- عبدالواحد عبدالكريم الجمالي

- الفهرس :

1-مقدمة البايثون :

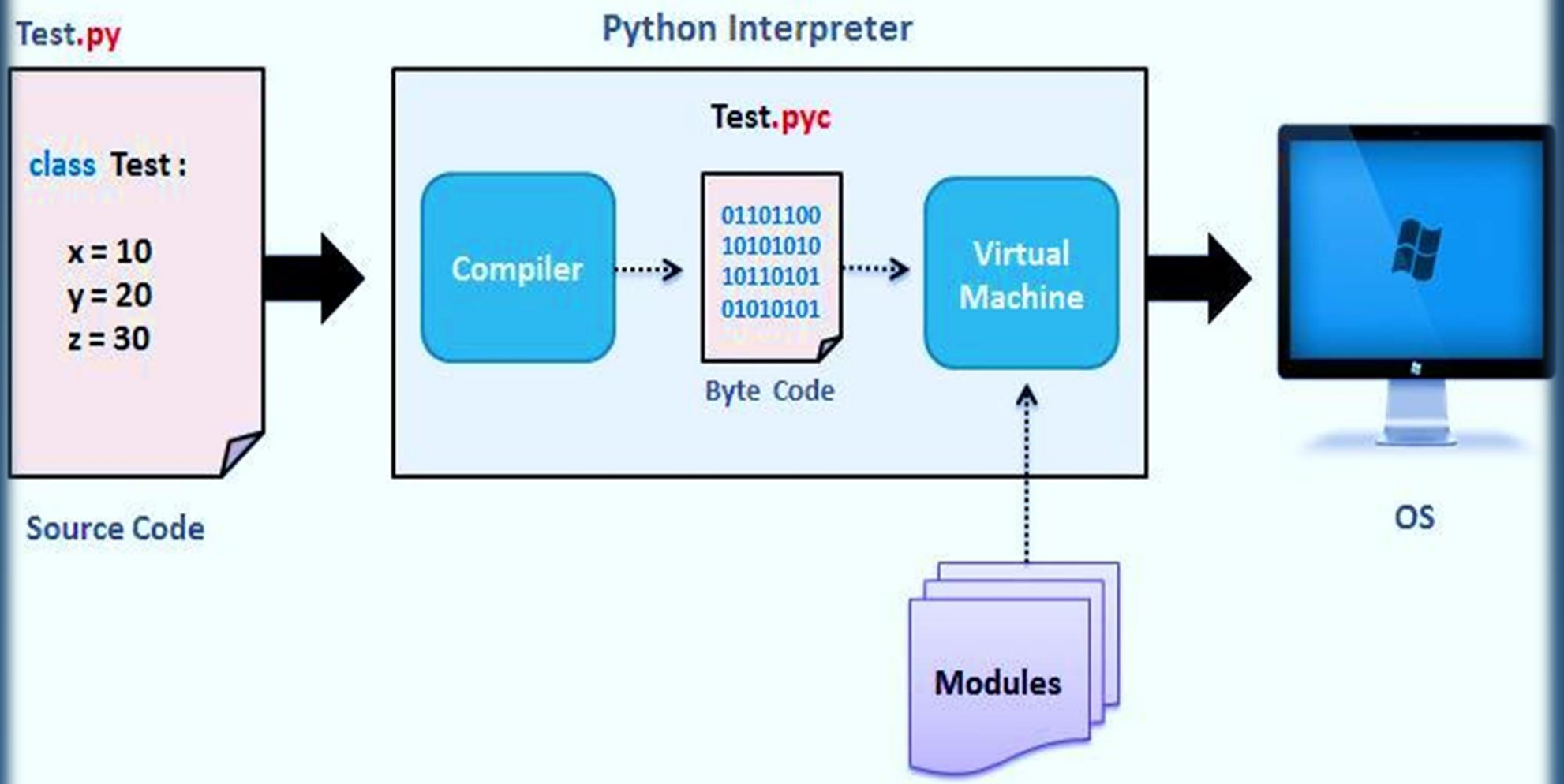
- التعريف : هي لغة برمجية تفسيرية من لغات المستوى العالي، تتميز ببساطة كتابتها وقراءتها، سهلة التعلم، وقابلة للتطوير. تستخدم أسلوب البرمجة الكائنية، مفتوحة المصدر
- , الاستخدام :
- تستخدم في بناء البرامج المستقلة باستخدام الواجهات الرسومية المعروفة
- تستخدم في عمل برامج الويب، بالإضافة إلى استخدامها كلغة برمجة نصية للتحكم في أداء بعض من أشهر البرامج المعروفة أو في بناء برامج ملحقة لها. وبشكل عام
- يمكن استخدام بايثون لبرمجة البرامج البسيطة للمبتدئين، ولإنجاز المشاريع الضخمة كأي لغة برمجية أخرى في نفس الوقت. غالباً ما يُنصح المبتدؤون في ميدان البرمجة بتعلم هذه اللغة لأنها من بين أسرع اللغات البرمجية تعلماً.
- نشأت بايثون
- في مركز العلوم والحاسب الآلي بأمستردام على يد جايدو روسم في أواخر الثمانينات من القرن المنصرم، وكان أول إعلان عنها في عام 1991.

