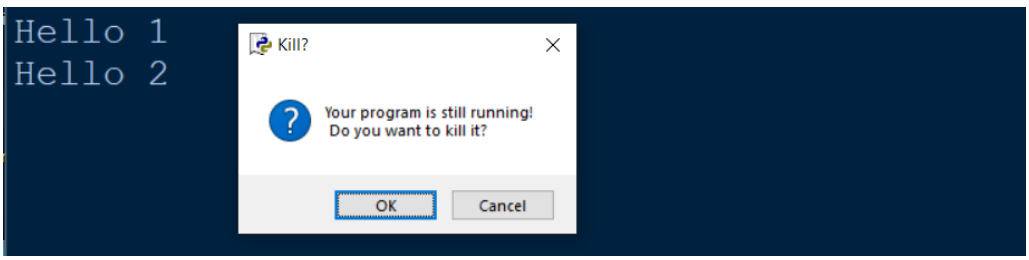
التنفيذ

الخروج من البرنامج – الدالة exit

لاحظ! أنه فيما يلي تم الخروج من البرنامج وعدم تكملة ما بعد الخروج الكود

```
print( 'Hello 1' )
print( 'Hello 2' )
exit()
print( 'Hello 3' )
```

التنفيذ



تنفيذ أكثر من أمر بسطر واحد

لاحظ! أنه فيما يلي تم تنفيذ اكثر من جملة برمجية باستخدام ; للعزل بين كل جملة وأخري

الكود

```
print('Ok 1'); print('Ok 2'); print('Ok 3')
```

التنفيذ

```
Ok 1
Ok 2
Ok 3
>>>
```

دمج النصوص بالرمز +

لاحظ! أنه فيما يلي تم دمج النصوص ولصقها بالرمز + بين النصوص

الكود

```
print( "Hello" + " " + "Ahmed" )
print( "Hello" + ' ' + 'Adel' )
print( 'Hello' + ' ' + 'Amr' )
print( 'My' + " " + 'name is ' + '"' + "Ali" + '"' )
```

التنفيذ

```
Hello Ahmed
Hello Adel
Hello Amr
My name is "Ali"
>>>
```

المعاملات الحسابية

لاحظ! أنه فيما يلي تم استخدام الجمع والطرح والضرب والقسمة وباقي القسمة

الكود

```
print(7+3)
print(7-3)
print(7*3)
print(7/3)
print(7%3)
```

التنفيذ

```
10
4
21
2.3333333333333335
1
>>>
```

لاحظ! أنه فيما يلي يمكن إزالة الكسر مع القسمة باستخدام // ويمكن حساب الأس الرياضي باستخدام **

الكود

```
print( 7//3 )
print( 5**3 )
```

التنفيذ

```
2
125
>>>
```

المتغيرات

لاحظ! أنه فيما يلي تم تعريف متغيرين ثم جمعهم في متغير ثالث

الكود

```
num1 = 7
num2 = 3
result = num1 + num2
print(result)
```

التنفيذ

```
10
>>>
```

لاحظ! أنه فيما يلي تم متغير ثم دمجه مع الكلمة Hello مسافة

الكود

```
name = 'Ahmed'
say_hello = 'Hello ' + name
print(say_hello)
```

التنفيذ

```
Hello Ahmed
>>>
```

لاحظ! أنه فيما يلي تم تعريف متغيرات على التوالي، ثم اعطاؤهم القيم أيضاً على التوالي

الكود

```
name1, name2, name3 = 'Ahmed', 'Adel', 'Amr'
print( name1 + ' ' + name2 + ' ' + name3 )
```

التنفيذ

```
Ahmed Adel Amr
>>>
```

الكود

```
num, name, salary, is_active = 1, 'Ali', 4500.66, True
print( num )
print( name )
print( salary )
print( is_active )
```

التنفيذ

```
1
Ali
4500.66
True
>>>
```

لاحظ! أنه فيما يلي المتغير لا يحمل بداخلة الا قيمة واحدة فقط

الكود

```
name = 'Adel'
print(name)
name = 'Ahmed'
name = 'Amr'
print(name)
```

التنفيذ

```
Adel
Amr
>>>
```

طباعة نوع البيانات – الدالة type

لاحظ! أنه فيما يلي تم اظهار نوع بيانات كل متغير باستخدام type

الكود

```
var1 = 733
var2 = 99.55
var3 = True
var4 = 'Hello'
var5 = "Hi"
print( type(var1) )
print( type(var2) )
print( type(var3) )
print( type(var4) )
print( type(var5) )
```

التنفيذ

```
<class 'int'>
<class 'float'>
<class 'bool'>
<class 'str'>
<class 'str'>
>>>
```

متغير نصي متعدد الأسطر

لاحظ! أنه فيما يلي تم تعريف متغير نصي يحمل اسطر نصية تطبع كما هي

الكود

```
my_str =  """
Welcome to Hassouna Academy
   Windows
   Programming
   Development
Create Account On www.hassouna-academy.com
"""
print(my_str)
```

التنفيذ

```
Welcome to Hassouna Academy
   Windows
   Programming
   Development
Create Account On www.hassouna-academy.com

>>>
```

الكود

```
my_str =  '''
Welcome to Hassouna Academy
   I love Python
Now Create Account On www.hassouna-academy.com
'''
print(my_str)
```

التنفيذ

```
Welcome to Hassouna Academy
   I love Python
Now Create Account On www.hassouna-academy.com

>>>
```

نص متكرر بالرمز *

لاحظ! أنه فيما يلي تم استخدام معامل الضرب * لتكرار النص بعدد معين سواء نص ثابت او نص محمول في متغير

الكود

```
print( 'A ' * 5 )
name = ' AMR '
print( name * 3 )
```

التنفيذ

```
A A A A A
 AMR  AMR  AMR
>>>
```

السلاسل النصية

لاحظ! أنه فيما يلي تم استخدام السلاسل النصية أو ما تعرف بسلاسل الهروب Escape Sequence وتم توضيح ما يفعله كل واحد فيهم عند الطباعة مع كل كود

الكود

```
str1 = '\n'
str2 = '\N{copyright sign}'
str3 = '\N{registered sign}'
str4 = '\N{up down arrow}'
str5 = '\N{left right arrow}'
str6 = '\x41'
str7 = '\u0042'
str8 = '\U00000043'

print( 'This For New Line' , str1 , 'OK' )
print( str2, str3, str4,str5 )
print( str6, str7, str8 )
```

التنفيذ

```
This For New Line
 OK
© ® ↕ ↔
A B C
>>>
```

لاحظ! أنه فيما يلي تم استخدام الـ DOS عند التنفيذ لسماع الزمارة Beep والتي هي \a في الكود

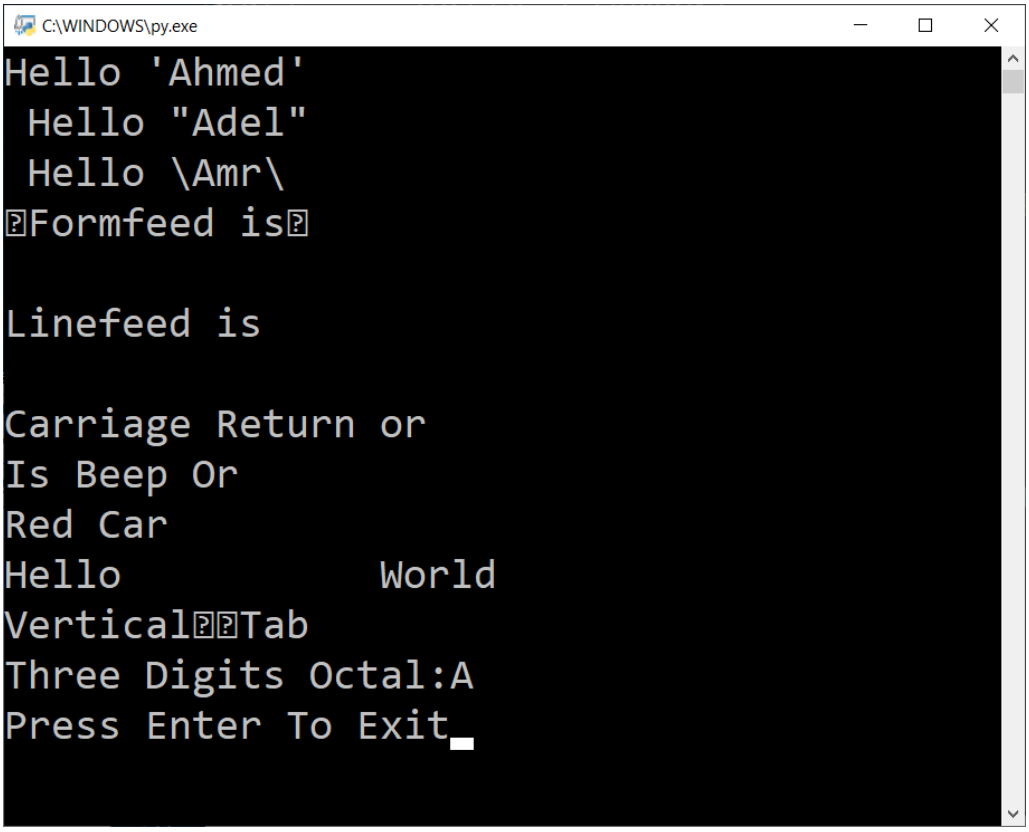
الكود

```
str1  = 'Hello \'Ahmed\''
str2  = "Hello \"Adel\""
str3  = 'Hello \\Amr\\'
str4  = '\fFormfeed is\N{FF}'
str5  = '\nLinefeed is\N{LF}'
str6  = 'Welcome to Egypt\rCarriage Return or \N{CR}'
str7  = '\aIs Beep Or \N{BEL}'
str8  = 'Rad Cat\br\b\b\b\b\N{BS}\be'
str9 = 'Hello\t\N{TAB}World'
str10 = 'Vertical\v\N{VT}Tab'
str11 = 'Three Digits Octal:\101'

print( str1, '\N{LF}', str2, '\n', str3 )
print( str4 )
print( str5 )
print( str6 )
print( str7 )
print( str8 )
print( str9 )
print( str10 )
print( str11 )

input('Press Enter To Exit')
```

التنفيذ



المتغير الصف Tuple الغير قابل لتغيير القيم

لاحظ! أنه فيما يلي تم استخدام متغير tuple به قيم والنداء عليها بالرقم index وهو يبدأ من الصفر، ولاحظ ان القيم بداخلة غير قابلة للتغيير

الكود

```
p1 = (1,'Ahmed',3900.50)
p2 = (2,'Adel',4600.60)
```

```
p3 = (3,'Amr',4500.55)
print(p1); print(p2); print(p3)
print( p1[0], p1[1], p1[2] )
print( type(p1) )
print( type(p1[0]), type(p1[1]), type(p1[2]) )
```

التنفيذ

```
(1, 'Ahmed', 3900.5)
(2, 'Adel', 4600.6)
(3, 'Amr', 4500.55)
1 Ahmed 3900.5
<class 'tuple'>
<class 'int'> <class 'str'> <class 'float'>
>>>
```

المتغير القائمة List القابل لتغيير القيم

لاحظ! أنه فيما يلي تم استخدام متغير list قائمة بها قيم والنداء عليها بالرقم index وهو يبدأ من الصفر، ويمكن تغيير القيم به

الكود

```
numbers = [11,22,33]
names   = ['Amr','Ali','Ezz']
print( numbers ); print( names )
print( numbers[0] )
print( numbers[1] )
print( numbers[2] )
names[0] = 'Akl'
print(names)
```

التنفيذ

```
[11, 22, 33]
['Amr', 'Ali', 'Ezz']
11
22
33
['Akl', 'Ali', 'Ezz']
>>>
```

الكود

```
person = [1,'Ahmed',3500.55,True]
print(person)
```

التنفيذ

```
[1, 'Ahmed', 3500.55, True]
>>>
```

لاحظ! أنه فيما يلي تم استخدام append لوضع قيمة واستخدام extend لوضع قيم دفعة واحدة واستخدام remove لحذف قيمة واستخدام insert لوضع قيمة في مكان معين واستخدام copy لنسخ القيم واستخدام clear لحذف القيم

الكود

```
names   = ['Amr','Ali','Ezz']
print(names)
names.append('Omar')
print(names)
names.extend( ['Adel','Akl'] )
print(names)
names.remove(names[1])
print(names)
names.insert(1,'Ali')
```

```
print(names)
names2 = names.copy()
names.clear()
print(names)
print(names2)
```

التنفيذ

```
['Amr', 'Ali', 'Ezz']
['Amr', 'Ali', 'Ezz', 'Omar']
['Amr', 'Ali', 'Ezz', 'Omar', 'Adel', 'Akl']
['Amr', 'Ezz', 'Omar', 'Adel', 'Akl']
['Amr', 'Ali', 'Ezz', 'Omar', 'Adel', 'Akl']
[]
['Amr', 'Ali', 'Ezz', 'Omar', 'Adel', 'Akl']
>>>
```

متغير الصف المتداخل

لاحظ! أنه فيما يلي تم عمل متغير يحمل بداخلة tuples تتفرع منه الكود

```
family1 = ('Ahmed','Adel','Amr')
family2 = ('Sarah','Hajer','Rehab')

family3 = ('Tawfeek','Ezzat','Foaad')
family4 = ('Hasan','Shokry','Ali','Akl')

home1   = ( family1 , family2  )
home2   = ( family3 , family4 )

print( home1[0][1] )
print( home1[1][0] )
print( home2[0][2] )
```

```
print( home2[1][3] )
print( home1[0] )
print( home2[1] )
print( home1 )
print( home2 )
```

التنفيذ

```
Adel
Sarah
Foaad
Akl
('Ahmed', 'Adel', 'Amr')
('Hasan', 'Shokry', 'Ali', 'Akl')
(('Ahmed', 'Adel', 'Amr'), ('Sarah', 'Hajer', 'Rehab'))
(('Tawfeek', 'Ezzat', 'Foaad'), ('Hasan', 'Shokry', 'Ali', 'Akl'))
>>>
```

