خطوة

أساسيات بايثون

خطوتك الأولى في البرمجة: من تسطيب بايثون لحد ما تكتب برنامج بياخد قرارات.

في الملخص ده

١ يعني إيه بايثون؟

٢ التسطيب وأول برنامج

٣ المتغيرات

٤ أنواع البيانات

٥ العمليات الحسابية والمقارنة

٦ التعامل مع النصوص

٧ إدخال البيانات وتحويل الأنواع

٨ الشروط: if و elif و else

٩ أخطاء كل المبتدئين بيقعوا فيها

١٠ تمارين وورقة مراجعة

محمود منصور

المستوى: مبتدئ

khatwa.cloud

الخطوة ١

يعني إيه بايثون؟

بايثون لغة برمجة بتكتب بيها أوامر للكمبيوتر بكلام قريب من الإنجليزي، وعشان كده هي من أسهل اللغات اللي ممكن تبدأ بيها.

البرنامج ببساطة هو مجموعة أوامر مكتوبة بالترتيب، والكمبيوتر بينفّذها سطر سطر من فوق لتحت. بايثون بتاخد الأوامر دي وبتنفّذها على طول، من غير ما تحتاج خطوات معقدة قبل التشغيل.

ليه نبدأ بيها؟

- **شكلها بسيط:** مفيش أقواس كتير ولا فواصل منقوطة في آخر كل سطر.
- **بتستخدم في حاجات كتير:** مواقع، وتحليل بيانات، وذكاء اصطناعي، وأتمتة الشغل المتكرر.
- **مجتمعها كبير:** أي مشكلة هتقابلك، غالبًا حد قابلها قبلك وحلها موجود.

شكل الكود في بايثون

```
# ده تعليق: بايثون بتتجاهل أي سطر يبدأ بعلامة #
print("أهلًا بيك في خطوة")
```

الناتج

```
أهلًا بيك في خطوة
```

السطر ده بيطلب من بايثون إنها تطبع الكلام اللي بين القوسين. والكلام نفسه لازم يتحط بين علامتين تنصيص `" "`، عشان بايثون تفهم إنه نص مش أمر.

هتطلع من الملخص ده بإيه؟

هتقدر تكتب برنامج صغير بياخد بيانات من المستخدم، ويحسب بيها حاجة، ويقرر يطبع إيه حسب النتيجة. وده الأساس اللي كل البرامج الأكبر مبنية عليه.

التسطيب وأول برنامج

١. سطّب بايثون

1. ادخل على python.org ونزّل آخر إصدار لنظامك.
2. على ويندوز: في أول شاشة في التسطيب علّم على **Add Python to PATH** قبل ما تدوس Install. دي أهم خطوة، ولو نسيتها الأوامر مش هتشتغل من الـ terminal.
3. اتأكد إن التسطيب نجح: افتح الـ terminal واكتب:

```
python --version
```

الناتج

```
Python 3.13.1
```

الرقم عندك ممكن يكون مختلف، المهم إنه يبدأ بـ 3.

٢. سطّب VS Code

ده المحرر اللي هتكتب فيه الكود. نزّله من code.visualstudio.com، وبعد ما تفتحه ادخل على Extensions وسطّب إضافة **Python** اللي من Microsoft.

٣. أول برنامج

1. اعمل فولدر جديد وافتحه في VS Code.
2. اعمل ملف جديد اسمه `hello.py`. الامتداد `.py` هو اللي بيعرّف المحرر إن ده ملف بايثون.
3. اكتب فيه:

```
print("Hello, World!")
print("ده أول برنامج ليا")
```

4. شغّله من زرار التشغيل ▷ فوق على اليمين، أو من الـ terminal:

```
python hello.py
```

الناتج

```
Hello, World!
ده أول برنامج ليا
```

لو كتبت python وطلعلك إنها مش معروفة

غالبًا نسيت تعلّم على Add Python to PATH. أسهل حل إنك تشغّل ملف التسطيب تاني وتختار **Modify** وتفعّل الاختيار ده، أو على ويندوز جرّب الأمر `py` بدل `python`.