لغات المستوى العالي والمستوى الأدنى

من الاحداث الاتية	اللغات منخفضة المستوى	اللغات عالية المستوى
قابلية القراءة	صيغة قابلة للقراءة آليًا للبرنامج	صيغة قابلة للقرة من قبل البشر
الكتابة	من الصعب كتابة اكواد اللغة منخفضة المستوى وتجميعها	من السهل كتابة اللغات ذات المستوى العالي بالإضافة إلى تجميعها.
المساحة في الذاكرة	اللغة منخفضة المستوى مضغوطة وتتطلب مساحة أقل للذاكرة	يتطلب مساحة كبيرة للذاكرة.
اكتشاف الأخطاء	صعوبة في تصحيح الأخطاء	سهولة اكتشاف الأخطاء وتصحيحها
الامثلة	لغات الالة	C ,python,

python™

تنفيذ البرنامج :



المميزات :

العمل على أكثر من منصة

البرنامج الذي تبنيه بواسطة لغة بايثون يعمل على جميع أنظمة التشغيل

كائينة التوجه

تدعم مفهوم الكلاس, الكائن, التغليف, الوراثة

تعدد المهام

بايثون توفر لك تقنية الـ Multithreadingو التي تسمح لك بجعل برنامجك قادراً على تنفيذ عدة أوامر مع بعض و بنفس الوقت.

قواعد البيانات

بايثون توفر إنترفيسات جاهزة للتعامل مع أهم قواعد البيانات.

واجهة المستخدم

يمكن بناء تطبيقات تطبيقات فيها واجهة مستخدم فيها.

اسلوب كتابة الكود في بايثون :

- **Case Sensitivity**
- تعني أن لغة البرمجة تميز بين الأحرف الكبيرة و الأحرف الصغيرة.
- بايثون تعامل الأسماء التي نستخدمها بتأني سواء للمتغيرات, الدوال, الكلاسات, الكائنات إلخ.
  مثال: noteو Noteليسوا شيئاً واحداً.

```
Note    Note
Print   print
```

- إسم الكلاس

```
class First:
class FirstPythonClass:
```

اسلوب كتابة الكود في بايثون :

- اسم المتغير :
- استخدم الأحرف الصغيرة عند وضع أسماء للمتغيرات و في حال كان إسم المتغير يتألف من أكثر من كلمة قم بوضع _ بين كل كلمتين

```
average = 10
total_score = 20
```

- إسم الدالة :
- إستخدم الأحرف الصغيرة عند وضع أسماء للدوال و في حال كان إسم الدالة يتألف من أكثر من كلمة قم بوضع _ بين كل كلمتين.

```
def display():
def display_user_info():
```

اسلوب كتابة الكود في بايثون :

- الكلمات المحجوزة في بايثون :
- جميع الكلمات التالية محجوزة للغة بايثون, أي لا يمكن إستخدامها كـ **Identifiers**.

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

اسلوب كتابة الكود في بايثون :

- قواعد ترتيب الكود في بايثون :
- لا تقم بإضافة أي مسافة فارغة باستخدام الزر TAB لأن المسافة التي يعطيها هذا الزر غير مسموح إستخدامها في لغة بايثون.
- إستخدم **4** مسافات فارغة Space عند وضع الكود بشكل متداخل.
- ضع سطر فارغ على الأقل بين السطر الذي تم فيه تعريف الكلاس و الدوال المعرفة بداخله.
- ضع سطر فارغ على الأقل بين كل دالتين.
- ضع سطر فارغ بين كل إثنين بلوك تضيفهما بداخل الدوال.
- ضع مسافة فارغة حول جمل التحكم و جمل الشرط.
- عند وضع التعليقات يفضل إستخدام الرمز # في بداية كل سطر حتى و إن كان التعليق يتألف من عدة أسطر.
- عدد الأحرف القصوى التي يمكن وضعها في كل سطر هو **79** حرف.