متغير القوائم المتداخلة

لاحظ! أنه فيما يلي تم عمل متغير يحمل بداخلة lists تتفرع منه

الكود

```
family1 = ['Ahmed','Adel','Amr']
family2 = ['Ehab','Mahmoud','Ezz']
family3 = ['Sarah','Hajer','Rehab']

family4 = ['Tawfeek','Ezzat','Foaad']
family5 = ['Abdelrahman','Abdelkareem']
family6 = ['Hasan','Shokry','Ali','Akl']

home1   = [ family1 , family2 , family3 ]
home2   = [ family4 , family5 , family6 ]

print( home1[0][1] )
print( home1[1][0] )
print( home2[0][2] )
```

```
print( home2[2][3] )
print( home1[0] )
print( home2[1] )
del(home1[1])
del(home2[1])
print( home1 )
print( home2 )
```

التنفيذ

```
Adel
Ehab
Foaad
Akl
['Ahmed', 'Adel', 'Amr']
['Abdelrahman', 'Abdelkareem']
[['Ahmed', 'Adel', 'Amr'], ['Sarah', 'Hajer', 'Rehab']]
[['Tawfeek', 'Ezzat', 'Foaad'], ['Hasan', 'Shokry', 'Ali', 'Akl']]
>>>
```

المتغير القاموس

لاحظ! أنه فيما يلي تم استخدام متغيرات من نوع خاص وهو dictionary حيث ان كل قيمة تحمل مفتاح key وقيمة value حيث يتم النداء على القيم باستخدام المفاتيح

الكود

```
person1 = {'name':'Amr', 'salary':5000, 'active':True}
person2 = {'name':'Ali', 'salary':4000, 'active':True}
person3 = {'name':'Ezz', 'salary':3000, 'active':True}
print( person1['name'], person1['salary'], person1['active'] )
print( person2['name'], person2['salary'], person2['active'] )
print( person3['name'], person3['salary'], person3['active'] )
```

التنفيذ

```
Amr 5000 True
Ali 4000 True
Ezz 3000 True
>>>
```

لاحظ! أنه فيما يلي تم اظهار المفاتيح والقيم

الكود

```
person = {'name':'Amr', 'salary':5000, 'active':True}
print( person.keys() )
print( person.values() )
```

التنفيذ

```
dict_keys(['name', 'salary', 'active'])
dict_values(['Amr', 5000, True])
>>>
```

لاحظ! أنه فيما يلي يمكن جعل المفاتيح والقيم عبارة عن list

الكود

```
person = {'name':'Amr', 'salary':5000, 'active':True}
print( list( person.keys() ) )
print( list( person.values() ) )
```

التنفيذ

```
['name', 'salary', 'active']
['Amr', 5000, True]
>>>
```

لاحظ! أنه فيما يلي تم استخدام items لإظهار كل مفتاح وبجانبه القيمة التابعة له، ويمكن جعلها list

الكود

```
person = { 'name':'Adel', 'city':'Giza', 'salary':3000 }
print( person.items() )
```

التنفيذ

```
dict_items([('name', 'Adel'), ('city', 'Giza'), ('salary', 3000)])
>>>
```

الكود

```
person = { 'name':'Adel', 'city':'Giza', 'salary':3000 }
print( list(person.items()) )
```

التنفيذ

```
[('name', 'Adel'), ('city', 'Giza'), ('salary', 3000)]
>>>
```

متغير مركب باستخدام set

لاحظ! أنه فيما يلي تم استخدام set فارغة بدون قيم

الكود

```
names = set()
print( names )
```

التنفيذ

```
set()
>>>
```

لاحظ! أنه فيما يلي تم وضع قائمة بداخل الـ set

الكود

```
L1 = ['Ahmed','Adel','Omar']
names = set(L1)
print( names )
```

التنفيذ

```
{'Omar', 'Ahmed', 'Adel'}
>>>
```

لاحظ! أنه فيما يلي تم وضع قائمة جديدة بـ update

الكود

```
L1 = ['Ahmed','Adel','Omar']
L2 = ['Amr','Ali','Ezz']
names = set(L1)
names.update(L2)
print( names )
```

التنفيذ

```
{'Amr', 'Adel', 'Ezz', 'Omar', 'Ahmed', 'Ali'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام add لإضافة قيمة

الكود

```
L1 = ['Amr','Ali']
names = set(L1)
names.add('Ezz')
names.add('Yaser')
print( names )
```

التنفيذ

```
{'Amr', 'Ezz', 'Ali', 'Yaser'}
>>>
```

لاحظ! أنه فيما يلي تم تكرار القيم، ثم لا يقبل التكرار

الكود

```
names = set()
names.add('Ahmed')
names.add('Ahmed')
names.add('Ahmed')
print( names )
```

التنفيذ

```
{'Ahmed'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام remove لحذف قيمة، ولو القيمة المراد حذفها غير موجودة سيصدر البرنامج خطأ

الكود

```
L1 = ['Amr','Ali']
names = set(L1)
names.add('Ezz')
names.add('Yaser')
names.remove('Ali')
print( names )
```

التنفيذ

```
{'Amr', 'Yaser', 'Ezz'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام discard لحذف قيمة، ولو القيمة المراد حذفها غير موجودة لن يصدر البرنامج خطأ

الكود

```
L1 = ['Amr','Ali']
names = set(L1)
names.add('Ezz')
names.add('Yaser')
names.discard('Ali')
print( names )
```

التنفيذ

```
{'Ezz', 'Amr', 'Yaser'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام pop لحذف آخر قيمة مع عدم مراعاة الترتيب، حيث ان الترتيب غير ثابت

الكود

```
L1 = ['Amr','Ali']
names = set(L1)
names.add('Ezz')
names.add('Yaser')
names.pop() #Remove the last item without sort
print( names )
```

التنفيذ

```
{'Amr', 'Ali', 'Ezz'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام clear لحذف كل القيم

الكود

```
L1 = ['Amr','Ali']
names = set(L1)
names.add('Ezz')
names.add('Yaser')
names.clear() #Remove all items
print( names )
```

التنفيذ

```
set()
>>>
```

لاحظ! أنه فيما يلي تم استخدام union لعمل اتحاد

الكود

```
names1 = set(['Adel','Omar','Atef'])
names2 = set(['Amr','Ali','Ezz'])

all_names = names1.union(names2)
print( all_names )
```

التنفيذ

```
{'Ezz', 'Amr', 'Omar', 'Adel', 'Ali', 'Atef'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام union لعمل اتحاد، ولاحظ أن التكرار غير مقبول

الكود

```
names1 = set(['Adel','Omar','Atef','Amr'])
```

```
names2 = set(['Amr','Ali','Omar','Ezz','Adel'])

all_names = names1.union(names2)
print( all_names )
```

التنفيذ

```
{'Ezz', 'Adel', 'Amr', 'Atef', 'Ali', 'Omar'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام intersection لتجميع المشترك بينهم

الكود

```
names1 = set(['Adel','Omar','Atef','Amr'])
names2 = set(['Amr','Ali','Omar','Ezz','Adel'])

all_names = names1.intersection(names2)
print( all_names )
```

التنفيذ

```
{'Amr', 'Adel', 'Omar'}
>>>
```

لاحظ! أنه فيما يلي تم استخدام difference لتجميع المختلف

الكود

```
names1 = set(['Adel','Omar','Atef','Amr'])
names2 = set(['Amr','Ali','Omar','Ezz','Adel'])

all_names1 = names1.difference(names2)
all_names2 = names2.difference(names1)
print( all_names1 )
```

```
print( all_names2 )
```

التنفيذ

```
{'Atef'}
{'Ezz', 'Ali'}
>>>
```

البحث في الأشياء باستخدام in

لاحظ! أنه فيما يلي تم البحث عن الحرف داخل النص

الكود

```
name = 'Ahmed Hassouna'
print( 'H' in name )
```

التنفيذ

```
True
>>>
```

لاحظ! أنه فيما يلي تم البحث عن العنصر داخل القائمة

الكود

```
names = ['Amr','Ali','Ezz']
print( 'Amr' in names )
```

التنفيذ

```
True
>>>
```

لاحظ! أنه فيما يلي تم البحث عن العنصر داخل المجموعة

الكود

```
names = {'Amr','Ali','Ezz'}
print( 'Amr' in names )
```

التنفيذ

```
True
>>>
```

لاحظ! أنه فيما يلي تم البحث عن العنصر داخل الصف

الكود

```
names = ('Amr','Ali','Ezz')
print( 'Amr' in names )
```

التنفيذ

```
True
>>>
```

لاحظ! أنه فيما يلي تم البحث عن العنصر داخل الـ set

الكود

```
names = set(['Amr','Ali','Ezz'])
print( 'Amr' in names )
```

التنفيذ

```
True
>>>
```

لاحظ! أنه فيما يلي تم البحث عن رقم بالنص داخل النص

الكود

```
my_dept = 'Department 3'
print( '3' in my_dept )
```

التنفيذ

```
True
>>>
```

حذف الأشياء – الدالة del

لاحظ! أنه فيما يلي تم حذف متغير، وبنفس الطريقة يمكن حذف أي شيء

الكود

```
name = 'Amr'
print( name )
del name
```

التنفيذ

```
Amr
>>>
```

الكود

```
names   = ['Amr','Ali','Ezz']
print(names)
del(names[2])
print(names)
```

التنفيذ

```
['Amr', 'Ali', 'Ezz']
['Amr', 'Ali']
>>>
```

عدد الأشياء – الدالة len

لاحظ! أنه فيما يلي تمت عملية عد لحروف النص وتم معرفة عدد العناصر

الكود

```
name1  = 'Ahmed'
name2  = 'Adel'
name3  = 'Amr'

names1 = ('Ahmed','Adel','Amr')
names2 = ['Sarah','Hajer','Rehab','Heba']

length_name1 = len(name1)
length_name2 = len(name2)
length_name3 = len(name3)

length_names1 = len(names1)
length_names2 = len(names2)

print( 'length_name1  :', length_name1, 'Characters' )
print( 'length_name2  :', length_name2, 'Characters' )
print( 'length_name3  :', length_name3, 'Characters' )

print( 'length_names1 :', length_names1, 'Items' )
print( 'length_names1 :', length_names2, 'Items' )
```

التنفيذ

```
length_name1  : 5 Characters
length_name2  : 4 Characters
length_name3  : 3 Characters
length_names1 : 3 Items
length_names1 : 4 Items
>>>
```

التعليقات التي لا تنفذ مع الكود

لاحظ! أنه فيما يلي تم عمل تعليقات كملاحظات ولا تنفذ مع الكود

الكود

```
#This Words Not Run, But Comment
name = 'Ahmed' #This is my name
print(name) #Print name in here
#The Hash Symbol For One Line Comment
#Can be multiline using # in each first line
'''
And Can be multiline using triple single quote
In First Paragraph
And In Last Paragraph
'''
"""
And Can be multiline using triple double quote
In First Paragraph
And In Last Paragraph
"""
```

التنفيذ

```
Ahmed
>>>
```

عوامل التخصيص

لاحظ! أنه فيما يلي تم استخدام معامل = ومعاملات تخصيص الإضافة مثل المعامل =+ أي اجعل المتغير كما هو وزد عليه، او المعامل =/ أي اجعل المتغير كما هو واقسم عليه، وهكذا

الكود

```
num = 5; print(num)
num += 6; print(num)
num -= 4; print(num)
num *= 2; print(num)
num **= 2; print(num)
num /= 3; print(num)
num //= 3; print(num)
num %= 12; print(num)
```

التنفيذ

```
5
11
7
14
196
65.33333333333333
21.0
9.0
>>>
```

لاحظ! أنه فيما يلي تم استخدام معامل التخصيص =+ مع النص، بحيث يحتفظ بالنص الموجود ثم يزيد عليه

الكود

```
my_str =  ""
my_str += "Hello"
my_str += " "
my_str += "Ahmed"
print(my_str)
```

التنفيذ

```
Hello Ahmed
>>>
```

لاحظ! أنه فيما يلي تم تخصيص قيمة واحدة لعدة متغيرات

الكود

```
num1 = num2 = num3 = 7
print(num1)
print(num2)
print(num3)
```

التنفيذ

```
7
7
7
>>>
```

التحويل من رقم لنص – الدالة str

لاحظ! أنه فيما يلي تم تحويل الرقم الى نص باستخدام الدالة str حتى نستطيع دمجه مع نص آخر

الكود

```
num = 99
print( 'My number is: ' + str(num) )
```

التنفيذ

```
My number is: 99
>>>
```

التحويل من نص لرقم – الدوال int, float

لاحظ! أنه فيما يلي تم عمل تحويل عدد صحيح int وعدد كسور float

الكود

```
num_1 = '77'
num_2 = '77.33'
num1  = int(num_1)
num2  = float(num_2)
print( num1 , type(num1) )
print( num2 , type(num2) )
```

التنفيذ

```
77 <class 'int'>
77.33 <class 'float'>
>>>
```

التحويل من رقم لقيمة منطقية – الدالة bool

لاحظ! أنه فيما يلي تم تحويل القيم الصحيحة الى قيم منطقية Boolean حيث أن الصفر يمثل False وأي قيمة رقمية أخري تمثل True حتى ولو كانت بالسالب

الكود

```
num1  = 1
num2  = 0
bool1 = bool(num1)
bool2 = bool(num2)
print( bool1 , type(bool1) )
print( bool2 , type(bool2) )
```

التنفيذ

```
True <class 'bool'>
False <class 'bool'>
>>>
```

تحويل الحرف الى ASCII – الدالة ord

لاحظ! أنه فيما يلي تم الوصول الى ascii الحرف بالدالة ord

الكود

```
c = 'A'
i = ord(c)
print( i )
```

التنفيذ

```
65
>>>
```

تحويل الـ ASCII لحرف – الدالة chr

لاحظ! أنه فيما يلي تم الوصول الى الحرف بوضع ascii بالدالة chr

الكود

```
i = 65
c = chr(i)
print( c )
```

التنفيذ

```
A
>>>
```

متغير بالمدي – الدالة range

لاحظ! أنه فيما يلي تم استخدام range لصناعة قائمة سريعة وتحديد نهاية العدد الذي لا يصل اليه

الكود

```
r = range(5)
print( r[0], r[1], r[2], r[3], r[4] )
```

التنفيذ

```
0 1 2 3 4
>>>
```

الكود

```
r = range(7)
print( r[0], r[1], r[2], r[3], r[4], r[5], r[6] )
```

التنفيذ

```
0 1 2 3 4 5 6
>>>
```

لاحظ! أنه فيما يلي تم تحديد البداية ثم النهاية التي لا يصل اليها

الكود

```
r = range( 1 , 8 )
print( r[0], r[1], r[2], r[3], r[4], r[5], r[6] )
```

التنفيذ