الخطوة ٣

المتغيرات

المتغير اسم بتحطه على قيمة عشان تقدر ترجعلها وتستخدمها بعدين. تخيّله علبة مكتوب عليها اسم، وجواها حاجة.

```
name = "محمود"
age = 25
print(name)
print(age)
```

الناتج

```
محمود
25
```

علامة `=` هنا معناها **"خزّن"** مش "يساوي" زي الرياضيات. السطر `age = 25` معناه: خزّن 25 في متغير اسمه `age`.

القيمة ممكن تتغير

```
score = 10
score = score + 5
print(score)
```

الناتج

```
15
```

بايثون بتحسب الجانب اليمين الأول (10 + 5)، وبعدين بتحط النتيجة في `score` مكان القيمة القديمة.

قواعد تسمية المتغيرات

مسموح	ممنوع	ليه؟
`user_name`	`user name`	الاسم مينفعش يبقى فيه مسافة
`price2`	`2price`	الاسم مينفعش يبدأ برقم
`total`	`print` أو `if`	دي كلمات محجوزة لبايثون

نصيحة

سمّي المتغير باسم يوضح هو شايل إيه. `price` أحسن بكتير من `x`. وبايثون بتفرّق بين الحروف الكبيرة والصغيرة: `Name` و `name` متغيرين مختلفين.

الخطوة ٤

أنواع البيانات

كل قيمة في بايثون ليها **نوع**، والنوع هو اللي بيحدد تقدر تعمل بيها إيه. ودي أشهر 4 أنواع:

النوع	بيشيل إيه	مثال
`int`	أرقام صحيحة	`age = 25`
`float`	أرقام بكسور عشرية	`price = 99.5`
`str`	نصوص	`city = "القاهرة"`
`bool`	صح أو غلط بس	`is_student = True`

تعرف النوع إزاي؟

```
print(type(25))
print(type(99.5))
print(type("25"))
print(type(True))
```

الناتج

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

خلي بالك من التالت: `"25"` بين علامتين تنصيص، يعني ده **نص** مكتوب فيه رقمين، مش رقم. والفرق ده هيفرق جدًا لما تيجي تحسب.

الـ bool

النوع ده ليه قيمتين بس: `True` و `False`، وبيتكتبوا بحرف أول كبير. هتستخدمه كتير جدًا مع الشروط في الخطوة ٨.

> **من غير ما تقول لبايثون**
>
> في بايثون مش محتاج تكتب نوع المتغير قبل ما تستخدمه. هي بتعرفه لوحدها من القيمة اللي حطيتها فيه.

الخطوة ٥

العمليات الحسابية والمقارنة

العمليات الحسابية

العلامة	معناها	مثال	النتيجة
`+`	جمع	`7 + 3`	`10`
`-`	طرح	`7 - 3`	`4`
`*`	ضرب	`7 * 3`	`21`
`/`	قسمة	`7 / 2`	`3.5`
`//`	قسمة من غير كسور	`7 // 2`	`3`
`%`	باقي القسمة	`7 % 2`	`1`
`**`	أُس	`2 ** 3`	`8`

القسمة العادية `/` بتطلّع `float` دايمًا، حتى لو القسمة مظبوطة: `10 / 2` بتطلّع `5.0` .

مثال: فاتورة بسيطة

```
price = 120
quantity = 3
total = price * quantity
print(total)
```

الناتج

```
360
```

المقارنة

المقارنة بتسأل سؤال، والإجابة دايمًا `True` أو `False` :

العلامة	معناها	مثال	النتيجة
`==`	يساوي	`5 == 5`	`True`
`!=`	مش بيساوي	`5 != 3`	`True`
`>` و `<`	أكبر من، أصغر من	`5 > 8`	`False`
`>=` و `<=`	أكبر من أو يساوي، أصغر من أو يساوي	`18 >= 18`	`True`

= ولا ==؟

`=` واحدة معناها "خزّن قيمة". و `==` اتنين معناها "قارن". ودي من أكتر الحاجات اللي بتلخبط في الأول.