انواع البيانات :

- 1-القيم المنطقية (boolean) :
- القيمتان المنطقيتان (Boolean)التي تدعمهما بايثون هما True و False وهما كائنان ثابتا يعبران عن صحّة تعبير ما، فإمّا أن يكون صحيحًا True أو خطأً .False وتسلك القيمتان False و True سلوك القيمتين 0 و 1 على التوالي في معظم السياقات تقريبًا، ويستثنى تحويل القيم المنطقية إلى سلاسل نصية

```
>>> foo = True
>>> bar = False11
```

انواع البيانات :

2-الاعداد

- هو أي عدد موجب أو سالب لا يتضمن فاصلة عشرية، ويمكن تمثيله بالنظام العشري والست عشري والثماني والثنائي
- يجب أن يكون العدد الصحيح مسبوقًا بالقيمة 0o لاستخدامه في النظام الثماني، وبالقيمة 0x لاستخدامه في النظام الست عشري، وبالقيمة 0b لاستخدامه في النظام الثنائي، وفيما يلي مجموعة من الأمثلة

```
>>> q = 3571       # عدد صحيح في النظام العشري
>>> q = -424       # عدد صحيح سالب
>>> q = 0o675      # في النظام الثماني
>>> q = 0xAB23     # في النظام الست عشري
>>> q = 0b11001010 # في النظام الثنائي
```

انواع البيانات :

- 1-2 الدالة (int) :
- دالة تقوم بتحويل الاعداد العشرية الى اعداد صحيحة :

```
>>> x = 3.9
>>> int(x)
3
```

3- الاعداد العشرية (float) :

- هي الأعداد التي تتضمن فاصلة عشرية أو علامة أسية :

```
>>> x = 2.5
>>> y = -1.609
```

انواع البيانات :

- 4-السلاسل (String):
- هي متغير يحتوي على عدد من الاحرف وهي غير قابلة للتعديل
- يوجد طرق مختلفة للتعبير عنها

- سلسلة بعلامة الاقتباس منفردة

```
X='------'
```

سلسلة بعلامة اقتباس مزدوجة

```
X="------"
```

سلسلة من عدة اسطر بثلاث علامة اقتباس

```
X='''------'''
```

4 – السلاسل (STRING):

- الرمز (\)
- يستخدم لتهريب الاحرف من السلسلة حيث اذا اتا قبل اي حرف يتم اللغاة من السلسلة

```
x='any \ st\ring'
>>> print x
```

- اختصار (r)
- هذة هو اختصار raw يستخدم من اجل اللغا عملية تهريب الحروف وذلك بكتابة قبل علامة الاقتباس

```
>>> print('C:\some\name')
C:\some
ame
 >>> print(r'C:\some\name')
C:\some\name
```

4 – السلاسل (STRING):

- ربط السلاسل النصية وتكرارها:
- يمكن ربط السلاسل النصية بعضها ببعض بواسطة العامل +، ويمكن تكرار السلاسل باستخدام العامل

```
>>> 3 * 'un' + 'ium'
 'ium''ununununium'
```

- يمكنا ربط السلسل بالمتغيرات كما في المثال التالي :

```
>>> x = 'Py‘
>>> x +='thon'
 >>>print x
```

4 – السلاسل (STRING):

4-1 فهرسة السلاسل النصية

- يمكن فهرسة السلاسل النصية في بايثون، ويحمل الحرف الأول رقم الفهرس0.
- السلاسل النصية في بايثون غير قابلة للتعديل لذا فإنّ إسناد قيمة إلى موقع مفهرس ضمن السلسلة سيؤدي إلى حدوث خطأ :

```
>>> s[0] = 'J '          TypeError: 'str' object does not support item assignment
```

```
s="any    string"
```

الاندكس(+)	0	1	2	3	4	5	6	7	8	9
السلسلة	a	n	y		s	t	r	i	n	g
الاندكس (-)	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

4 – السلاسل (STRING):

2-4 قطع السلسلة

- تستخدم الفهرسة للحصول على حرف واحد اما التقطيع نحصل فيه على جز من السلسلة.
- خوارزمية القطع :

```
<string> [START : STOP]
```

4 –2 القطع (SUB):

