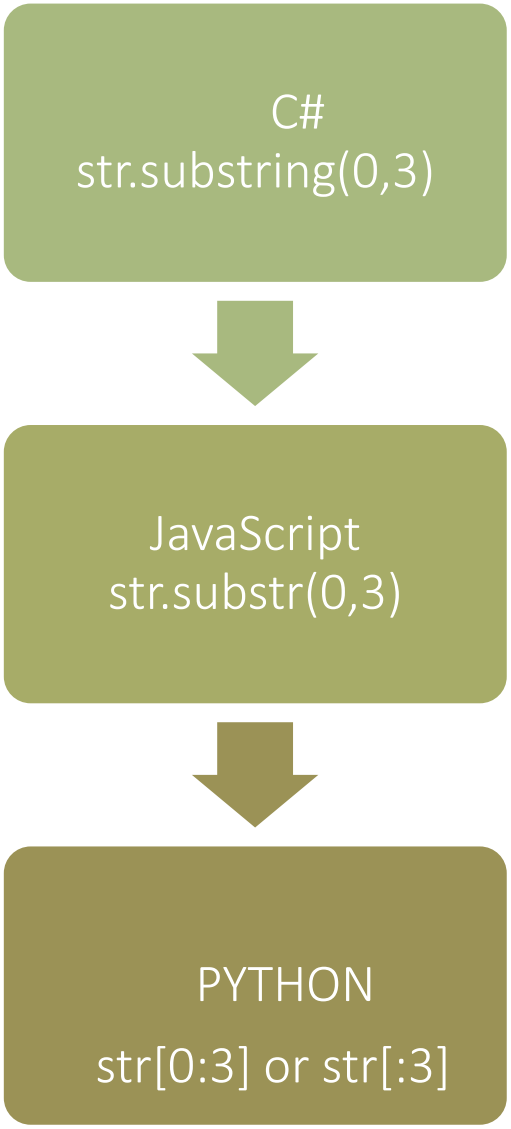
محاور المحاضرة

- مقدمه عن لغة البايثون
- اختراعها وتطورها
- الصيغه النحويه للغة البايثون
- مجالات استخدام لغة البايثون
- مميزات وعيوب لغة بايثون


C#
str.substring(0,3)
JavaScript
str.substr(0,3)
PYTHON
str[0:3] or str[:3]

مقدمة عن لغة البايثون

تعتبر لغة بايثون لغة تفسيرية متعددة الإغراض وتستخدم بشكل واسع في العديد من المجالات كبناء البرامج المستقلة باستخدام الواجهات الرسومية المعروفة وفي عمل برامج الويب، باللإضافة 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 وإنجاز المشاريع الضخمة كأي لغة برمجية أخرى في نفس الوقت.

اختراعها وتطورها

نشأت بايثون في مركز (CWI) مركز العلوم والحاسب الاليء بأمستردام على يد جويدو فان رُزوم في أواخر الثمانينات،

وكان أول إعلان عنها في عام ١٩٩١.

تم كتابة نواة اللغة بلغة C.

أطلق اسم "بايثون" على لغته تعبيرا عن إعجابه بفِرقَة مسرحية هزلية شهيرة من بريطانيا، كانت تطلق على نفسها اسم مونتي بايثون بالإنجليزية: (Python Monty)

الصيغة النحوية للغة بايثون

تستخدم بايثون المسافات البيضاء والإزاحات بدلا من الأقواس وذلك لكي يتم تحديد حجم الجمل البرمجيه.

لغة بايثون أبتكرت، وطورت لتكون لغة عالية المقروئيه فهي تستخدم كلمات ومفاهيم إنجليزية شائعة الإستخدام بينما تستخدم اللغات البرمجية الإخرى علامات الترقيم.

عالم البيانات وهو بإختصار التعامل مع البيانات الضخمة المعقدة وعلى دراية بـ قواعد الرياضيات الصعبة، ولغات البرمجة التي من شأنها التعامل مع البيانات الضخمة، وهو بإختصار منجم ذهب لكل من يعمل في ذلك المجال لإنهم يستخرجون العديد من المعلومات الهامة من مليارات البيانات التي قد لا تبدو مهمة وعالم البيانات في تلك المهمة لانك تستطيع أن تستخدمها في الحصول على راتب عالي ومميز، ولغة بايثون تلعب دوراً رائعاً للتعامل مع البيانات الكبيرة.

مجالات استخدام لغة البايثون

١. تستخدم لغة البرمجة بايثون جوجل وناسا واليوتيوب والأنستقرام وبرمج COM..