```
1 2 3 4 5 6 7
>>>
```

لاحظ! أنه فيما يلي تم استخدام وتحديد الخطوة سواء بالموجب او بالسالب

الكود

```
r = range(1,11 , 2)
print( r[0], r[1], r[2], r[3], r[4] )
```

التنفيذ

```
1 3 5 7 9
>>>
```

الكود

```
r = range(2,11 , 2)
print( r[0], r[1], r[2], r[3], r[4] )
```

التنفيذ

```
2 4 6 8 10
>>>
```

الكود

```
r = range(2,21 , 3)
print( r[0], r[1], r[2], r[3], r[4], r[5], r[6] )
```

التنفيذ

```
2 5 8 11 14 17 20
>>>
```

لاحظ! أنه فيما يلي تم استخدام ord لـ ascii الحرف من الحرف وchr للحرف من ascii الحرف

الكود

```
r = range( ord('A'), ord('Z')+1 )
print(  chr(r[0]),  chr(r[1]),  chr(r[2]),  chr(r[3]),
chr(r[4]), chr(r[5]) )
```

التنفيذ

الكود

```
r = range( 6, 1, -1 )
print( r[0], r[1], r[2], r[3], r[4] )
```

التنفيذ

الكود

```
r = range( 5, 0, -1 )
print( r[0], r[1], r[2], r[3], r[4] )
```

التنفيذ

الكود

```
r = range( 10, 0, -2 )
print( r[0], r[1], r[2], r[3], r[4] )
```

التنفيذ

```
10 8 6 4 2
>>>
```

استلام من المستخدم – الدالة input

لاحظ! أنه فيما يلي انتظرك البرنامج لتدخل قيمة بالدالة input ولاحظ ان هذه القيمة نصية حتى ولو أدخلت رقم

الكود

```
name = input("Enter your name:")
print( 'Hello ' + name )
```

التنفيذ

```
Enter your name:Ahmed
Hello Ahmed
>>>
```

الكود

```
name = input('Enter any name:')
print( type(name) )
```

التنفيذ 1

```
Enter any name:Amr
<class 'str'>
>>>
```

التنفيذ 2

```
Enter any name:123
<class 'str'>
>>>
```

لاحظ! أنه فيما يلي تم تحويل الرقم النصي باستخدام int ليكون قابل لتنفيذ العمليات الحسابية

الكود

```
num1 = int( input('Enter number 1:') )
num2 = int( input('Enter number 2:') )
result = num1 + num2
print( result )
```

التنفيذ

```
Enter number 1:7
Enter number 2:4
11
>>>
```

الاستيراد والعشوائية – import والدالة randint

لاحظ! أنه فيما يلي تم استيراد الـ module الذي سيمي random بالكلمة import وهكذا لباقي الـ modules، ولاحظ ان الدالة randint تنتج رقم عشوائي حسب تحديد الـ min والـ max لها

الكود

```
import random
num = random.randint(1,10); print(num)
num = random.randint(1,10); print(num)
num = random.randint(1,10); print(num)
num = random.randint(1,10); print(num)
num = random.randint(1,10); print(num)
```

التنفيذ 1

```
2
8
5
3
9
>>>
```

التنفيذ 2

```
1
8
4
10
3
>>>
```

الكود

```
from random import randint
num = randint(10,20); print(num)
num = randint(20,30); print(num)
num = randint(30,40); print(num)
num = randint(40,50); print(num)
num = randint(50,60); print(num)
num = randint(60,70); print(num)
```

التنفيذ

```
13
21
32
40
52
64
>>>
```

العوامل المنطقية بالكلمات and, or, not

لاحظ! أن and لا ترجع True الا إذا كل أطرافها True

لاحظ! أن or لا ترجع False الا إذا كل أطرافها False

لاحظ! أن not تعكس القيمة من True الى False والعكس

لاحظ! أن فهم القيم المنطقية لابد منه وسينعكس عليك بالسلب إذا لم تفهمه جيداً

لاحظ! أنه فيما يلي تم تفصيل الموضوع باستفاضة للمعاملات and وor وnot لأهميتهم في عالم البرمجة

الكود

```
bool1 = True and True;    print('true AND true   =',bool1)
bool2 = False and True;   print('false AND true  =',bool2)
bool3 = True and False;   print('true AND false  =',bool3)
bool4 = False and False;  print('false AND false =',bool4)
bool5 = True and True and False and True;  print('T&T&F&T =',bool5)
bool6 = True and True and True and True;   print('T&T&T&T =',bool6)
print('==============')
bool7  = True or True;    print('true OR true    =',bool7)
bool8  = False or True;   print('false OR true   =',bool8)
bool9  = True or False;   print('true OR false   =',bool9)
bool10 = False or False;  print('false OR false  =',bool10)
bool11 = False or False or False or False; print('F|F|F|F =',bool11)
bool12 = False or False or True or False;  print('F|F|T|F =',bool12)
bool13 = True or True or True or True;     print('T|T|T|T =',bool13)
print('==============')
bool14 = not True; print('NOT true   =',bool14)
bool15 = not False; print('NOT false =',bool15)
```

التنفيذ

```
true AND true    = True
false AND true   = False
true AND false   = False
false AND false  = False
T&T&F&T = False
T&T&T&T = True
==============
true OR true     = True
false OR true    = True
true OR false    = True
false OR false   = False
F|F|F|F = False
F|F|T|F = True
T|T|T|T = True
==============
NOT true    = False
NOT false = True
>>>
```

عوامل المقارنة

لاحظ! أنه فيما يلي تم استخدام عوامل المقارنة لاختبار شرط معين

الكود

```
x , y = 7 , 9
b1 = x>y; print(b1)
b2 = x<y; print(b2)
b3 = x>=y; print(b3)
b4 = x<=y; print(b4)
b5 = x==y; print(b5)
b6 = x!=y; print(b6)
```

التنفيذ

```
False
True
False
True
False
True
>>>
```

تحويلات النظم العددية

لاحظ! أنه فيما يلي تم التحويل ما بين النظم العددية بأكثر من طريقة

الكود

```
num_d = 255
num_h = 0xff
num_o = 0o377
num_b = 0b11111111
print( 'Hexadecimal   :', hex(num_d) )
print( 'Octal         :', oct(num_d) )
print( 'Binary        :', bin(num_d) )
print( 'Decimal from h:', int(num_h) )
print( 'Decimal from o:', int(num_o) )
print( 'Decimal from b:', int(num_b) )
```

التنفيذ

```
Hexadecimal   : 0xff
Octal         : 0o377
Binary        : 0b11111111
Decimal from h: 255
Decimal from o: 255
Decimal from b: 255
>>>
```

الكود

```
num_d = 255
num_h = 0xff
num_o = 0o377
num_b = 0b11111111
print( 'Hexadecimal   :', format(num_d,'x') )
print( 'Octal         :', format(num_d,'o') )
print( 'Binary        :', format(num_d,'b') )
print( 'Decimal from h:', format(num_h,'d') )
print( 'Decimal from o:', format(num_o,'d') )
print( 'Decimal from b:', format(num_b,'d') )
```

التنفيذ

```
Hexadecimal   : ff
Octal         : 377
Binary        : 11111111
Decimal from h: 255
Decimal from o: 255
Decimal from b: 255
>>>
```

الكود

```
print( int('11111111',2) )
print( int('377',8) )
print( int('255',10) )
print( int('ff',16) )
```

التنفيذ

```
255
255
255
255
>>>
```

تقطيع النص

لاحظ! أنه فيما يلي تم تقطيع النص بمنتهي السهولة وكأننا نتعامل مع قائمة list ونحدد لها ما نريد بالمدي بين النقطتين : فقط

الكود

```
my = 'Welcome to Hassouna Academy'
print( my[0], my[1], my[2], my[3], my[4], my[5], my[6] )
print( my[:7] )
print( my[0:7] )
print( my[11:] )
print( my[:-len(my)+7] )
print( my[11:len(my)] )
```

التنفيذ

```
W e l c o m e
Welcome
Welcome
Hassouna Academy
Welcome
Hassouna Academy
>>>
```

لاحظ! أنه فيما يلي تم استخدام الدالة split لفصل النص الى قائمة

الكود

```
str_names  = 'Ahmed;Adel;Amr;Ali;Omar;Haitham'
list_names = str_names.split(';')
print( str_names )
print( list_names )
```

التنفيذ

```
Ahmed;Adel;Amr;Ali;Omar;Haitham
['Ahmed', 'Adel', 'Amr', 'Ali', 'Omar', 'Haitham']
>>>
```

توصيل النص

لاحظ! أنه فيما يلي تم توصيل محتوي النص سواء كان نص او list

الكود

```
str1 = 'Hello'
str2 = '-'.join(str1)
print( str1 )
print( str2 )
```

التنفيذ

```
Hello
H-e-l-l-o
>>>
```

الكود

```
list_names = ['Amr','Ali','Ezz']
str_names = ';'.join(list_names)
print( list_names )
print( str_names )
```

التنفيذ

```
['Amr', 'Ali', 'Ezz']
Amr;Ali;Ezz
>>>
```

الكود

```
list_names = ['Amr','Ali','Ezz','Ehab']
str_names = '\n'.join(list_names)
print( list_names )
print( str_names )
```

التنفيذ

```
['Amr', 'Ali', 'Ezz', 'Ehab']
Amr
Ali
Ezz
Ehab
>>>
```

تهيئة النص

لاحظ! أنه فيما يلي تم استخدام الرموز الخاصة بتهيئة النص

الكود

```
name = 'Amr'
my = 'Hello %s' % name
print( my )
```

التنفيذ

```
Hello Amr
>>>
```

الكود

```
num1 = 7
num2 = 9
my = '%d + %d = %d' %(num1, num2, num1+num2)
print( my )
```

التنفيذ

```
7 + 9 = 16
>>>
```

الكود

```
my = '65 is ASCII for %c' % 'A'
print( my )
```

التنفيذ

```
65 is ASCII for A
>>>
```

الكود

```
my  = 'Character %c'     % 'A'
my += '\nString %s'      % 'Hi'
my += '\nDecimal %d'     % 55.99
my += '\nInteger %i'     % 77
my += '\nexponent %e'    % 33
my += '\nExponent %E'    % 33
my += '\nFloat %f'       % 99.77
my += '\nFloat %0.2f'    % 99.77
my += '\nNumber %g,%g'   % ( 3.7 , 99 )
my += '\nOctal %o'       % 65
my += '\nHexadecimal %x' % 65
```

```
print( my )
```

التنفيذ

```
Character A
String Hi
Decimal 55
Integer 77
exponent 3.300000e+01
Exponent 3.300000E+01
Float 99.770000
Float 99.77
Number 3.7,99
Octal 101
Hexadecimal 41
>>>
```

لاحظ! أنه فيما يلي تم استخدام نوع آخر من التهيئة، حيث يتم كتابة أسماء بين اقواس { } في النص ثم يتم تحديد أي قيم لها ليتم عرضها كما نرغب بفضل الله

الكود

```
name      = 'Ahmed'
say_hello = 'Hello {my}'
my_format = say_hello.format(my=name)
print(my_format)
```

التنفيذ

```
Hello Ahmed
>>>
```

الكود

```
my_format = '{n1} + {n2} = {r}'.format(n1=7, n2=3, r=7+3)
print( my_format )
```

التنفيذ

```
7 + 3 = 10
>>>
```

تحولات النص الكبير والصغير ABC, abc

لاحظ! أنه فيما يلي تم تحويل النص upper وlower

الكود

```
str1 = 'HELLO'
str2 = 'welcome'
print( str1.lower() )
print( str2.upper() )
```

التنفيذ

```
hello
WELCOME
>>>
```

تحقق النص

لاحظ! أنه فيما يلي تم التحقق من النص لمعرفة هل هو احرف كبيرة او صغيرة او ارقام او حروف او مسافات او ما الى ذلك، ولاحظ أنه لو

كان التحقق بـ نعم فإن الناتج هو True ولو كان بـ لا فإن الناتج هو False

الكود

```
print( 'HELLO'.isupper() )
print( 'hello'.islower() )
print( 'HEllo'.isalpha() )
print( 'ABC45'.isalnum() )
print( '12345'.isdigit() )
print( '     '.isspace() )
print( '1 AB@'.isprintable() )
print('==========')
print( 'HeLLO'.isupper() )
print( 'hEllo'.islower() )
print( 'HE7lo'.isalpha() )
print( 'AB@45'.isalnum() )
print( '12A45'.isdigit() )
print( '  .  '.isspace() )
print( '\n'.isprintable() )
```

التنفيذ

```
True
True
True
True
True
True
True
==========
False
False
False
False
False
False
False
>>>
```

البحث في النص

لاحظ! أنه فيما يلي تم الوصول الى index الكلمة عن طريق البحث

الكود

```
my = 'Hello Amr and Welcome Back Amr'
indexFind1 = my.find('amr')
indexFind2 = my.find('Amr')
print( indexFind1 )
print( indexFind2 )
```

التنفيذ

```
-1
6
>>>
```

الكود

```
my = 'Hello Amr and Welcome Back Amr'
i  = my.find('Welcome')
print( my[i:] )
```

التنفيذ

```
Welcome Back Amr
>>>
```

استبدال النص

لاحظ! أنه فيما يلي تم استبدال نص بنص آخر

الكود

```
my1 = 'Hello Amr and Welcome Back Amr'
my2 = my1.replace('Amr','Adel')
print( my1 )
print( my2 )
```

التنفيذ

```
Hello Amr and Welcome Back Amr
Hello Adel and Welcome Back Adel
>>>
```

اتخاذ القرار – جملة if

لاحظ! أنه فيما يلي تم تنفيذ الكود متوقف على شرط معين فإذا كان ناتج الشرط بـ True ينفذ وإذا كان بـ False لا ينفذ

الكود

```
x = 5
if x==5: print('OK')
```

التنفيذ

```
OK
>>>
```

الكود

```
x = 7
if x>5:print('OK1');print('OK2');print('OK3')
```

التنفيذ

```
OK1
OK2
OK3
>>>
```

الكود

```
x = 7
if x<=7:
    print('OK1')
    print('OK2')
    print('OK3')
```

التنفيذ

```
OK1
OK2
OK3
>>>
```

الكود

```
num = int( input('Enter any number:') )
if num<0:
    print('Negative Number')
else:
    print('Positive Number')
```

التنفيذ 1