الخطوة ٦

التعامل مع النصوص

دمج النصوص بـ f-string

أسهل طريقة تحط متغير جوه جملة إنك تكتب حرف f قبل علامة التنصيص، وتحط المتغير بين قوسين `{ }`.

```
name = "منة"
age = 21
print(f"اسمي {name} وعندي {age} سنة")
```

الناتج

```
اسمي منة وعندي 21 سنة
```

وتقدر تحط حسابات جوه القوسين كمان: `f"{age + 1} السنة الجاية هيبقى عندي"` .

حاجات مفيدة مع النصوص

```
word = "Python"
print(len(word))        # عدد الحروف
print(word.upper())     # كله كابيتال
print(word.lower())     # كله سمول
print(word[0])          # أول حرف
print(word[-1])         # آخر حرف
print(word[0:3])        # من أول حرف لحد قبل الحرف رقم 3
```

الناتج

```
6
PYTHON
python
P
n
Pyt
```

العدّ في بايثون بيبدأ من **صفر** مش من واحد. فأول حرف رقمه `0` ، والأرقام السالبة بتعدّ من الآخر.

جمع نصين

```
first = "خطوة"
second = "بخطوة"
print(first + " " + second)
```

الناتج

```
خطوة بخطوة
```

علامة `+` مع النصوص معناها "لزّق"، مش "اجمع". عشان كده ضفنا مسافة `" "` في النص، وإلا الكلمتين كانوا هيلزقوا في بعض.

الخطوة ٧

إدخال البيانات وتحويل الأنواع

الأمر `input()` بيوقف البرنامج ويستنى المستخدم يكتب حاجة ويدوس Enter:

```
name = input("اسمك إيه؟ ")
print(f"أهلًا يا {name}")
```

الناتج

```
اسمك إيه؟ يوسف
أهلًا يا يوسف
```

المشكلة المشهورة

input() بترجّع نص دايمًا، حتى لو المستخدم كتب رقم. فلو عايز تحسب بيه، لازم تحوّله الأول:

```
age = input("عندك كام سنة؟ ")
print(age + 1)   # هيطلع خطأ!
```

الناتج

```
TypeError: can only concatenate str (not "int") to str
```

بايثون بتقولك: مقدرش ألزّق نص في رقم. والحل إنك تحوّل النص لرقم بـ `int()`:

```
age = int(input("عندك كام سنة؟ "))
print(f"السنة الجاية هيبقى عندك {age + 1}")
```

الناتج

```
عندك كام سنة؟ 20
السنة الجاية هيبقى عندك 21
```

دوال التحويل

الدالة	بتحوّل لـ	مثال	النتيجة
`int()`	رقم صحيح	`int("42")`	`42`
`float()`	رقم بكسور	`float("3.5")`	`3.5`
`str()`	نص	`str(42)`	`"42"`

ولو المستخدم كتب حروف وإنت بتحوّل لرقم، زي `int("ahmed")`، البرنامج هيطلّع `ValueError`. ودي حاجة هتتعلم تتعامل معاها بعدين.

الخطوة ٨

الشروط: if و elif و else

الشروط بتخلي البرنامج ياخد قرار: لو الكلام ده صح، نفّذ الجزء ده، ولو لأ، نفّذ جزء تاني.

```
age = int(input("عندك كام سنة؟ "))

if age >= 18:
    print("تقدر تطلع رخصة")
else:
    print("لسه بدري شوية")
```

المسافة في أول السطر مهمة جدًا

السطور اللي تحت `if` لازم تبدأ بمسافة (4 مسافات أو Tab). دي اللي بتعرّف بايثون إن السطر ده جوه الشرط. ولازم تحط علامة `:` في آخر سطر الشرط.

أكتر من احتمال بـ elif

```
score = int(input("درجتك كام من 100؟ "))

if score >= 85:
    print("امتياز")
elif score >= 75:
    print("جيد جدًا")
elif score >= 50:
    print("ناجح")
else:
    print("حاول تاني")
```

الناتج

```
درجتك كام من 100؟ 78
جيد جدًا
```

بايثون بتجرّب الشروط بالترتيب من فوق لتحت، وأول شرط يطلع صح بتنفّذه وتسيب الباقي. عشان كده الترتيب يفرق: لو حطيت `score >= 50` الأول، أي درجة فوق 50 هتطلع "ناجح" حتى لو 95.