٢. قد استخدمت في برمجة مشروع ZOPE.

٣. تستخدم في عدة تطبيقات، ومشاريع عالمية كمشروع BLENDER، وهو مشروع تصميم 3D مثل Softimage للتصاميم 3D، وتطبيق مايا لعمل تصاميم 3D .

٤. استخدمت في مشروع FIERFOX .

٥. تستخدم في أنظمة تشغيل مثل (.MAC-Linux – windows).

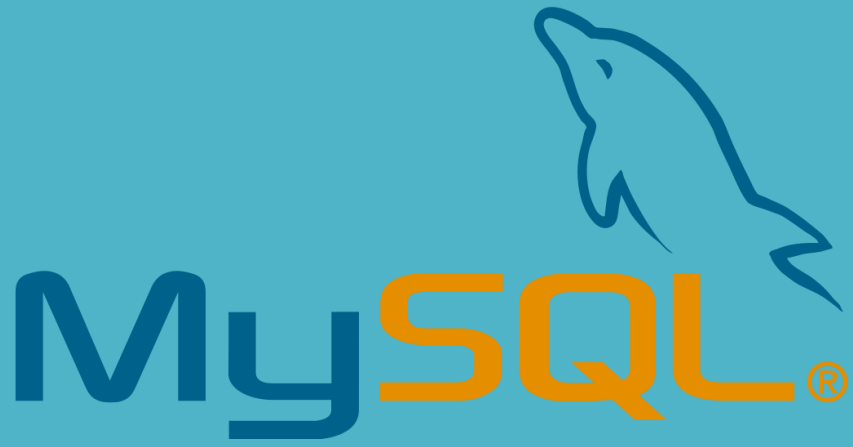
٦. تستخدم كلغة برمجة نصية.

٧. لعبه Civilization .

٨. تستخدم في عمل سكربتات لعدة ألعاب الكترونية شهيرة، مثل online، Second Life.

٩. برامج تصميم الخرائط الجغرافية كبرنامج . ArcGIS

١٠. تُستخدم لغة بايثون في مجال الحماية الرقمي (Security) .



تتميز بايثون بمجتمعها النشط ، كما أن لها الكثير من المكتبات البرمجية ذات الاغراض الخاصة والتي برمجها أشخاص من مجتمع هذه اللغة، مثالً مكتبة PyGame التي توفر مجموعه من الوظائف من أجل برمجة الالعاب ويمكن للبايثون التعامل مع العديد من أنواع قواعد البيانات مثل MySQL وغيره.

مميزات لغة البايثون

١ـ يمكن إستخدامها على العديد من أنظمة تشغيل الحاسب الاليء مثل (.MAC-Linux – windows)

٢ـ سهولة تعلم لغة بايثون وسهولة و قرائتها.

٣ـ لغة البايثون مفتوحة المصدر (Source Open).

٤ـلغة بايثون تدعم البرمجة الكائنية Programming Oriented Object وتدعم أيضا البرمجة الإجرائية

٥ـالتعامل مع قواعد البيانات المختلفة مثل: – MySql Oracle- Sql Microsoft كما يمكن إستخدام الـ API للتعامل مع DATA.

٦ـ لغة بايثون عالية المستوي فلا تحتاج لمراجعة التفاصيل.

عيوب لغة البايثون

الحالات الحساسه

الحالات الحساسه تعني أن لغة البرمجة تميز بين الأحرف الكبيرة و الأحرف الصغيرة.

بايثون تعامل الأسماء التي نستخدمها بتأني سواء كنا نضع هذه الأسماء للمتغيرات، الدوال، الكلاسات، الكائنات إلخ.

مثال: note و Note ليسوا شيئاً واحداً.

إسم الكلاس

دائماً إبدأ إسم الكلاس بحرف كبير و في حال كان إسم الكلاس يتألف من أكثر من كلمة، إجعل أول حرف من كل كلمة كبيراً.

أمثلة:

- في حال كان إسم الكلاس يتألف من كلمة واحدة:

class First:

- في حال كان إسم الكلاس يتألف من أكثر من كلمة:

class FirstPythonClass:

إسم المتغير

إستخدم الأحرف الصغيرة عند وضع أسماء للمتغيرات و في حال كان إسم المتغير يتألف من أكثر من كلمة قم بوضع _ بين كل كلمتين.