```
Enter any number:-7
Negative Number
>>>
```

التنفيذ 2

```
Enter any number:10
Positive Number
>>>
```

الكود

```
degree = int( input('Enter student degree:') )
if degree<0 or degree>100:
    print('Degree Error')
elif degree<50:
    print('F')
elif degree<60:
    print('E')
elif degree<70:
    print('D')
elif degree<80:
    print('C')
elif degree<90:
    print('B')
else:
    print('A')
```

التنفيذ 1

```
Enter student degree:-1
Degree Error
>>>
```

التنفيذ 2

```
Enter student degree:0
F
>>>
```

التنفيذ 3

```
Enter student degree:40
F
>>>
```

التنفيذ 4

```
Enter student degree:70
C
>>>
```

التنفيذ 5

```
Enter student degree:80
B
>>>
```

التنفيذ 6

```
Enter student degree:95
A
>>>
```

معامل التعبير الشرطي الثلاثي

لاحظ! أنه فيما يلي تم استخدام if على سطر واحد للاختصار

الكود

```
num1 = int( input('Enter Number 1: ') )
num2 = int( input('Enter Number 2: ') )

str_big = 'Number 1' if num1>num2 else 'Number 2'

print( str_big )
```

التنفيذ 1

```
Enter Number 1: 7
Enter Number 2: 5
Number 1
>>>
```

التنفيذ 2

```
Enter Number 1: 5
Enter Number 2: 7
Number 2
>>>
```

حلقات التكرار – الجملة for

لاحظ! أنه فيما يلي تم استخدام حلقات التكرار لاستخدام تكرار الكود وعمل عمليات متعددة بأسطر كود قليلة لتوفير الوقت والجهد

الكود

```
for x in (1,2,3,4,5): print(x)
```

التنفيذ

```
1
2
3
4
5
>>>
```

الكود

```
for x in [10,20,30,40,50]: print(x)
```

التنفيذ

```
10
20
30
40
50
>>>
```

الكود

```
for x in range(2,11 , 2):
    print(x)
```

التنفيذ

```
2
4
6
8
10
>>>
```

الكود

```
for x in range(1,6):
    if x != 4:
        print( x )
```

التنفيذ

الكود

```
for x in range( 5, 0, -1 ):
    print(x)
```

التنفيذ

الكود

```
alpha = ''
for x in range( ord('A'), ord('Z')+1 ):
    alpha += chr(x)
    if x<ord('Z'): alpha += ', '

print( alpha )
```

التنفيذ

```
A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P
, Q, R, S, T, U, V, W, X, Y, Z
>>>
```

الكود

```
alpha = ''
for x in range( ord('Z'), ord('A')-1, -1 ):
    alpha += chr(x)
    if x>ord('A'): alpha += ', '

print( alpha )
```

التنفيذ

```
Z, Y, X, W, V, U, T, S, R, Q, P, O, N, M, L,
K, J, I, H, G, F, E, D, C, B, A
>>>
```

الكود

```
names = ['Ahmed','Adel','Amr','Omar','Ali']
for x in range( len(names) ):
    print( 'Hello ' + names[x] )
```

التنفيذ

```
Hello Ahmed
Hello Adel
Hello Amr
Hello Omar
Hello Ali
>>>
```

الكود

```
names = ['Ahmed','Adel','Amr','Omar','Ali']
for name in names:
    print( 'Hello ' + name )
```

التنفيذ

```
Hello Ahmed
Hello Adel
Hello Amr
Hello Omar
Hello Ali
>>>
```

الكود

```
my_list = [ 3, 'A', True, 5.7 ]
for v in my_list:
    print( v , type(v) )
```

التنفيذ

```
3 <class 'int'>
A <class 'str'>
True <class 'bool'>
5.7 <class 'float'>
>>>
```

الكود

```
emp = { 'name':'Adel', 'city':'Giza', 'salary':3000 }
for x in emp:
    print(x)
```

التنفيذ

```
name
city
salary
>>>
```

الكود

```
emp = { 'name':'Adel', 'city':'Giza', 'salary':3000 }
for x in emp:
    print( emp[x] )
```

التنفيذ

```
Adel
Giza
3000
>>>
```

الكود

```
emp = { 'name':'Ahmed', 'city':'Giza', 'salary':3000 }
for k,v in emp.items():
    print( str(k) + ':' , v )
```

التنفيذ

```
name: Ahmed
city: Giza
salary: 3000
>>>
```

التكرار المتداخل nested loop

لاحظ! أنه فيما يلي تم استخدام التكرار المتداخل والذي يحتاج منك قمة التركيز والفهم ليكون بسيط بالنسبة لك، ولاحظ ان التكرار المتداخل يمكن ان يكون loop داخل loop أو أكثر من ذلك

الكود

```
family1 = ['Ahmed','Adel','Amr']
family2 = ['Ehab','Mahmoud','Ezz']
family3 = ['Sarah','Hajer','Rehab']

home1   = [ family1 , family2 , family3 ]

for x in range( len(home1) ):
    print( 'Family:', x+1 )
    for y in range( len(home1[x]) ):
        print( '   Name', y+1, 'is:', home1[x][y] )
```

التنفيذ

```
Family: 1
   Name 1 is: Ahmed
   Name 2 is: Adel
   Name 3 is: Amr
Family: 2
   Name 1 is: Ehab
   Name 2 is: Mahmoud
   Name 3 is: Ezz
Family: 3
   Name 1 is: Sarah
   Name 2 is: Hajer
   Name 3 is: Rehab
>>>
```

الكود

```
family1 = ['Adel','Amr']
family2 = ['Ehab','Ezz']
family3 = ['Sarah','Hajer']

family4 = ['Ezzat','Foaad']
family5 = ['Abdelrahman','Abdelkareem']
family6 = ['Ali','Akl']

home1   = [ family1 , family2 , family3 ]
home2   = [ family4 , family5 , family6 ]

homes = [ home1 , home2 ]

for x in range( len(homes) ):
    print( 'Home', x+1 )
    for i in range( len(homes[x]) ):
        print( '   Family', i+1 )
        for y in range( len(homes[x][i]) ):
            print( '      Name', y+1, 'is:', homes[x][i][y]
)
```

التنفيذ

```
Home 1
   Family 1
      Name 1 is: Adel
      Name 2 is: Amr
   Family 2
      Name 1 is: Ehab
      Name 2 is: Ezz
   Family 3
      Name 1 is: Sarah
      Name 2 is: Hajer
Home 2
   Family 1
      Name 1 is: Ezzat
      Name 2 is: Foaad
   Family 2
      Name 1 is: Abdelrahman
      Name 2 is: Abdelkareem
   Family 3
      Name 1 is: Ali
      Name 2 is: Akl
>>>
```

انشاء تكرار for بأكثر من متغير

لاحظ! أنه فيما يلي تم استخدام enumerate مع القائمة list
وتم التعامل مع متغيرين داخل التكرار for

الكود

```
for i, name in enumerate(['amr','ali','ezz']):
    print( i, name )
```

التنفيذ

```
0 amr
1 ali
2 ezz
>>>
```

لاحظ! أنه فيما يلي تم استخدام متغير عداد وأيضاً تم استخدام مفاتيح وقيم القاموس في نفس الوقت

الكود

```
person1 = { 'name':'Amr', 'salary':5000 }
person2 = { 'name':'Ali', 'salary':4000 }
person3 = { 'name':'Ezz', 'salary':3000 }
persons = [ person1, person2, person3 ]

for x in range( len(persons) ):
    print( 'Person', x+1 )
    for index, (k, v) in enumerate(persons[x].items()):
        print( '   ', index+1 , ':', k, v )
```

التنفيذ

```
Person 1
    1 : name Amr
    2 : salary 5000
Person 2
    1 : name Ali
    2 : salary 4000
Person 3
    1 : name Ezz
    2 : salary 3000
>>>
```

حلقات التكرار – الجملة while

لاحظ! أنه فيما يلي تم استخدام حلقات التكرار لاستخدام تكرار الكود وعمل عمليات متعددة بأسطر كود قليلة لتوفير الوقت والجهد

الكود

```
x = 1
while x <= 5:
    print( x )
    x +=1
```

التنفيذ

الكود

```
x = 5
while x > 0:
    print( x )
    x -=1
```

التنفيذ

الكود

```
x = 2
while x <= 10:
    print( x )
```

```
    x +=2
```

التنفيذ

```
2
4
6
8
10
>>>
```

الكود

```
x = 1
while x <= 10:
    print( x )
    x +=2
```

التنفيذ

```
1
3
5
7
9
>>>
```

الكود

```
x = 1
while x <= 10:
    print( x )
    x +=2
else:
    print('X After Loop Is:', x)
```

التنفيذ

```
1
3
5
7
9
X After Loop Is: 11
>>>
```

الكود

```
x = 1
while x < 1:
    print( x )
    x +=2
else:
    print('Condition is False')
```

التنفيذ

```
Condition is False
>>>
```

الكود

```
my_list = [7,'A',9.9,False]
x = 0
while x < len(my_list):
    print( my_list[x] , type(my_list[x]) )
    x += 1
```

التنفيذ

```
7 <class 'int'>
A <class 'str'>
9.9 <class 'float'>
False <class 'bool'>
>>>
```

الكود

```
emp = { 'name':'Adel', 'city':'Giza', 'salary':3000 }
my_keys = list(emp.keys())
x = 0
while x < len(emp):
    print( emp[ my_keys[x] ] )
    x += 1
```

التنفيذ

```
Adel
Giza
3000
>>>
```

الكود

```
again = 'y'
while again=='y':
    name = input('Enter your name:')
    print( 'Hello ' + name )
    again = input('Again(y/n)?:')
```

التنفيذ

```
Enter your name:Ahmed
Hello Ahmed
Again(y/n)?:y
Enter your name:Amr
Hello Amr
Again(y/n)?:y
Enter your name:Adel
Hello Adel
Again(y/n)?:n
>>>
```

التكرار اللانهائي

لاحظ! أن التكرار اللانهائي يجعل البرنامج لا يقف ويظل يعمل

الكود

```
x = 1
while True:
    print(x)
    x+=1
```

التنفيذ

```
5159
5160
5161
5162
5163
5164
5165
5166
```

الكود

```
from itertools import count
for x in count():
    print(x)
```

التنفيذ

```
5162
5163
5164
5165
5166
5167
5168
5169
```

الخروج من التكرار - break

لاحظ! أنه فيما يلي تم الخروج من التكرار نهائياً ووقفة تماماً باستخدام الجملة break

الكود

```
for x in range(1,6):
    if x == 4: break
    print( x )
```

التنفيذ

```
1
2
3
>>>
```

الكود

```
x = 1
while x <= 100:
    if x>5:
        break
    print( x )
    x += 1
```

التنفيذ

```
1
2
3
4
5
>>>
```

الكود

```
x = 1
while x <= 100:
    if x>3: break
    print( 'OK', x )
    x += 1
```

التنفيذ

```
OK 1
OK 2
OK 3
>>>
```

الاستكمال في التكرارات - continue

لاحظ! أن الـ continue تتجاهل ما تحتها وتذهب لتستمر بشكل طبيعي في التكرار

الكود

```
numbers = [5,2,0,3,0,7]
mysum = 0
print( 'All Is:', len(numbers) )
for x in range( len(numbers) ):
    if numbers[x]==0: continue
    mysum += numbers[x]
    print('Sum OK Without Zero(s)','x:',x)
print( 'Sum:', mysum )
```

التنفيذ

```
All Is: 6
Sum OK Without Zero(s) x: 0
Sum OK Without Zero(s) x: 1
Sum OK Without Zero(s) x: 3
Sum OK Without Zero(s) x: 5
Sum: 17
>>>
```

انشاء قائمة من التكرار

لاحظ! أنه فيما يلي تم انشاء قائمة من لا شيء باستخدام التكرار

الكود

```
numbers = [ num for num in range(11)]
print( numbers )
```

التنفيذ

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>>
```

الكود

```
numbers = [ chr(num) for num in range(ord('A'),ord('Z')+1)]
print( numbers )
```

التنفيذ

```
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I',
'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', '
S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
>>>
```

الكود

```
numbers = [ num for num in range(21) if num%2==0 ]
print( numbers )
```

التنفيذ

```
[0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
>>>
```

طباعة الوقت والتاريخ

لاحظ! أنه فيما يلي تم استخدام datetime ثم now للوقت والتاريخ الحالي، ثم نستخدم منهم ما نريد، ولاحظ أنه يمكن عمل وقت وتاريخ بالقيم التي نريد

الكود

```
import datetime
```

```
dt1 = datetime.datetime.now()
dt2 = datetime.datetime.now().date()
dt3 = datetime.datetime.now().time()
dt4 = datetime.date(2005,12,31)
dt5 = datetime.time(15,20,1,1234)

print( dt1 )
print( dt2 )
print( dt3 )
print( dt4 )
print( dt5 )
```

التنفيذ

```
2019-07-29 02:47:41.562444
2019-07-29
02:47:41.562444
2005-12-31
15:20:01.001234
>>>
```

تخصيص التاريخ والوقت

لاحظ! أنه فيما يلي تم تخصيص الوقت والتاريخ كما نريد

الكود

```
import datetime

now = datetime.datetime.now()

d  = str(now.day)
m  = str(now.month)
y  = str(now.year)
```

```
h  = str(now.hour)
mi = str(now.minute)
s  = str(now.second)
ms = str(now.microsecond)

print( d + '-' + m + '-' + y )
print( y + '/' + m + '/' + d )
print( d + '-' + m + '-' + y + '\t' + h +':'+ mi +':'+ s )
```

التنفيذ

```
29-7-2019
2019/7/29
29-7-2019       17:33:57
>>>
```

تهيئة التاريخ والوقت

لاحظ! أنه فيما يلي تم عمل تهيئة للوقت والتاريخ لتسهل علينا عمليات طباعة ما نريد

الكود

```
import datetime

now = datetime.datetime.now()

print( 'Long day name   :', now.strftime('%A') )
print( 'Short day name  :', now.strftime('%a') )
print( 'Long month name :', now.strftime('%B') )
print( 'Short month name:', now.strftime('%b') )
print( 'Date time       :', now.strftime('%c') )
print( 'Day of month    :', now.strftime('%d') )
```

```
print( 'Hour number 24  :', now.strftime('%H') )
print( 'Hour number 12  :', now.strftime('%I') )
print( 'Day of year     :', now.strftime('%j') )
print( 'Month of year   :', now.strftime('%m') )
print( 'Minutes         :', now.strftime('%M') )
print( 'AM or PM        :', now.strftime('%p') )
print( 'Seconds         :', now.strftime('%S') )
print( 'Short date      :', now.strftime('%x') )
print( 'Short time      :', now.strftime('%X') )
print( 'Short year      :', now.strftime('%y') )
print( 'Long year       :', now.strftime('%Y') )

my_format = '%d/%m/%Y - %I:%M:%S %p'
print( 'Date time 12    :', now.strftime(my_format) )
```

