مقدمة إلى لغة بايثون

الفصل الأول: مقدمة إلى بايثون

ما هي بايثون؟

بايثون هي لغة برمجة عالية المستوى وسهلة التعلم، تُستخدم في تطوير التطبيقات، الذكاء الاصطناعي، تحليل البيانات، تطوير الويب، وغير ذلك.

مميزات بايثون

- سهلة التعلم والقراءة
- تدعم البرمجة الكائنية
- تمتلك مكتبة قياسية ضخمة
- متعددة الاستخدامات

تثبيت بايثون

يمكنك تحميل بايثون من الموقع الرسمي: python.org

الفصل الثاني: أساسيات بايثون

المتغيرات وأنواع البيانات

```
x = 10  # عدد صحيح
y = 3.14 # عدد عشري
z = "Hello" # نص²
```

شرح الكود:

- x هو متغير يحتوي على عدد صحيح.
- y يحتوي على عدد عشري (Float).
- z يحتوي على نص (String).

العمليات الحسابية

```
sum = 10 + 5  # الجمع
sub = 10 - 5  # الطرح
mul = 10 * 5  # الضرب
div = 10 / 5  # القسمة
mod = 10 % 3  # باقي القسمة
pow = 2 ** 3  # الأس
```

درس مستخلص:

بايثون تدعم العمليات الحسابية الأساسية مثل:

- + للجمع
- - للطرح
- * للضرب
- / للقسمة
- % لاستخراج باقي القسمة
- ** لحساب القوة (الأس)

الفصل الثالث: التحكم في تدفق البرنامج

الجمل الشرطية

```
x = 10
if x > 5:
    print("x أكبر من 5")
elif x == 5:
    print("x يساوي 5")
else:
    print("x أصغر من 5")
```

تمرين:

اكتب برنامجًا يطلب من المستخدم إدخال رقم، ثم يحدد ما إذا كان الرقم موجبًا أو سالبًا أو صفرًا.

الفصل الرابع: الحلقات التكرارية

```
for i in range(5):
    print(i)
x = 0
while x < 5:
    print(x)
    x += 1
```

تمرين:

اكتب برنامجًا يطبع الأعداد الزوجية بين 1 و 20 باستخدام حلقة for.

الفصل الخامس: الدوال في بايثون

```
def greet(name):
    return "مرحبا " + name
print(greet("أحمد"))
```

تمرين:

اكتب دالة تحسب مجموع قائمة من الأرقام.

الفصل السادس: التعامل مع الأخطاء

```
try:
    x = int(input("أدخل رقمًا: "))
    print(10 / x)
except ZeroDivisionError:
    print("لا يمكن القسمة على صفر!")
except ValueError:
    print("الرجاء إدخال رقم صحيح!")
```

تمرين:

عدل الكود أعلاه بحيث يعيد طلب الإدخال عند حدوث خطأ.

الفصل السابع: التعامل مع الملفات

```
with open("file.txt", "w") as file:
    file.write("مرحبًا بالعالم!")
```

تمرين:

اكتب برنامجًا يقرأ محتوى ملف نصي ويعرضه على الشاشة.

الفصل الثامن: البرمجة الكائنية (OOP) في بايثون

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def greet(self):
        return f"مرحبا، أنا {self.name}"

person = Person("أحمد", 25)
print(person.greet())
```

تمرين:

أنشئ كائن Car يحتوي على brand و year وطريقة لطباعة التفاصيل.

الفصل التاسع: التعامل مع قواعد البيانات

```
import sqlite3
```

```
conn = sqlite3.connect("database.db")
()cursor = conn.cursor
cursor.execute("CREATE TABLE users (id INTEGER PRIMARY KEY, name TEXT)")
()conn.commit
()conn.close
```

تمرين:

أضف سجلات إلى جدول users ثم اعرضها.

الفصل العاشر: مشاريع عملية

آلة حاسبة بسيطة

```
:()def calculator
((" :أدخل الرقم الأول")input)num1 = float
(" :(/ ,* ,- ,+) أدخل العملية")op = input
((" :أدخل الرقم الثاني")input)num2 = float
:"+" == if op
print(num1 + num2)
:"-" == elif op
print(num1 - num2)
:"*" == elif op
print(num1 * num2)
:"/" == elif op
print(num1 / num2)
:else
("عملية غير صالحة")print
()calculator
```

توليد كلمات مرور عشوائية

```
import random

import string

:def generate_password(length=8)

chars = string.ascii_letters + string.digits + string.punctuation

return "".join(random.choice(chars) for _ in range(length))

print(generate_password(12))
```

الخاتمة

بايثون لغة قوية وسهلة التعلم. يمكنك الاستمرار في تعلم المزيد من المفاهيم المتقدمة مثل البرمجة المتعددة المسارات، تحليل البيانات، وتطوير تطبيقات الويب!