أمثلة

- في حال كان إسم المتغير يتألف من كلمة واحدة:

average = 10

- في حال كان إسم المتغير يتألف من أكثر من كلمة:

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.

أمثلة

- في حال كان إسم الدالة يتألف من كلمة واحدة.

```
def display():
```

- في حال كان إسم الدالة يتألف من أكثر من كلمة.

```
def display_user_info():
```

التعليقات

نستخدم التعليقات لنضع ملاحظات حول الكود الذي كتبناه فقط لكي لا ننسى كيف برمجنا الكود في حال أردنا مراجعته أو التعديل عليه بعد وقت طويل.

التعليقات لا تؤثر إطلاقاً على الكود المكتوب الموضوع في البرنامج و يمكن وضع عدد غير محدود من التعليقات.

لتضع تعليق، ضع الرمز # ثم أكتب بعده ما شئت.

مثال:

```
#    هذا تعليق يتألف من سطر واحد و هو لا يؤثر أبداً على الكود الموضوع
#  هذا تعليق آخر.. كما تلاحظ، يمكنك وضع العدد الذي تريده من التعليقات
```

كتابة أكثر من أمر واحد على نفس السطر

إفتراضياً، بايثون تعتبر أن كل أمر يكتب على سطر واحد.

إذا أردت كتابة أكثر من أمر على نفس السطر قم بوضع فاصلة منقوطة ؛ بين كل أمرين و هكذا سيفهم مترجم لغة بايثون أن السطر عليه أكثر من أمر.

مثال:

- هنا قمنا بوضع ثلاث أوامر على سطر واحد. فعلياً، كل أمر هنا عبارة عن تعريف متغير و إعطائه قيمة:

```
x = 1; y = 2; z = 3
```

كتابة أمر واحد على أكثر من سطر

إذا أردت كتابة أمر واحد على أكثر من سطر قم بوضع الرمز \ في نهاية كل سطر و هكذا سيفهم مترجم لغة بايثون أن الأمر يتألف من أكثر من سطر.

هنا قمنا بتعريف ثلاث متغيرات:

```
item_1 = 10
item_2 = 20
item_3 = 30
```

الثلاث أسطر التالية عبارة عن أمر واحد:

وضع الناتج في المتغير item_3 و item_2 و item_1 إذا هنا سيتم جمع قيم المتغيرات

```
total = item_1 + \
        item_2 + \
        item_3
```

هنا قمنا بعرض قيمة المتغير:

```
print("total contains:", total)
```

سنحصل على النتيجة التالية عند التشغيل:

```
total contains: 60
```

المثال الثاني

- هنا قمنا بتعريف مصفوفة من النصوص.

الأربعة أسطر التالية عبارة عن أمر واحد:

```
seasons = ['Autumn',
        'Winter',
        'Spring',
        'Summer']
```

هنا قمنا بعرض القيم المخزنة في المصفوفة:

```
print("Seasons contains:", seasons)
```

- سنحصل على النتيجة التالية عند التشغيل:

```
Seasons contains: ['Autumn', 'Winter', 'Spring', 'Summer']
```

الأحرف المستخدمة في وضع الأسماء في بايثون

أي إسم نضعه لمتغير، دالة، كلاس، كائن إلخ.. يسمى identifierفي البرمجة.

في بايثون كل عنصر نريد إنشاؤه علينا إعطاؤه إسم خاص، أي علينا تحديد الـ identifier له.

إذاً يتم التمييز بين العناصر في بايثون من خلال أسمائهم، أي من خلال الـ Identifiers.

قواعد إلزامية عند إعطاء الأسماء

١. الـ Identifiers يجب أن يبدأوا بحرف كبير بين A-Z أو حرف صغير بين a-z أو الشحطة _.
٢. يمنع إستخدام أي كلمة من الكلمات المحجوزة (Keywords)
٣. لا تنسى أن بايثون تطبق مبدأ الـ Case Sensitive.

ركز على أول حرف فقط :

أمثلة للأسماء التي يسمح باستخدامها: Harmash , name , _number

أمثلة للأسماء التي يمنع استخدامها: 1st , -cash , @user

الكلمات المحجوزه في بايثون

جميع الكلمات التالية محجوزة للغة بايثون، أي لا يمكن إستخدامها كـ Identifiers

exec	and
False	assert
finally	break
for	class
fromglobal	continue
if	def
import	del
in	elif
islambda	else
None	except
nonlocal	try
not	while
orpass	with
print	yield
raise	
return	
True	

PYTHON

A Highly Expressive
Programming Language..

THE END