التنفيذ

```
Long day name    : Monday
Short day name   : Mon
Long month name  : July
Short month name: Jul
Date time        : Mon Jul 29 17:54:10 2019
Day of month     : 29
Hour number 24   : 17
Hour number 12   : 05
Day of year      : 210
Month of year    : 07
Minutes          : 54
AM or PM         : PM
Seconds          : 10
Short date       : 07/29/19
Short time       : 17:54:10
Short year       : 19
Long year        : 2019
Date time 12     : 29/07/2019 - 05:54:10 PM
>>>
```

فتح ملف نصي موجود للقراءة منه

لاحظ! أنه فيما يلي تم استخدام open لفتح ملف موجود

الكود

```
file = open( 'my_file.txt' )
file.close()
```

التنفيذ

يتم فتح الملف واغلاقه، ولو كان غير موجود يحدث خطأ

انشاء ملف نصي فارغ للكتابة عليه

لاحظ! أنه فيما يلي تم استخدام w ومعني ذلك ان الملف سيتم إنشاؤه، ولو كان موجود سيتم حذفه ثم انشاء ملف جديد، ولاحظ أنه لابد من اغلاق الملف بالدالة close بعد استخدامه

الكود

```
file = open( 'my_file.txt' , 'w' )
file.close()
```

التنفيذ

يتم انشاء الملف في نفس مكان ملف الأكواد فقط ثم اغلاقه

انشاء ملف نصي والكتابة عليه

لاحظ! أنه فيما يلي تم إضافة كلام على الملف باستخدام الدالة write

الكود

```
file = open( 'my_file.txt' , 'w' )
file.write('Hello Abdelhamid\nWelcome Amr')
file.write(' Hi')
file.close()
```

التنفيذ

الكود

```
file = open( 'my_file.txt' , 'w' )
names = ['Amr','Ali','Ezz']
file.writelines(names)
file.close()
```

التنفيذ

القراءة العادية من ملف موجود

الكود

```
f = open( 'my_file.txt' , 'r' )
names = f.read()
f.close()
print( names )
```

التنفيذ

```
Adel
Amr
Ali
Ezz
>>>
```

القراءة من ملف موجود وبه سطر واحد

لاحظ! أنه فيما يلي تم استخدام readlines لجلب محتوي الملف في قائمة list

الكود

```
f = open( 'my_file.txt' , 'r' )
names = f.readlines()
f.close()
print( names )
```

التنفيذ

```
['AmrAliEzz']
>>>
```

القراءة من ملف موجود وبه عدة أسطر

لاحظ! أنه فيما يلي تم استخدام readlines حيث كل سطر محتوي من محتويات القائمة list مع إضافة \n أي سطر جديد في كل عنصر

الكود

```
f = open( 'my_file.txt' , 'r' )
names = f.readlines()
f.close()
print( names )
```

التنفيذ

```
['Adel\n', 'Amr\n', 'Ali\n', 'Ezz']
>>>
```

القراءة من ملف من مكان معين

لاحظ! أنه فيما يلي تم استخدام seek لإيقاف مؤشر القراءة في مكان معين قبل القراءة، ولاحظ ان read تقرأ الى النهاية

الكود

```
f = open( 'my_file.txt' , 'r' )
f.seek(2)
names = f.read()
f.close()
print( names )
```

التنفيذ

```
el
Amr
Ali
Ezz
>>>
```

القراءة من ملف سطر تلو السطر

لاحظ! أنه فيما يلي تم استخدام readline والتي تبدأ بقراءة اول سطر ثم التالي ثم الذي يليه وهكذا

الكود

```
f = open( 'my_file.txt' , 'r' )
print( f.readline() )
print( f.readline() )
print( f.readline() )
print( f.readline() )
f.close()
```

التنفيذ

```
Adel

Amr

Ali

Ezz
>>>
```

لاحظ! أن الدالة readline تكون بصفر او بـ False عند الوصول الى نهاية الملف

الكود

```
f = open( 'my_file.txt' , 'r' )
line = True
while line:
    line = f.readline()
    print( line )
f.close()
```

التنفيذ

```
Adel

Amr

Ali

Ezz

>>>
```

تكملة الكتابة على ملف نصي موجود

لاحظ! أنه فيما يلي تم استخدام a لعمل Append وتكملة الكتابة على الملف دون مساس الكلام القديم

الكود

```
f = open( 'my_file.txt' , 'a' )
f.write('Omar')
f.close()
```

التنفيذ

تكملة الكتابة مع القراءة من ملف نصي

لاحظ! أنه فيما يلي تم استخدام a+ للتكملة ومعها يمكن القراءة

الكود

```
f = open( 'my_file.txt' , 'a+' )
f.seek(0)
print( f.readlines() )
f.write('\nOmar')
f.seek(0)
print( f.readlines() )
f.close()
```

التنفيذ

```
['Adel\n', 'Amr\n', 'Ali\n', 'Ezz']
['Adel\n', 'Amr\n', 'Ali\n', 'Ezz\n', 'Omar']
>>>
```

القراءة مع الكتابة لملف نصي

لاحظ! أنه فيما يلي تم استخدام w+ للكتابة ومعها يمكن القراءة

الكود

```
f = open( 'my_file.txt' , 'w+' )
f.write('Ahmed\n')
f.seek(0); print( f.readlines() )
f.write('Omar\n')
f.seek(0); print( f.readlines() )
f.write('Adel\n')
f.seek(0); print( f.readlines() )
f.close()
```

التنفيذ

```
['Ahmed\n']
['Ahmed\n', 'Omar\n']
['Ahmed\n', 'Omar\n', 'Adel\n']
>>>
```

الكتابة مع القراءة

لاحظ! أنه فيما يلي تم استخدام r+ للقراءة ومعها يمكن الكتابة

الكود

```
f = open( 'my_file.txt' , 'r+' )
f.seek(0); print( f.readlines() )
f.write('Ammar\n')
f.seek(0); print( f.readlines() )
f.write('Yaser\n')
f.seek(0); print( f.readlines() )
f.close()
```

التنفيذ

```
['Ahmed\n', 'Omar\n', 'Adel\n']
['Ahmed\n', 'Omar\n', 'Adel\n', 'Ammar\n']
['Ahmed\n', 'Omar\n', 'Adel\n', 'Ammar\n', 'Yaser\n']
>>>
```

القراءة والكتابة والتكملة لملف binary

لاحظ! أنه فيما يلي تم استخدام نفس الأنماط السابقة مثل w و r و r+ وغيرها ولكن مع زيادة حرف b بعدها للـ Binary

الكود

```
f = open( 'my_file.jpg' , 'wb' )
f = open( 'my_file.jpg' , 'rb' )
f = open( 'my_file.jpg' , 'wb+' )
f = open( 'my_file.jpg' , 'rb+' )
f = open( 'my_file.jpg' , 'ab' )
f = open( 'my_file.jpg' , 'ab+' )
f.close()
```

التنفيذ

الكود هنا للمعرفة، والتعامل معه بعد ذلك حسب الطلب لملف الـBinary

الجملة with لغلق الأشياء بعد استخدامها

لاحظ! أنه فيما يلي تم غلق الملف تلقائياً لأنه تم استخدامه مع with

الكود

```
with open('my_file.txt','r') as file:
    print( file.readlines() )
```

```
    print( 'Closed: ', file.closed )

print( 'Closed: ', file.closed )
```

التنفيذ

```
['Ahmed\n', 'Omar\n', 'Adel\n', 'Ammar\n', 'Yaser\n']
Closed:  False
Closed:  True
>>>
```

التحقق من وجود ملف او مجلد وانشاء المجلدات

الكود

```
import os
if not os.path.exists('My New Folder 1'):
    os.mkdir('My New Folder 1')
if not os.path.exists('My New Folder 2'):
    os.makedirs('My New Folder 2')
if not os.path.exists('my_file.txt'):
    f = open( 'my_file.txt' , 'w' )
    f.close()
```

التنفيذ

سوف يتم انشاء المجلدات والملف اذ لم يكونوا موجودين

حذف الملفات والمجلدات

الكود

```
import os
if os.path.exists('My New Folder 1'):
    os.rmdir('My New Folder 1')
```

```
if os.path.exists('My New Folder 2'):
    os.rmdir('My New Folder 2')
if os.path.exists('my_file.txt'):
    os.remove('my_file.txt')
```

التنفيذ

سيتم حذف المجلدات والملف إذا كانوا موجودين

معرفة الملفات والمجلدات الموجودة بمجلد معين

لاحظ! أن الدالة listdir تأتي بجميع الملفات والمجلدات للمسار المحدد لها داخل الاقواس، ولاحظ أن الدالة isfile تخبر عن ما اذا كان ملف ام لا

الكود

```
import os
files = os.listdir('New Folder')
for file in files:
    if os.path.isfile('New Folder/' + file):
        print('File  :', file )
    else:
        print('Folder:', file )
```

التنفيذ

```
File  : New Bitmap Image.bmp
Folder: New folder
Folder: New folder (2)
Folder: New folder (3)
File  : New Microsoft Excel Worksheet.xlsx
File  : New Text Document.txt
File  : New WinRAR archive.rar
>>>
```

انشاء قائمة الملفات والمجلدات

الكود

```
import os
all_files = [f for f in os.listdir('New Folder')]
for file in all_files:
    print( file )
```

التنفيذ

```
New Bitmap Image.bmp
New folder
New folder (2)
New folder (3)
New Microsoft Excel Worksheet.xlsx
New Text Document.txt
New WinRAR archive.rar
>>>
```

تنفيذ أوامر الدوس

الكود

```
import os
os.system('mkdir my_folder_using_dos')
```

التنفيذ

سيتم انشاء مجلد باستخدام نظام DOS، أو أي امر آخر حسب الاحتياج.

نسخ الملفات - copy

الكود

```
import shutil
shutil.copy2( 'New Folder/Image.bmp' , 'NewFile.bmp' )
```

التنفيذ

سيتم نسخ الملف من المكان الأول الى المكان الثاني

نقل الملفات - cut

الكود

```
import shutil
shutil.move( 'New Folder/Image.bmp' , 'NewFile.bmp' )
```

التنفيذ

سيتم نقل الملف من المكان الأول الى المكان الثاني

نسخ المجلدات

الكود

```
import shutil
shutil.copytree('New Folder/New Folder (3)' , 'New' )
```

التنفيذ

سيتم نسخ المجلد من المكان الأول الى المكان الثاني

Regular Expression استخدام

الكود

```
import re

pattern = '^[A-Z][a-z]{1,15}$'
text    = input('Enter first name:')

v = re.match( pattern , text )
if v != None: print( 'Correct' )
else: print( 'Incorrect' )
```

التنفيذ 1

```
Enter first name:a7mad
Incorrect
>>>
```

التنفيذ 2

```
Enter first name:ahmed
Incorrect
>>>
```

التنفيذ 3

```
Enter first name:Ahmed
Correct
>>>
```

استخدام الرياضيات math

الكود

```
import math
```

```
print( 'PI       :', math.pi )
print( 'SQRT     :', math.sqrt(81) )
print( 'Round    :', round(1.5) )
print( 'Round    :', round(1.4) )
print( 'Ceil     :', math.ceil(1.1) )
print( 'Floor    :', math.floor(1.99) )
print( 'Absolute :', math.fabs(-50) )
print( 'Absolute :', abs(-10) )
print( 'Bower    :', math.pow(5,3) )
print( 'Bower    :', pow(5,3) )
print( 'Factorial:', math.factorial(5) )
print( 'Summation:', math.fsum( [1,2,3,4,5,6] ) )
print( 'Summation:', sum( [1,2,3,4,5,6] ) )
s = [2500,6700,8400,2500,3400,2100]
print( 'Average  :', sum(s) / len(s) )
```

التنفيذ

```
PI       : 3.141592653589793
SQRT     : 9.0
Round    : 2
Round    : 1
Ceil     : 2
Floor    : 1
Absolute : 50.0
Absolute : 10
Bower    : 125.0
Bower    : 125
Factorial: 120
Summation: 21.0
Summation: 21
Average  : 4266.666666666667
>>>
```

معالجة الأخطاء

الكود

```
try:
    num1, num2 = int(input('Num1:')) , int(input('Num2:'))
    result = num1/num2
    print( result )
    names = ['Amr','Ali']
    print( names[int(input('Name Index:'))] )
except ZeroDivisionError as ex1:
    print( ex1 )
except IndexError as ex2:
    print( ex2 )
except:
    print( 'Unknown Error' )
finally:
    print( 'Finally' )
```

التنفيذ 1

```
Num1:7
Num2:3
2.3333333333333335
Name Index:0
Amr
Finally
>>>
```

التنفيذ 2

```
Num1:7
Num2:0
division by zero
Finally
>>>
```

التنفيذ 3

```
Num1:7
Num2:3
2.3333333333333335
Name Index:2
list index out of range
Finally
>>>
```

التعامل مع ملفات csv

اولاً انشاء ملف info.csv وبه المعلومات التالية

```
ahmed,amr,adel,ehab,mahmoud
omar,ammar,yasser,haytham
1500,2500,3500,4500,5500
```

الكود

```
import csv

f = open('info.csv')
r = csv.reader(f)

g1 = next(r)
print(g1)

g2 = next(r)
print(g2)

g3 = next(r)
print(g3)

f.close()
```

التنفيذ

```
['ahmed', 'amr', 'adel', 'ehab', 'mahmoud']
['omar', 'ammar', 'yasser', 'haytham']
['1500', '2500', '3500', '4500', '5500']
>>>
```

انشاء دوال functions عادية لا تنفذ شيء

لاحظ! أنه تم تعريف ثلاث دوال ليس بهم اكواد وpass لعدم التنفيذ

الكود

```
def my_func1():
    '''This is my function 1'''
def my_func2():
    '''This is my function 1'''
def my_func3():
    pass
```