دمج الشروط

```
age = 20
has_id = True

if age >= 18 and has_id:
    print("اتفضل ادخل")
```

`and` معناها الشرطين لازم يكونوا صح، و `or` معناها كفاية واحد منهم، و `not` بتعكس الشرط.

الخطوة ٩

أخطاء كل المبتدئين بيقعوا فيها

الأخطاء جزء طبيعي من البرمجة، ورسالة الخطأ هي أحسن صاحب ليك: اقراها من آخر سطر، هتلاقي نوع الخطأ ومكانه.

الخطأ	معناه	الحل
`SyntaxError`	كتبت حاجة بايثون مش فاهماها	دوّر على قوس ناقص، أو `:` ناقصة بعد `if` ، أو علامة تنصيص مقفلتش
`IndentationError`	المسافات في أول السطر غلط	اتأكد إن السطور اللي جوه الشرط ليها نفس عدد المسافات
`NameError`	استخدمت اسم بايثون متعرفهوش	غالبًا غلطة في كتابة اسم المتغير، أو استخدمته قبل ما تعرّفه
`TypeError`	عملية على نوع مينفعش معاه	غالبًا بتجمع نص على رقم، حوّل الأنواع الأول
`ValueError`	القيمة مش مناسبة للتحويل	زي `int("abc")` ، اتأكد إن اللي داخل رقم

مثال: اقرا الخطأ

```
price = 50
print(prise)
```

الناتج

```
Traceback (most recent call last):
  File "shop.py", line 2, in <module>
    print(prise)
NameError: name 'prise' is not defined. Did you mean: 'price'?
```

الرسالة قالتلك كل حاجة: الخطأ في السطر 2، ونوعه `NameError` ، وكمان اقترحت عليك الاسم الصح. يعني غلطة كتابة بسيطة.

عادة هتوفّر عليك وقت كتير

شغّل الكود بتاعك كل شوية وإنت بتكتب، متكتبش 50 سطر وبعدين تشغّل. كده لو حصل خطأ، هتعرف إنه في آخر كام سطر كتبتهم.

تمارين وورقة مراجعة

تمارين

1. **التعارف:** اسأل المستخدم عن اسمه ومدينته، واطبع جملة واحدة فيهم الاتنين.
2. **الآلة الحاسبة:** خد رقمين من المستخدم، واطبع مجموعهم وفرقهم وحاصل ضربهم وقسمتهم.
3. **زوجي ولا فردي:** خد رقم، واطبع إذا كان زوجي ولا فردي. (تلميح: باقي القسمة على 2)
4. **الخصم:** خد سعر منتج. لو السعر أكبر من 500 اطبع السعر بعد خصم 10%، ولو لأ اطبع السعر زي ما هو.

حل التمرين التالت

```
number = int(input(" :اكتب رقم"))

if number % 2 == 0:
    print(f"رقم زوجي {number}")
else:
    print(f"رقم فردي {number}")
```

ورقة مراجعة سريعة

الكود	بيعمل إيه
`print("...")`	يطبع على الشاشة
`x = 10`	يخزّن قيمة في متغير
`type(x)`	يعرّفك نوع القيمة
`input("...")`	ياخد نص من المستخدم
`int()` / `float()` / `str()`	يحوّل بين الأنواع
`f"... {x} ..."`	يحط متغير جوه نص
`len(text)`	عدد الحروف
`if` / `elif` / `else`	ياخد قرار حسب شرط
`and` / `or` / `not`	يدمج أو يعكس الشروط

الخطوة الجاية

بعد ما تخلّص التمارين، هتبقى جاهز للتكرار بـ `for` و `while`، والقوائم `list`. ولو وقفت في أي تمرين، اسأل في **مجتمع خطوة** على khatwa.cloud.