البرمجة و الذكاء الإصطناعي 2

السنة 1 ماستر - علم الإجتماع

جامعة سطيف 2

الأستاذة: د.حياة دحمري

جوان 2026

قائمة المحتويات

I أساسيات البرمجة في Python

آ. *المتغيرات* Variables

المتغير هو مكان في الذاكرة يُستخدم لتخزين قيمة يمكن الرجوع إليها واستخدامها لاحقًا في البرنامج.
المتغير = اسم + قيمة[1]

1. قواعد تسمية المتغيرات:

يجب أن:
- يبدأ بحرف أو _
- يحتوي على حروف، أرقام، أو _

لا يجب أن:
- يبدأ برقم
- يحتوي على مسافات
- يكون كلمة محجوزة مثل : *if, for, while*

ملاحظة: المتغيرات حساسة لحالة الأحرف (كبيرة أو صغيرة)، مثلا *name ≠ Name*

2. أنواع المتغيرات في بايثون:

- الأعداد الصحيحة *integer* و يرمز لها بـ *int*. مثال: *x = 10*
- الأعداد العشرية *float* و يرمز لها بـ *float*. مثال: *y = 3.14*
- النصوص *strings* و يرمز لها بـ *str*. مثال: *name = "Python"*
- القيم المنطقية *booleans* و يرمز لها بـ *boll*، و تحمل قيمتين فقط صحيح أو خطأ. مثال: *is_student = True*

الأنواع المركبة (**Collections**)
- القوائم (*list*) : قابلة للتغيير ويمكن تعديل عناصرها. مثال: numbers=[1,2,3]
- الصفوف (*tuple*): غير قابلة للتغيير ولا يمكن تعديل عناصرها. مثال: date = (30,10,2021)
- القواميس (*dict*): قابلة للتغيير، تخزن البيانات على شكل مفتاح وقيمة. مثال:
- student = {"name": "Ahmed", "age": 20, "grade": 15.5}

ملاحظة: بايثون لغة ديناميكية (لا تحتاج تحديد النوع مسبقًا)

3. العمليات على المتغيرات

1- معرفة نوع المتغير: لمعرفة نوع المتغير نستعمل الدالة *type*()، مثال:

```
x = 10
print(type(x))     # <class 'int'>

name = "Ali"
print(type(name))  # <class 'str'>
x = 10
print(type(x))     # <class 'int'>

name = "Ali"
print(type(name))    # <class 'str'>

```

2- تحويل نوع المتغير: لتحويل متغير إلى نوع آخر، نكتب النوع الذي نريد التحويل اليه ثم داخل قوسين نكتب اسم المتغير. مثال:

```
x = "10"
print(type(x))     #<class 'str'>
y = int(x)
print(type(x))    # <class 'int'>
```

3- العمليات الحسابية: الجمع (+)، الطرح(-)، الضرب (*)، القسمة (/)، باقي القسمة (%)، الأس (**). مثال:

```
a = 10
b = 5
s = a + b     # s=15
d = a / b     #d=2
```

4- العمليات على النصوص: دمج نصين (+)، تكرار نص (*). مثال:

```
name1 = "Ali"
name2 = "Ahmed"

print(name1 + " " + name2)    # Ali Ahmed
print("Hi " * 3)     # Hi Hi Hi

```

5- الإسناد المتعدد: يمكن انشاء عدة متغيرات في سطر واحد. مثال:

```
a, b, c = 1, 2, 3
x = y = z = 0
```

4. إدخال البيانات من المستخدم

لجعل قيمة متغير تدخل بواسطة المستخدم، نستعمل التعليمة ***input***. مثال:

```
name = input("ادخل اسمك: "(
age = int(input("ادخل عمرك: "((
print("مرحبا ", name)
print("عمرك: ", age)
```

ب. *الشروط* Conditions

تُستخدم الشروط لاتخاذ قرار بناءً على حالة معينة.[2]

1. if

إذا تحقق الشرط، يتم تنفيذ التعليمات.

```
if condition :
   instructions
```

مثال:

```
X= int(input("ادخل رقم"))
if x>0 :
   print("عدد موجب")
```

2. if - else

تستخدم لتنفيذ أحد خيارين.

```
if condition :
   instructions 1
else:
    instructions 2
```

مثال:

```
X= int(input("ادخل رقم"))
if x>0 :
   print("عدد موجب")
else :
   print("عدد سالب")
```

3. if - elif - else

تستخدم عند وجود عدة شروط.

```
if condition 1:
    instructions 1
elif condition 2:
   instructions 2
else :
    instructions 3
```

مثال:

```
X= int(input("ادخل رقم"))
if x>0 :
   print("عدد موجب")
elif x<0 :
   print("عدد سالب")
else :
   print("عدد معدوم")
```

4. العمليات المنطقية

تستخدم عندما يكون الشرط مركب.
1- ***and***: كل الشروط صحيحة. مثال:

```
if moyenne_sem1 >=10 and moyenne_sem2 >=10 :
        print("ناجح")
```

2- ***or***: شرط واحد صحيح. مثال:

```
if is_present or has_excuse:
     print("لا يحتسب الغياب")
```

3- ***not***: عكس النتيجة. مثال:

```
if not moyenne >= 10:
 print("راسب")
```

ب. *الحلقات* Loops

تُستخدم لتكرار تنفيذ مجموعة من التعليمات.[3]

1. for

تُستخدم عند معرفة عدد التكرارات.

```
for i in range(number_of_iterations):
   instructions
```

مثال:

```
for i in range(5):
   print(i)

#result: 0 1 2 3 4
```

الدالة () ***range*** تطلق سلسلة من الأرقام، مثلا:
range(5) : من 0 إلى 4. (0 1 2 3 4)
range(2,6) : من 2 إلى 5. (2 3 4 5)
range(0,10,2) : من 0 إلى 8 ب 2 خطوة. (0 2 4 6 8)

2. while

تُستخدم عندما يعتمد التكرار على شرط.

```
while condition :
   instructions
```

مثال:

```
X=0
while x<5 :
   print(x)
   x=x+1
#result: 0 1 2 3 4
```

ا) قواعد مهمة للحلقات

- يجب التأكد من أن الحلقات تنتهي (خاصة *while*).

- يجب زيادة أو نقصان المتغير داخل حلقة *while*.
- يمكن استخدام *break* للخروج من الحلقة.
- يمكن استخدام *continue* لتخطي خطوة.

قائمة المراجع

[1] أحمد منصور. (2020). مدخل إلى البرمجة بلغة بايثون للمبتدئين. دار شعاع، دمشق.

[2] هديل طاهر. تعلم بايثون للمبتدئين. archive.org

مراجع الأنترنيت

https://harmash.com/tutorials/python/overview [3]