التنفيذ

لا شيء سينفذ

انشاء دوال بها اكواد ثم النداء عليها

الكود

```
def my_func1():
    '''This is my function 1'''
    print('Welcome to function 1')
    print('Function 1 is easy')

def my_func2():
    '''This is my function 1'''
    print('Welcome to function 2')
```

```
    print('Function 2 is easy')

def my_func3():
    print('Welcome to function 3')
    print('Function 3 is easy')

my_func2()
my_func3()
my_func1()
```

التنفيذ

```
Welcome to function 2
Function 2 is easy
Welcome to function 3
Function 3 is easy
Welcome to function 1
Function 1 is easy
>>>
```

انشاء دالة تستقبل وسائط

لاحظ! أنه فيما يلي تم عمل دالة باسم say_hello ولكن هذه الدالة تستقبل متغير باسم name ثم يتم استخدام هذا المتغير بداخلها فقط

لاحظ! أن الدوال مفيدة لأنك تستخدمها أكثر من مرة وباستخدامات مختلفة لنفس الدالة

لاحظ! أن الدوال توفر الوقت والجهد، ومفيدة جداً ويجب التركيز المكثف عليها

الكود

```
def say_hello( name ):
```

```
    print( 'Hello ' + name )

say_hello( 'Ahmed' )
say_hello( 'Adel' )
say_hello( 'Amr' )
```

التنفيذ

```
Hello Ahmed
Hello Adel
Hello Amr
>>>
```

لاحظ! أنه فيما يلي تم انشاء دالة تستقبل الرقم الأول والرقم الثاني والعملية operation لتقوم بحساب ما تريد عند النداء عليها سواء كان جمع او طرح او قسمة او ضرب في نفس الدالة

الكود

```
def calc( num1 , num2 , ope ):
    if ope == '+':
        print( num1 + num2 )
    elif ope == '-':
        print( num1 - num2 )
    elif ope == '*':
        print( num1 * num2 )
    elif ope == '/':
        if num2==0: num2=1
        print( num1 / num2 )
    elif ope == '%':
        if num2==0: num2=1
        print( num1 % num2 )

calc( 7, 3, '+' )
calc( 7, 3, '-' )
```

```
calc( 7, 3, '*' )
calc( 7, 3, '/' )
calc( 7, 3, '%' )
```

التنفيذ

```
10
4
21
2.3333333333333335
1
>>>
```

انشاء دالة بوسائط لا نهائية

لاحظ! أنه فيما يلي تم استخدام الدالة بوسيط او أكثر، ولاحظ أن كل الوسائط تتجمع في Tuple واحد

الكود

```
def names(*names):
    print( type(names), names )

names('Adel','Amr','Ali','Ezz','Akl')
```

التنفيذ

```
<class 'tuple'> ('Adel', 'Amr', 'Ali', 'Ezz', 'Akl')
>>>
```

الكود

```
def get_values(*values):
    print( values )
    for val in values:
```

```
        print( type(val), val )

get_values(111,'Amr',4655.50,True)
```

التنفيذ

```
(111, 'Amr', 4655.5, True)
<class 'int'> 111
<class 'str'> Amr
<class 'float'> 4655.5
<class 'bool'> True
>>>
```

انشاء دالة ترجع قيمة

لاحظ! أنه فيما يلي تم ارجاع قيمة، أي أنه عند النداء على هذه الدالة سوف ترجع قيمة يمكن طباعتها او وضعها في متغير

لاحظ! أن الكود بعد الارجاع return ليس له أي قيمة

الكود

```
def my_func():
    print('Before Return')
    return 'Test Return'
    print('After Return')

my_return_val = my_func()
print( my_return_val )
```

التنفيذ

```
Before Return
Test Return
>>>
```

الكود

```
def say_hello( name ):
    return 'Hello ' + name

print( say_hello('Ahmed') )
print( say_hello('Adel') )
print( say_hello('Amr') )
```

التنفيذ

```
Hello Ahmed
Hello Adel
Hello Amr
>>>
```

انشاء دالة تعتمد على فكرة القاموس

لاحظ! أنه فيما يلي تم استخدام ارجاع قيمة لقاموس بداخل الدالة، بحيث القيمة التي سوف يرسلها للدالة هي التي سيتم ارجاعها من القاموس

الكود

```
def num_name(number):
    return {
        0:'zero', 1:'one', 2:'two', 3:'three',
        4:'four', 5:'five', 6:'siz', 7:'seven',
        8:'eight', 9:'nine', 10:'ten'
    }[number]

print( num_name(0) )
print( num_name(1) )
print( num_name(2) )
print( num_name(3) )
```

```
print( num_name(4) )
print( num_name(5) )
print( num_name(6) )
print( num_name(7) )
print( num_name(8) )
print( num_name(9) )
print( num_name(10) )
```

التنفيذ

```
zero
one
two
three
four
five
siz
seven
eight
nine
ten
>>>
```

استخدام تعبيرات Lambda كأنها دالة

الكود

```
calc = lambda num1, num2: num1 + num2
print( calc(7,3) )
print( calc(5,8) )
print( calc(6,9) )
```

التنفيذ

```
10
13
15
>>>
```

انشاء دالة مرجعية تنادي على نفسها

لاحظ! أنه فيما يلي تم انشاء دالة لحساب وارجاع المضروب factorial

لاحظ! أن الدالة المرجعية لها فهم خاص، ولفهمها جيداً ننصح بالدخول على يوتيوب والبحث عن كورس بايثون اكاديمية حسونة ثم مذاكرة الدرس رقم 84 جيداً

الكود

```
def fact(num):
    if num == 0: return 1
    else: return num * fact( num - 1 )

print( 'Factorial 3 is:', fact(3) )
print( 'Factorial 4 is:', fact(4) )
print( 'Factorial 5 is:', fact(5) )
```

التنفيذ

```
Factorial 3 is: 6
Factorial 4 is: 24
Factorial 5 is: 120
>>>
```

انشاء Module كمكتبة أكواد قابلة للتنفيذ

أولاً انشاء ملف باسم my.py في نفس مكان الملف التنفيذي الذي نجرب عليه الأكواد الأساسية، أما الملف my.py هو الذي يعبر عن الموديول

ثانياً كتابة الكود التالي في الملف my.py وحفظه

```
def say_hello( name ):
    print( 'Hello ' + name )

def calc( num1 , num2 , ope ):
    if ope == '+':
        print( num1 + num2 )
    elif ope == '-':
        print( num1 - num2 )
    elif ope == '*':
        print( num1 * num2 )
    elif ope == '/':
        if num2==0: num2=1
        print( num1 / num2 )
    elif ope == '%':
        if num2==0: num2=1
        print( num1 % num2 )
```

ثالثاً تجربة ما يلي على ملف الأكواد الأساسي

الكود

```
import my
my.calc(10,3,'+')
my.calc(10,3,'-')
my.calc(10,3,'*')
my.calc(10,3,'/')
my.calc(10,3,'%')
my.say_hello('Ahmed')
```

التنفيذ

```
13
7
30
3.3333333333333335
1
Hello Ahmed
>>>
```

الكود

```
from my import calc
from my import say_hello
calc(81,18,'+')
calc(150,51,'-')
calc(33,3,'*')
calc(495,5,'/')
calc(23,12,'%')
say_hello('Adel')
```

التنفيذ

```
99
99
99
99.0
11
Hello Adel
>>>
```

تنفيذ أكواد من ملف خارجي أو من نص

أولاً انشاء ملف باسم mycode.py في نفس مكان الملف التنفيذي الذي نجرب علية الأكواد الأساسية

ثانياً كتابة الكود التالي في الملف mycode.py وحفظه

```
print( 'Test From mycode.py' )
for x in range(3):
    print( 'Test From mycode.py ' + str(x) )

def calc( num1 , num2 , ope ):
    if ope == '+':
        print( num1 + num2 )
    elif ope == '-':
        print( num1 - num2 )
    elif ope == '*':
        print( num1 * num2 )
    elif ope == '/':
        if num2==0: num2=1
        print( num1 / num2 )
    elif ope == '%':
        if num2==0: num2=1
        print( num1 % num2 )
```

ثالثاً تجربة ما يلي على ملف الأكواد الأساسي

الكود

```
exec( open('mycode.py').read() )
exec( 'print("This Code from string")' )
```

التنفيذ

```
Test From mycode.py
Test From mycode.py 0
Test From mycode.py 1
Test From mycode.py 2
This Code from string
>>>
```

الكود

```
exec( open('mycode.py').read() )
calc(81,18,'+')
calc(150,51,'-')
calc(33,3,'*')
calc(495,5,'/')
calc(23,12,'%')
```

التنفيذ

```
Test From mycode.py
Test From mycode.py 0
Test From mycode.py 1
Test From mycode.py 2
99
99
99
99.0
11
>>>
```

الكود

```
exec("""
for x in range(3):
    print( 'X=' + str(x) )
""")
```

التنفيذ

```
X=0
X=1
X=2
>>>
```

اظهار المساعدة help

لاحظ! أنه فيما يلي تم استخدام الدالة help لإظهار المساعدة لأي موضوع، سواء كنت انت من صنعة مثل الدوال وغيرها او كان موجود مثل print وstr وtuple وغيرها حتى لو كان موديول صنعته بنفسك او موديول موجود مسبقاً في اللغة، حتى لو كان متغير فإنه يخبرك ان تستخدم ما يعادله في اللغة

الكود

```
def say_hello( name ):
    '''Send any name for say hello'''
    print( 'Hello ' + str(name) )

help( say_hello )
```

التنفيذ

```
Help on function say_hello in module __main__:

say_hello(name)
    Send any name for say hello

>>>
```

الكود

```
help( print )
```

التنفيذ

```
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.

>>>
```

الكود

```
import my
help( my )
```

التنفيذ

```
Help on module my:

NAME
    my

FUNCTIONS
    calc(num1, num2, ope)

    say_hello(name)

FILE
    c:\users\usernowa\desktop\my.py


>>>
```

الكود

```
help( 'Hello' )
```

التنفيذ

```
No Python documentation found for 'Hello'.
Use help() to get the interactive help utility.
Use help(str) for help on the str class.

>>>
```

استكشاف ما في أشياء لغة البايثون dir

لاحظ! أنه فيما يلي تم اظهار كل المستخدم في ملف الاكواد الخاص بك

لاحظ! أنه عند زيادة متغير او دالة او عمل import ثم عمل dir يظهر كمل ذلك مع الموجود، ويمكن البحث dir عن متغير او دالة معينة

الكود

```
print( dir() )
```

التنفيذ

```
['__annotations__', '__builtins__', '__doc__',
'__file__', '__loader__', '__name__', '__packa
ge__', '__spec__']
>>>
```

الكود

```
import math
import os
my_name = 'Mohamed'
def get_name():return 'Mohamed'
def print_name():print(get_name())
print( dir() )
```

التنفيذ

```
['__annotations__', '__builtins__', '__doc__',
'__file__', '__loader__', '__name__', '__packa
ge__', '__spec__', 'get_name', 'math', 'my_nam
e', 'os', 'print_name']
>>>
```

الكود

```
my_name = 'Mohamed'
def get_name():return 'Mohamed'
def print_name():print(get_name())
print( dir(get_name) )
```

التنفيذ

```
['__annotations__', '__call__', '__class__', '
__closure__', '__code__', '__defaults__', '__d
elattr__', '__dict__', '__dir__', '__doc__', '
__eq__', '__format__', '__ge__', '__get__', '_
_getattribute__', '__globals__', '__gt__', '__
hash__', '__init__', '__init_subclass__', '__k
wdefaults__', '__le__', '__lt__', '__module__'
, '__name__', '__ne__', '__new__', '__qualname
__', '__reduce__', '__reduce_ex__', '__repr__'
, '__setattr__', '__sizeof__', '__str__', '__s
ubclasshook__']
>>>
```

انشاء class به خصائص فقط

لاحظ! أنه فيما يلي تم عمل كلاس باسم employee ويحمل الرقم number والاسم name والعنوان address والمرتب salary والفاعلية active

لاحظ! أن الخصائص داخل الكلاس عبارة عن متغيرات

الكود

```
class employee:
    number  = 0
    name    = ''
    address = ''
    salary  = 0.0
    active  = True
```

التنفيذ

لا يوجد شيء يتنفذ لأنه لم يتم استخدام الكلاس

انشاء class به وظائف فقط

لاحظ! أن الوظائف للكلاس هي الدوال، ولابد من استلامها لمتغير للكائن

الكود

```
class my_print:
    def print1(self):
        print('Test Print 1')
    def print2(self):
        print('Test Print 2')
    def print3(self):
        print('Test Print 3')
```

التنفيذ

لا يوجد شيء يتنفذ لأنه لم يتم استخدام الكلاس

انشاء class به خصائص ووظائف معاً

لاحظ! أنه فيما يلي لابد من تعريف متغير داخل كل دالة بالكلاس ليعبر هذا المتغير عن الكائن الذي سوف يتم تعريفه، مع العلم أن هذا المتغير لا يتم إرساله عند النداء على الدالة، ولكن هو افتراضي للكائن

لاحظ! أنه فيما يلي تم النداء على عناصر الكلاس داخل الدوال باستخدام المتغير الافتراضي للدالة سواء سميناه self وهو المشهور او غيره

الكود

```
class employee:
    number  = 0
    name    = ''
    address = ''
    salary  = 0.0
    active  = True

    def get_data(o):
        info =  str(o.number)+';'+o.name+';'+o.address
        info += ';'+str(o.salary)+';'+str(o.active)
        return info

    def print_data(o):
        print( o.get_data() )
```

التنفيذ

لا يوجد شيء يتنفذ لأنه لم يتم استخدام الكلاس

انشاء class واستخدامه

لاحظ! أنه فيما يلي تم استخدام الكلاس عن طريق تعريف كائن object منه باسم emp1

الكود

```
class employee:
    number  = 0
    name    = ''
    address = ''
    salary  = 0.0
    active  = True
    def get_data(o):
        info =  str(o.number)+';'+o.name+';'+o.address
        info += ';'+str(o.salary)+';'+str(o.active)
        return info
    def print_data(o):
        print( o.get_data() )

emp1 = employee()
emp1.number  = 1
emp1.name    = 'Adel'
emp1.address = 'Giza'
emp1.salary  = 9500.5
emp1.print_data()
```

التنفيذ

```
1;Adel;Giza;9500.5;True
>>>
```

انشاء class وعمل أكثر من object له

لاحظ! أنه فيما يلي تم ارسال متغيرين model وcolor للدالة set_data وكأن المتغير self غير موجود