- 1- الرقم الأول يكون مشمولًا دائمًا بعملية القطع أما الرقم الأخير فلا يكون مشمولًا
- 2- هناك بعض القيم الافتراضية المفيدة في خاصية التقطيع، فإذا حذف الرقم الأول فإنّ القيمة الافتراضية له ستكون صفرًا، وإذا حُذف الرقم الثاني فإنّ القيمة الافتراضية تكون طول السلسلة النصية التي سيجري اقتطاعها.

```
>> s='any string ‘
>>> s[ : 4]+s[ 4:  ]
```

- 3-قبل التفكير في عملية القطع يجب عليك معرفة الفهرس لكل حرف
- يكون طول جملة القط عند استخدام الفهارس غير السالبة هو الفرق بين الرقمين إن كانا ضمن الحدود، فعلى سبيل المثال S=[1:3]

الاندكس(+)	0	1	2	3	4	5	6	7	8	9
السلسلة	a	n	y		s	t	r	i	n	g
الاندكس (-)	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
>> s='any  string ‘
>>>print s[0 :  2] =an
  print   s[1 : 5]=
   print   s[ : 4]=
  print  s[5,-1]=
```

- 4- 3الدالة(len)
- هي دالة تقوم بأيجاد طول السلسلة وتبدا العد من واحد والفراغ يدخل في العد .

```
>>> s = 'any    string‘
>>> len(s)
```

- 4-4الدالة (str) :
- لتحويل الكائنات الى سلاسل نصية و تعيد سلسلة فارغة في حال عدم تقديم أي كائن لها ن كان الكائن سلسلة نصية فإنّ الدالة تعيد السلسلة النصّية ذاتها.

```
x = [1, 2, 3]
print str(x)
‘[1, 2, 3]‘
```

- 4- 5الدالة capitalize ()
- هذة دالة تقوم بتحويل اول حرف من السلسلة الى حرف كابتل

```
x="ali"
print x.capitalize()
Ali
```

- الدالة find ( )
- دالة تقوم بأيجاد اندكس السلسلة الفرعية المراد البحث عنها داخل السلسلة الاصلية
- خوارزمية (find):
- <STRING >.find ('text' ,star, stop)
- تستخدم هذة الدالة لايجاد موقع الكلمة فقط .

الاندكس(+)	0	1	2	3	4	5	6	7	8	9
السلسلة	a	n	y		s	t	r	i	n	g
الاندكس (-)	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
>>> string.find("a")
0
>>> string.find("St")
-1
>>> string.find("y", 0, 2)
>>> string.find("mr")
>>> string.find("m", 1)
-1
```

- 6-4الدالة split ()
- تقسم السلسلة الاصلية الى قائمة من السلاسل النصية بالاعتماد على فاصل محدد

```
s="ali ahmed"
print s.split()
['ali', 'ahmed']
```

- 7-4 الدالة join()
- دالة تقوم بربط عدة قوائم في سلسلة واحد

```
s=['ali','ahmed','salah']
print','.join(s)
aliahmedsalah
```

- 4-8الدالة raplace ()
- تقوم بأستبدال النص القديم الموجود في السلسلة بنص جديد
- الخوارزمية

```
<string>.rplace[old, new,cont]
```

- اسم السلسلة =String
  السلسلة النصية الفرعية المراد استبدالها = Old
- السلسلة النصية الفرعية الجديدة =New
- عدد مرات التي نريد فيها استبدال الكلمة =Count

5- القوائم LIST

تمتلك بايثون عددًا من أنواع البيانات المركبة والتي تستخدم لتجميع القيم الأخرى مع بعضها البعض، والقوائم هي أوسع هذه الأنواع وأكثرها شمولًا، ويمكن كتابتها كقائمة من القيم (العناصر) المفصولة عن بعضها البعض بفواصل (،) ومحاطة بأقواس مربعة. يمكن للقوائم أن تتضمّن أنواعًا مختلفة، ولكن عادة ما تكون العناصر كلها من النوع نفسه.

```
>>> squares = [1, 4, 9, 16, 25]
>>> square
[1, 4, 9, 16, 25]
```

5-1 فهرسة القوائم :

كما هو الحال مع السلاسل النصية

ولكن على العكس من السلاسل النصية فإنّ القوائم قابلة للتعديل ((mutable، أي يمكن تعديل محتوياتها حسب الحاجة:

يمكن أيضًا إضافة عناصر جديدة إلى نهاية القائمة، وذلك بواسطة التابع append()

```
>>> cubes = [1, 8, 27, 65, 125]
>>> cubes.append(216)
```

list.insert()

يضيف التابع عنصرًا إلى القائمة في الموقع الذي يحدّده المستخدم.