الكود

```
class car:
    model = ''
    color = ''
    def set_data(self, model, color):
        self.model = model
        self.color = color
    def get_data(self):
        return self.model+' , '+self.color
    def print_data(self):
        print( self.get_data() )

car1 = car()
car2 = car()
car3 = car()

car1.set_data('Renault BMW','Red')
car2.set_data('Audi Mercedes Benz','Silver')
car3.set_data('MG Motor Maruti Suzuki','White')

car1.print_data()
car2.print_data()
car3.print_data()
```

التنفيذ

```
Renault BMW , Red
Audi Mercedes Benz , Silver
MG Motor Maruti Suzuki , White
>>>
```

دالة البناء في الكلاس

لاحظ! أنه فيما يلي تم كتابة كود بدالة البناء __init__ للكلاس employee وهذه الدالة يتم تنفيذها تلقائياً عند تعريف كائن جديد

الكود

```
class employee:
    def __init__(self):
        print( 'New object from ' + str( type(self) ) )

emp1 = employee()
emp2 = employee()
emp3 = employee()
```

التنفيذ

```
New object from <class '__main__.employee'>
New object from <class '__main__.employee'>
New object from <class '__main__.employee'>
>>>
```

وسائط مع دالة البناء

لاحظ! أنه فيما يلي تم ارسال الوسائط اجباري مع تعريف كل كائن من الكلاس لأنه يتم طلبها في دالة البناء

الكود

```
class employee:
    emp_id   = 0
    emp_name = ''
```

```
    def __init__(self, emp_id, emp_name):
        self.emp_id   = emp_id
        self.emp_name = emp_name

    def print_data(self):
        print( self.emp_id, self.emp_name )

emp1 = employee(1,'Ahmed')
emp2 = employee(2,'Adel')
emp3 = employee(3,'Amr')

emp1.print_data()
emp2.print_data()
emp3.print_data()
```

التنفيذ

```
1 Ahmed
2 Adel
3 Amr
>>>
```

انشاء عداد وindex للـ objects

لاحظ! أنه فيما يلي تم عمل عداد لعدد الكائنات objects_count
وكل كائن يحتفظ بالـ index الخاص به

الكود

```
from itertools import count

class employee:
    objects_count = count(0)
    index = 0
```

```
    def __init__(self):
        self.index = next( self.objects_count )

e1 = employee()
e2 = employee()
e3 = employee()
print( e1.index, e2.index, e3.index )
print( e1.objects_count )
```

التنفيذ

```
0 1 2
count(3)
>>>
```

دالة الهدم في الكلاس

لاحظ! أنه فيما يلي تم استخدام دالة الهدم __del__ والتي تنفذ عند حذف الكائن

الكود

```
class employee:
    def __init__(self):
        print('Create object from employee')

    def __del__(self):
        print('Object is deleted')

e1 = employee()
e2 = employee()
e3 = employee()
del e1
del e2
del e3
```

التنفيذ

```
Create object from employee
Create object from employee
Create object from employee
Object is deleted
Object is deleted
Object is deleted
>>>
```

تضمين خصائص جديدة للكائن

لاحظ! أنه فيما يلي تم استخدام خاصية الـ number وخاصية الـ name مع انهم لم يتم بنائهم مع بناء الكلاس، ولاحظ أنه يوجد أكثر من طريقة لعمل ذلك

الكود

```
class employee:
    pass

e1 = employee()
e1.number = 1
e1.name   = 'Amr'
print( e1.number, e1.name )
```

التنفيذ

```
1 Amr
>>>
```

الكود

```
class employee:
```

```
    pass

e1 = employee()
setattr( e1, 'number', 1 )
setattr( e1, 'name', 'Amr')
print( e1.number, e1.name )
```

التنفيذ

```
1 Amr
>>>
```

تضمين خصائص تنفذ أكواد

لاحظ! أنه فيما يلي تم النداء على الخاصية فقط لتقوم بتنفيذ كود محدد لها وكأنها دالة

الكود

```
class employee:
    pass

e1 = employee()
setattr( e1, 'test', exec('print("test")') )
e1.test
```

التنفيذ

```
test
>>>
```

انشاء عناصر سرية داخل الـ class

لاحظ! أن كل ما تبدأ تسميته بالرمز underscore مرتين __ او أكثر بدون إضافة underscore في النهاية يصبح عنصر سري سواء كان خاصية او دالة، ولو تم النداء عليه لاستخدامه سيحدث خطأ

الكود

```
class my_class:
    __my_attr1__ = 'attr1'
    __my_attr2   = 'attr2'
    ___my_attr3  = 'attr3'

    def func1(self):
        print( self.__my_attr1__ )
        self.__func2()
        self.__func3()

    def __func2(self):
        print( self.__my_attr2 )

    def __func3(self):
        print( self.___my_attr3 )

my = my_class()
my.__my_attr1__ += ', OK'
my.func1()
```

التنفيذ

```
attr1, OK
attr2
attr3
>>>
```

التحقق من وجود عنصر داخل الكائن

لاحظ! أنه فيما يلي تم اختبار وجود خاصية وأيضاً تم اختبار وجود دالة داخل الكائن عن طريق الدالة hasattr

لاحظ! أنه بهذه الطريقة نستطيع اختبار وجود الشيء قبل استخدامه لتفادي الأخطاء

الكود

```
class employee:
    def test():
        print( 'Employee' )

emp1 = employee()
emp2 = employee()
emp1.name = 'Ahmed'

print( hasattr( emp1, 'name' ) )
print( hasattr( emp2, 'name' ) )
print( hasattr( emp2, 'test' ) )
```

التنفيذ

```
True
False
True
>>>
```

حذف الخصائص من الكائن

لاحظ! أنه فيما يلي تم حذف الخاصية تماماً بالدالة delattr

الكود

```
class employee:
    pass

emp = employee()
emp.name = 'Ahmed'

print( hasattr( emp, 'name' ) )
delattr( emp, 'name')
print( hasattr( emp, 'name' ) )
```

التنفيذ

```
True
False
>>>
```

كلاس متداخل inner class

لاحظ! أنه فيما يلي تم كلاس وبداخلة كلاسات اخري inline

لاحظ! أنه يمكن استخدام الكلاس الاساسي ويمكن استخدام الكلاسات المتفرعة منه

الكود

```
class computer:
    name = 'pc'
    generation = 5

    class hard:
        name = 'hard'
        capacity = 0
        speed = 0
```

```
    class ram:
        name = 'ram'
        ramtype = 'ram'
        size = 0

r1 = computer.ram()
print(r1.name)

com1 = computer()
print( com1.name )
print( com1.ram.name )
print( com1.hard.name )
```

التنفيذ

```
ram
pc
ram
hard
>>>
```

الوراثة

لاحظ! أنه فيما يلي تمت عملية الوراثة بين الكلاس person والكلاس employee حيث أن الـ employee ورث كل ما في الـ person وذلك بالأقواس بها person بعد تعريف employee الكود

```
class person:
    name = 'Person'
    address = 'Egypt'

    def printdata(self):
```

```
        print( self.name + ' ; ' + self.address )

class employee(person):
    pass

emp1 = employee()
print( emp1.name )
print( emp1.address )
print('==============')
emp1.printdata()
```

التنفيذ

```
Person
Egypt
==============
Person ; Egypt
>>>
```

الوراثة المتعددة

لاحظ! أنه فيما يلي تم توريث الـ person من الـ otherdata ثم توريث الـ employee من الـ person فأصبحت وراثتين

الكود

```
class otherdata:
    email = 'example@domain.com'
    phone = '000000'

class person(otherdata):
    name = 'Person'
    address = 'Egypt'
```

```
    def printdata(self):
        print( self.name + ' ; ' + self.address )

class employee( person ):
    pass

emp1 = employee()
print( emp1.email )
print( emp1.phone )
```

التنفيذ

```
example@domain.com
000000
>>>
```

لاحظ! أنه تم عمل وراثة متعددة داخل اقواس الـ employee

الكود

```
class otherdata:
    email = 'example@domain.com'
    phone = '000000'

class person():
    name = 'Person'
    address = 'Egypt'

    def printdata(self):
        print( self.name + ' ; ' + self.address )

class employee( person , otherdata):
    pass

emp1 = employee()
print( emp1.email )
```

```
print( emp1.phone )
```

التنفيذ

```
example@domain.com
000000
>>>
```

اظهار توثيق الأشياء

لاحظ! أنه فيما يلي تم استخدام __doc__ لإظهار التوثيق

الكود

```
class person:
    '''This is person class
This for Employee or docto or student
'''

class employee:
    'This is employee class'

print( employee.__doc__ )
print( person.__doc__ )

print('=============')

emp = employee()
print( emp.__doc__ )
```

التنفيذ

```
This is employee class
This is person class
This for Employee or docto or student

=============
This is employee class
>>>
```

اظهار قاموس الأشياء

لاحظ! أنه فيما يلي تم معرفة كل ما في أي شيء باستخدام __dict__

الكود

```
class employee:
    'This is employee class'
    name = 'empty'
    def printname(self):
        print( self.name )

print( employee.__dict__ )
print('================')
emp = employee()
emp.city = 'Cairo'
print( emp.__dict__ )
```

التنفيذ

```
{'__module__': '__main__', '__doc__': 'This is employee class', 'na
me': 'empty', 'printname': <function employee.printname at 0x000002
1250C32B70>, '__dict__': <attribute '__dict__' of 'employee' object
s>, '__weakref__': <attribute '__weakref__' of 'employee' objects>}
================
{'city': 'Cairo'}
>>>
```

اظهار اسم الكلاس

الكود

```
class employee: pass
class doctor:   pass
class computer:
    class hard: pass
class student:  pass

print( employee.__name__ )
print( doctor.__name__ )
print( computer.__name__ )
print( computer.hard.__name__ )
print( student.__name__ )
```

التنفيذ

```
employee
doctor
computer
hard
student
>>>
```

اظهار الموديول للكلاس

أولاً انشاء ملف my.py ووضع الكود التالي بداخله

```
class person:
    pass
```

ثانياً تطبيق التالي على ملف التنفيذ الأساسي test.py

لاحظ! أنه فيما يلي تم استخدام __module__ لإظهار الموديول

الكود

```
import my

class employee:
    pass

print( employee.__module__ )
print( my.person.__name__ )
print( my.person.__module__ )
```

التنفيذ

```
__main__
person
my
>>>
```

اظهار الكلاس الأب

لاحظ! أنه فيما يلي تم استخدام __base__ لإظهار الكلاس الذي تمت عملية الوراثة منه

لاحظ! أنه فيما يلي تم التوريث من الـ object بشكل تلقائي

الكود

```
class other:
    pass

class person:
    name = ''
    address = ''
```

```
class employee(person , other):
    pass

class doctor(employee):
    pass

print( employee.__base__ )
print( doctor.__base__ )
print( person.__base__ )
```

التنفيذ

```
<class '__main__.person'>
<class '__main__.employee'>
<class 'object'>
>>>
```

اظهار كل الكلاسات الآباء

لاحظ! أنه فيما يلي تم استخدام getmro من inspect لإظهار قصة حياة الكلاس

الكود

```
class other:pass
class person:pass
class employee(person , other):pass

import inspect

print( inspect.getmro(employee) )
```

التنفيذ

```
(<class '__main__.employee'>, <class '__main_
_.person'>, <class '__main__.other'>, <class
'object'>)
>>>
```

لاحظ! أنه فيما يلي تم استخدام __bases__ لإظهار كل الكلاسات التي تم التوريث منها

الكود

```
class otherdata2:pass

class otherdata:
    email = ''
    phone = ''

class person:
    name = ''
    address = ''

class employee(person , otherdata , otherdata2):
    employeeid = 0

class doctor(employee):pass

print( '---------------' )
print( otherdata.__bases__ )
print( '---------------' )
print( employee.__bases__ )
print( '---------------' )
print( doctor.__bases__ )
print( '---------------' )
```

التنفيذ

```
---------------
(<class 'object'>,)
---------------
(<class '__main__.person'>, <class '__main__.
otherdata'>, <class '__main__.otherdata2'>)
---------------
(<class '__main__.employee'>,)
---------------
>>>
```

إعادة الكتابة على الدوال

لاحظ! أنه فيما يلي تم إعادة الكتابة على الدوال عن طريق كتابة الدالة مرة اخري في الكلاس الذي ورث لتكون مخصصة له

الكود

```
class person:
    def printtype(self):
        print('Person')

class customer(person):
    def printtype(self):
        print('Customer')
    pass

class employee(person):
    def printtype(self):
        print('Employee')
    pass

class doctor(employee):
    def printtype(self):
```

```
        print('Doctor')
    pass

p = person()
c = customer()
e = employee()
d = doctor()

p.printtype()
c.printtype()
e.printtype()
d.printtype()
```

التنفيذ

```
Person
Customer
Employee
Doctor
>>>
```

اظهار اسم كلاس الكائن

لاحظ! أنه فيما يلي تم معرفة اسم الكلاس لأي كائن عن طريق __class__ ثم __name__ بسهولة

الكود

```
class person:
    def printtype(self):
        print(self.__class__.__name__)
class customer(person):pass
class employee(person):pass
class doctor(employee):pass
```

```
p = person()
c = customer()
e = employee()
d = doctor()

p.printtype()
c.printtype()
e.printtype()
d.printtype()
```

التنفيذ

```
person
customer
employee
doctor
>>>
```

كنوز وهدايا

اليك كنوز وهدايا، ووصيتي لك، أن تنشرها في كل مكان من الروابط الأصلية، وأيضاً وصيتي لك بأن تشاهدها وتنصح الجميع بمشاهدتها من اليوتيوب من أكاديمية حسونة، لتنجح هذه الكورسات بفضل الله 😊.

شرح في قصص نجاح

https://www.youtube.com/playlist?list=PLHIfW1KZRIflvUNrnmYsLPbU2t4oG0P-8

شرح في الكمبيوتر بشكل عام

https://www.youtube.com/playlist?list=PLHIfW1KZRIfm_dFhGnawAlnOOE1aSwoSN

شرح عن مقدمة في الحاسبات

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmt3FFalD6y-UjTshUC_zXM

شرح في أنظمة ماك ولينكس

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmTj0gddZTgcPdqDGP1qscF

شرح في الويندوز Windows من الأساس وبالتفصيل

https://www.youtube.com/playlist?list=PLHIfW1KZRIfkKu8K8Dz603sxLa3eQ-g6G

شرح في برنامج وورد Word من الأساس

https://www.youtube.com/playlist?list=PLHIfW1KZRIfk_q1_xWEJ8gl_nbPY_Y75A

شرح في برنامج اكسيل Excel من الأساس

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmeaZUramzmRwJVLQThBiuu

شرح في برنامج الفوتوشوب Photoshop

https://www.youtube.com/playlist?list=PLHIfW1KZRIfntLWyOJQKyfR5RcuXdzfnU

شرح في البرامج الجاهزة

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmCe0esfmzTZTQ_AfoLOH1M

شرح في الخوارزميات وخرائط التدفق + أمثلة

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmOvX3-91zteoSP-b0VS44o

شرح آخر في الخوارزميات وخرائط التدفق

https://www.youtube.com/playlist?list=PLHIfW1KZRIflwIO8IiPxNkSSb-ezkNqDs

شرح في تحليل البيانات
https://www.youtube.com/playlist?list=PLHIfW1KZRIfkDoNSoObO0s_OZn67DFAti

برنامج المبرمجتية
https://www.youtube.com/playlist?list=PLHIfW1KZRIfnLcjFtaI9O5vt1DKdSVM4K

برنامج أساس مبرمج
https://www.youtube.com/playlist?list=PLHIfW1KZRIfkb-fDWH14OjJowQX_Rztgk

شرح في لا اعرف شيء عن الكمبيوتر وأريد أن اكون مبرمج
https://www.youtube.com/playlist?list=PLHIfW1KZRIfnKiRVeW9H2k4TnwXZbaoc8

شرح في منهج الصف الثاني الاعدادي
https://www.youtube.com/playlist?list=PLHIfW1KZRIfmkYyJx1FJcSx1QOJnHanpQ

شرح في منهج الصف الثالث الاعدادي
https://www.youtube.com/playlist?list=PLHIfW1KZRIfmM2mbNfA_4jTI3vyK4iCOE

شرح في كورس قديم عن الويب HTML
https://www.youtube.com/playlist?list=PLHIfW1KZRIfmpHW37aNxMFosXD_p1jQz0

شرح في كورس جديد عن الويب HTML
https://www.youtube.com/playlist?list=PLHIfW1KZRIfnXKjZ9UcT5BVt9x19BlItX

شرح في كورس عن الويب CSS

https://www.youtube.com/playlist?list=PLHIfW1KZRIflO6xnz4jpmzzSuo5B63aJf

شرح في كورس عن الويب JavaScript

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmiYYwt29BG7N9zzsd-jLRQ

شرح في دردشة برمجة عملي

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl0CJzVbN-DHGUJ_Y2DpPim

شرح في الويب جي كويري j-query

https://www.youtube.com/playlist?list=PLHIfW1KZRIfll3ObMFi02Ry7oRU3MJVRG

شرح في الوي بوت ستراب bootstrap

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmzBcYnuLNrmdLxYCDzJgQi

شرح في جاسون JSON

https://www.youtube.com/playlist?list=PLHIfW1KZRIflA5jVQbAAHVizSxoeOAtgq

شرح في انجولار Angular

https://www.youtube.com/playlist?list=PLHIfW1KZRIflbtKBrCp5U2UdbeV1c7F-5

شرح في الويب asp.net وسي شارب ومع الخدمات الأساسية

https://www.youtube.com/playlist?list=PLHIfW1KZRIflkDPHlOLTKV1daEp4pvIgA

شرح في الويب ومع AJAX

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmYd8YFZtSmCzc3hcZXVua9

شرح في قواعد البيانات MySQL

https://www.youtube.com/playlist?list=PLHIfW1KZRIflO4GPp9vMtUgry-xAQKl7i

شرح في الويب ولغة البرمجة PHP

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmPDQahhTDdd1DJ4PwQl3pC

شرح في الويب ومع دبلومة الويب الكاملة

https://www.youtube.com/playlist?list=PLHIfW1KZRIflYXyXHNiG-hyyVaz7-9ujn

شرح في الويب ومع سلسلة تعلم الفيديو الواحد

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnPUohWz2vPjm0obEjzZJD8

شرح في الويب ومع صناعة آلة حاسبة بسيطة

https://www.youtube.com/playlist?list=PLHIfW1KZRIfkYSpySS9Czt3Dxx2rsvqcN

شرح في البرمجة واساسيات السي شارب C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfm8nQAoJF5u2aV43tMRAAmr

شرح في شاشات السي شارب C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnbNoGB0NdoRdllq9fdo6uM

شرح في البرمجة كائنية التوجه OOP وسي شارب C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl6UP-PlUli03pokSc4af2S

شرح في الويب مع سي شارب وC# ASP.NET

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmi8jwSbqiQuVfxeXIBk-bW

شرح في عالم واسع من قواعد البيانات DB مع C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIflAus00vgdVEzLUBCx8ooH

شرح في أمثلة عملية مع قواعد البيانات و C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmYdNXeeaBjt4mxSX5RWjzo

شرح في أسرار قواعد البيانات Database

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnKHFH1uUwdpeW8h-A8lAI

شرح في الـ LINQ مع C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnW4RDln5tzw6htvNhnkr7t

شرح في عمل برنامج المطاعم والأكلات

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnK119pOw2oEDXc0AEpIfd5

شرح في عمل برنامج المبيعات الشامل

https://www.youtube.com/playlist?list=PLHIfW1KZRIfm2vgnbTfoxoUSlxoag0UHX

شرح في دورة السي شارب العملاقة بفضل الله

https://www.youtube.com/playlist?list=PLHIfW1KZRIfkDF2xTIB5kX8gdthmLTufx

شرح في قواعد البيانات الشامل والسي شارب C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmg0C_6ON0IRFWrifni6CTq

شرح في ربط MySQL مع C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfkQ7fgtqD4b4VuhxV6RzNu-

شرح في ربط Oracle مع C#

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl0vnBDg28R0DW9HBkyHvc-

شرح في عمل متصفح بالسي شارب

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl7ToU9EDuRVYiki9C9wxnJ

شرح في كورس سي شارب مختصر

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmIhzWCyXnYlxz8ALlfPRAn

شرح في الويب ومع الدورة التنفيذية في سي شارب وASP.NET

https://www.youtube.com/playlist?list=PLHIfW1KZRIfn6kI3NyvzTDm-wdw9EX7D6

شرح في البرمجة ومع اساسيات فيجوال بيسك VB.NET

https://www.youtube.com/playlist?list=PLHIfW1KZRIflKvIMpDSgFTJsR_IO1dH9i

شرح في شاشات الفيجوال بيسك VB.NET

https://www.youtube.com/playlist?list=PLHIfW1KZRIfn7Pz7MnzABWryuHRP9ab7O

شرح في دورة الفيجوال بيسك العملاقة بإذن الله

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmLGhFsQ1G5cPv1xokzBsbT

شرح في فيجوال بيسك وسكول سيرفر مع التقارير الكاملة

https://www.youtube.com/playlist?list=PLHIfW1KZRIfkkhf3ha77qccAWYMvgM7LR

شرح في ربط Oracle مع VB.NET

https://www.youtube.com/playlist?list=PLHIfW1KZRIflAkiMFHY-0Jc_XXnpLWQO0

شرح في ربط Access مع VB.NET

https://www.youtube.com/playlist?list=PLHIfW1KZRIfkdbHPdTBpcYZzPQtPUeR7I

شرح في الفيجوال بيسك القديم VB 6

https://www.youtube.com/playlist?list=PLHIfW1KZRIfk5L0DQbV8DyUwfoAhQyieg

شرح في البرمجة مع أساسيات لغة الجافا JAVA

https://www.youtube.com/playlist?list=PLHIfW1KZRIflflvzYY7B1d5CJjXpbIriU

شرح في بداية البرمجة الكائنية مع جافا JAVA

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmb25mVAsCJy_Ehgu5rbbQx

شرح في البرمجة الكائنية مع جافا JAVA

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnT2i7Ba4F2nrWb_ENXjyTs

شرح في جافا مع قواعد البيانات

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnKQjOoZc540z3fmGe-slLU

شرح في جافا مع اندرويد Android

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnQ6WRvLbWok4ZZww4k3T1u

شرح في شاشات الجافا

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl2XxZwryDuPO5ZLkRhL-CS

شرح في دورة الجافا العملاقة بفضل الله

https://www.youtube.com/playlist?list=PLHIfW1KZRIfn9BnepQuzWiM_ZPIwUDawL

شرح في أمثلة عن جافا

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnUMvey6UAveK5msn8L_ulS

شرح في التعامل مع ملف الـ jar للجافا

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnDa6squg4OjDjYfx5YC8Zx

شرح في البرمجة مع لغة البرمجة كوتلن Kotlin

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl5UHtM6DRFAVn3PBuJ7NGD

شرح في كلام عن لغة كوتلن Kotlin

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl_KgeClVL_xH_JXFqIsV0Y

شرح في كوتلن مع اندرويد Android

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmbpu6cVCxo98u_Yk0Hfxah

شرح في البرمجة ومع اساسيات البايثون Python

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmm35tXSbxsFDouRJAPP9Y2

شرح في البرمجة الكائنية OOP مع البايثون

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmHvfmFZZ0XuzZYuoosG9kl

شرح في البرمجة مع لغة البايثون Python اساسيات وOOP

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnVNMUcSLFFyDzuLDheF99n

شرح في شاشات البايثون Python

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmMhy8GltKHW5UJ_ZPnGJ0K

شرح في دورة البايثون العملاقة بفضل الله

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnM9y0sQRwjVz2-IwvnEJep

شرح في البرمجة مع لغة الرينج Ring

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl6KzfLziFl650MmThnQ0jT

شرح في البرمجة الكائنية OOP ورينج Ring

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl5TA4nTIIVWsfDyVwIBP6y

شرح في شاشات رينج Ring

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmE8gCtJQpIOGmBz3bc_Nrs

شرح في البرمجة مع لغة السي بلس بلس C++

https://www.youtube.com/playlist?list=PLHIfW1KZRIfkz_N9aNWXRByEJOudCxSRh

شرح متنوع وعام في عالم البرمجة

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl6aI0bGb0zWZyGHc2ctcW7

شرح في قواعد البيانات SQL Server

https://www.youtube.com/playlist?list=PLHIfW1KZRIflvUe-YfDpIqqnU70BopQ3q

شرح في قواعد البيانات ولغة السكول SQL

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmNd2URuqVO5Itboj_LQyrx

شرح في قواعد البيانات مع أكسس Access

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnKpy1msMUPqVYdpu3nOdkY

شرح في قواعد بيانات الوسائط المتعددة Multimedia DB

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmMYvE8mnEH4VT-4v21XV6p

شرح في عمل برنامجك مع الأداة Magic DB

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnkb0hIry8E0LbA0flhaXTo

شرح في برنامج امتحان الطلاب

https://www.youtube.com/playlist?list=PLHIfW1KZRIfnd5v0u8hxwTKXebJK9QpqG

تسالي مبرمجين

https://www.youtube.com/playlist?list=PLHIfW1KZRIfl_zc__duOyEhaCCPZRAuHv

اكاديمية حسونة

https://www.youtube.com/playlist?list=PLHIfW1KZRIfmxKFdzlyAna7sYE_L2hI0o

لقاءات وأحداث Events

شكر خاص

الشكر أولاً وأخيراً وابتداء وانتهاء لله عز وجل، ونشكر كل من:

- شكر لأبي وأمي اللذان ربياني صغيرا.
- شكر لكل أصدقائي واصحابي واحبابي في كل مكان.
- شكر لكل من عرف قدر العلم.
- شكر لكل طالب او طالبة او اخ او صاحب او صديق او حبيب ينشر هذا العمل في كل مكان ليفيد الجميع بإذن الله.
- شكر للأستاذ احمد باريان على دعمه في نشر العلم في كل مكان.
- شكر للمهندس أحمد إبراهيم سالم (Ahmed Ibrahim)
- شكر للدكتور عادل عبد الصبور (Adel Sabour).
- شكر للمهندس محمود سمير فايد (Mahmoud Fayed).
- شكر للمهندس أسامة محمد (Osama Elzero).

- شكر للقائمين على موقع developer.mozilla.org
- شكر للقائمين على موقع wikipedia.org
- شكر للقائمين على موقع w3.org

نشكرك على قراءتك

أولاً: نتقدم لك بخالص الشكر على قراءتك هذا الكتاب، ونتمنى لك الاستفادة الكاملة من محتواه.

ثانياً: لا تنسى أن تسجل لنفسك حساب في أكاديمية حسونة لتنضم إلينا وتشاركنا معنا كل محتويات الموقع، سجل من الرابط التالي:
https://www.hassouna-academy.com/register

ثالثاً: يسر أكاديمية حسونة أن تخبرها برأيك عن هذا الكتاب من خلال تقديم كلمة شكر 😍 من الرابط التالي:

https://www.hassouna-academy.com/thanks

References المراجع

الجدول التالي يحتوي على المراجع التي تم الرجوع إليها أثناء تأليف هذا الكتاب، سواء كانت مراجع تعليمية مرئية أو مقروءة، أو معلوماتية.

المرجع	الرابط
اكاديمية حسونة	https://www.hassouna-academy.com/

الختام The End

والآن، أتوجه الى أحبابي بكلمة من القلب ، خالصة إن شاء الله، فكر معي في اختراعات الإنسان، فكر في تطويرات وصناعات الإنسان، فكر في الإنسان وقدراته، وقل سبحان الله الذي خلق الإنسان، فكر في قول الله عن الإنسان في القرآن الكريم "لَقَدْ خَلَقْنَا الإِنْسَانَ فِي أَحْسَنِ تَقْوِيمٍ"، فكر في قدرة الله الواحد الأحد، الفرد الصمد، الذي لم يلد ولم يولد، ولم يكن له كفؤاً أحد، فكر في الله العظيم، الذي خلق السماوات والأرض، فكر في رب العزة، فكر في رب الكون، وقل، سبحان الله، وقل، سبحان الملك العظيم، وقل سبحان ذي العزة والجبروت، سبحان ذو الملك والملكوت، سبحان الله العظيم، سبحان الله الكريم، سبحان الله، عدد خلقه، ورضا نفسه، وزنة عرشه، ومداد كلماته، وآخر دعوانا "الحمد لله رب العالمين".