البنية العامة l.insert()


```
>>> cubes = [1, 8, 27, 65, 125]
>>> cubes.append(216)
```

list.pop()

يحذف هذا التابع العنصر في الموقع المحدّد من قبل المستخدم
ويحذف العنصر الأخير اذا لم يحدد الموقع .

```
>>> fruits = ['orange', 'apple', 'pear', 'banana', 'kiwi', 'apple', 'banana']
>>> fruits.pop(2)
'pear'
>>> fruits.pop()
'banana' >>>
```

Tuples

يتكوّن الصفّ من عدد من القيم المفصولة عن بعضها بفاصلة، وتنتهي ( )
لا يمكنك تعديل قيمة اي عنصر وهذا مايميز القائمة علية :

```
>>> t = (12345, 54321, 'hello!‘)
```

جمل الإرجاع والطباعة :

جملة الارجاع :

ترجع قيمة وليس طباعتها

جملة الطباعة :

تطبع قيمةا

جملة الإرجاع قبل جملة الطباعة

اذا اتت جملة الإرجاع قبل جملةالطباعة يتم تجاهل جملة الطباعة .

جملة الطباعة قبل جملة الإرجاع

يتم الطباعةوالإرجاع

المدخلات والمخرجات :

المدخلات :

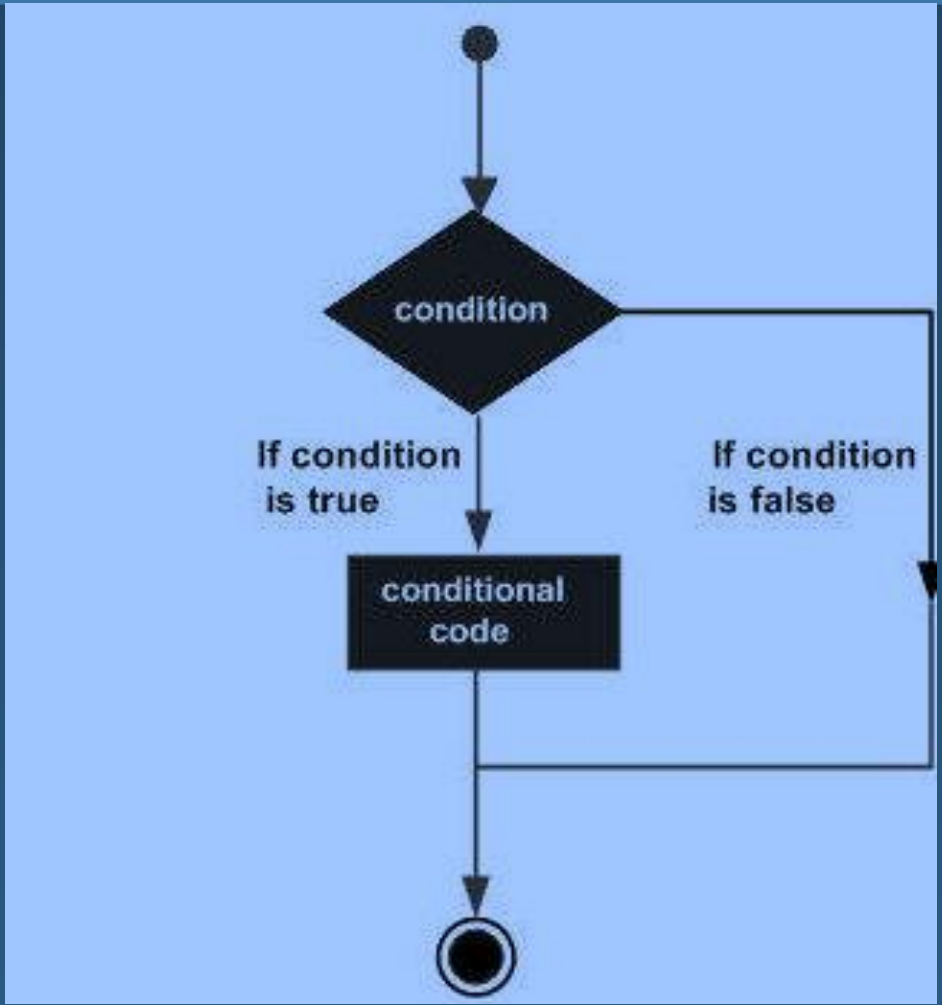
هي القيم التي نرسلها عبر استدعاء الدالة .
المتغير داخل الدالة الذي يستقبل القيم يعتبر مدخل .

المخرجات :

هي نتائج القيم المدخلة وتتمثل في جمل الطباعة والإرجاع فقط.

python™

جمل التحكم :




python™