الوحدة الأولى: أساسيات البرمجة والمتغيرات

1- المتغيرات (Variables)

- **تعريف المتغير:** هو اسم نستخدمه لتخزين قيمة في الذاكرة. يمكن أن تتغير قيمته أثناء تنفيذ البرنامج.
- **قواعد تسمية المتغيرات:**
  - يبدأ بحرف (a-z, A-Z) أو شرطة سفلية _.
  - لا يمكن أن يبدأ برقم.
  - لا يحتوي على مسافات أو علامات (- أو @ أو %...).
  - يميز بين الحروف الكبيرة والصغيرة. (Case-sensitive)
- **إعادة تعيين المتغير:** يمكنك تغيير قيمته ونوعه في أي وقت.

```python
x = 100   # عدد صحيح
x = "مرحبا" # أصبح نصا
```

- **المتغيرات العامة والمحلية:**
  - **محلي (Local):** يُعرف داخل دالة ولا يُرى خارجها.
  - **عالمي (Global):** يُعرف خارج أي دالة ويمكن استخدامه في كل مكان.
- **الثوابت (Constants):** حسب الاتفاق، نكتبها بأحرف كبيرة، لكن بايثون لا تمنع تغييرها.

```python
PI = 3.1416
```

2- الأعداد (Numbers)

- **الأنواع الرئيسية:**
  - **Integer (int):** أعداد صحيحة (مثال: 2024, -10, 5).
  - **Float:** أعداد عشرية (مثال: 3.14, -0.5).
- **العمليات الحسابية:**

- +جمع، -طرح، *ضرب.
- /قسمة (تعطي عددًا عشريًا).
- //قسمة صحيحة (تقرب لأسفل).
- %باقي القسمة.
- **الأس.

- **أولويات العمليات: (PEMDAS)**

1. أقواس ()
2. أسس **
3. ضرب وقسمة * // / %
4. جمع وطرح + -

- **تحويل الأنواع:**
  - int(): يحول إلى عدد صحيح.
  - float(): يحول إلى عدد عشري.
  - str(): يحول إلى نص.

3- النصوص(Strings)

- **تعريف النص** :يُكتب بين علامتي تنصيص مفردة '...'أو مزدوجة "...".
- **الخصائص:**
  - **غير قابلة للتغيير (Immutable):** لا يمكن تعديل حرف واحد داخل النص.
  - **مفهرسة** :الحرف الأول رقمه 0.
- **عمليات مهمة على النصوص:**
  - **التقطيع (Slicing)** : text[start:end]
  - **الدمج (Concatenation):** "Hello" + " " + "World"
  - **التكرار:** "Ha" * 3 --> HaHaHa
- **تنسيق النصوص (f-strings):** الأسهل والأحدث.

```python
name = "أحمد"
age = 20
print(f"اسمي أنا {name} وعمري {age} سنة")
```

- **دوال مهمة:**
  - **lower():** تحويل إلى أحرف صغيرة.
  - **upper():** تحويل إلى أحرف كبيرة.
  - **strip():** حذف المسافات الزائدة.
  - **replace(old, new):** استبدال جزء من النص.
  - **split():** تحويل النص إلى قائمة.
  - **len():** حساب طول النص.

الوحدة الثانية: هياكل البيانات الأساسية

1- القوائم(Lists)

- **تعريفها:** مجموعة مرتبة من العناصر، قابلة للتعديل، تكتب بين قوسين مربعين .[]

```python
درجات = [18, 15, 12, 19]
طلاب = ["أحمد", "سارة", "محمد"]
```

- **العمليات الرئيسية:**
  - **الوصول:** درجات [0](العنصر الأول).
  - **التعديل:** درجات[1] = 16.
  - **الإضافة:** درجات.append(20) (في النهاية، درجات.insert(1, 14) (في مكان محدد).
  - **الحذف:** درجات.remove(12) (بالحقيمة، درجات.pop() (حذف آخر عنصر).

- **الترتيب**: درجات.sort() (تصاعدي)، درجات.reverse() (عكس الترتيب).
- **الطول**: len(درجات).

2- القواميس (Dictionaries)

- **تعريفها**: مجموعة غير مرتبة من أزواج (مفتاح: قيمة)، تكتب بين أقواس مجعدة {}.

```python
طالب = {
    "الاسم": "سارة",
    "الدرجة": 19.5,
    "المادة": "اقتصاد"
}
```

العمليات الرئيسية:

- **الوصول**: طالب["الاسم"] أو طالب.get("الاسم").
- **الإضافة/التعديل**: طالب["العمر"] = 21 (يضيف مفتاح جديد أو يعدل موجودًا).
- **الحذف**: del طالب["المادة"] أو طالب.pop("الدرجة").

3- المجموعات (Sets)

- **تعريفها**: مجموعة غير مرتبة من العناصر **الفريدة** (بدون تكرار)، تكتب بين {}.

```python
اعداد = {1, 2, 3, 3}  # النتيجة ستكون {1, 2, 3}
```

- **العمليات الرئيسية:**
  - **الإضافة**: اعداد.add(4).
  - **الحذف**: اعداد.remove(2).
  - **عمليات رياضية**: الاتحاد |، التقاطع &، الفرق -.

الوحدة الثالثة: التحكم في تدفق البرنامج(Statements)

1- الشروط(if, elif, else)

- لتنفيذ قرارات بناءً على تحقق شرط.

```python
تقدير = 85
if تقدير >= 90:
    print("ممتاز")
elif تقدير >= 80:
    print("جيد جدا")
else:
    print("مقبول")
```

2- الحلقات التكرارية(Loops)

- for: نستخدمها عندما نعرف عدد التكرارات أو للتكرار على عناصر مجموعة.

```python
# تكرار على قائمة
for طالب in ["أحمد", "سارة", "محمد"]:
    print(f"مرحبا {طالب}")

# تكرار باستخدام range(start, stop, step)
for i in range(1, 6):
    print(i)
```

- while: نستخدمها عندما لا نعرف عدد التكرارات (تتكرر طالما الشرط صحيح).

```python
محاولة = 1
while محاولة <= 3:
    print(f"محاولة رقم {محاولة}")
    محاولة += 1
```

3- الدوال(Functions)

- **تعريفها :** هي كتلة من الكود تعمل معًا لتنفيذ مهمة محددة، وتُستخدم لإعادة استخدام الكود وتنظيمه.
- **هيكل الدالة :**

```python
def اسم_الدالة(معاملات):
    # الكود الذي ينفذ المهمة
    return قيمة_النتيجة # اختياري
```

- **مثال:**

```python
def احسب_الضريبة(السعر, نسبة_الضريبة=0.14):
    ضريبة = السعر * نسبة_الضريبة
    return ضريبة

ضريبة_سلعة = احسب_الضريبة(100) # ستكون 14
print(ضريبة_سلعة)
```

الوحدة الرابعة: المكتبات العلمية(NumPy & Pandas)

1- مكتبة (NumPy)الأعداد والمصفوفات

- **لماذا نستخدمها؟** للتعامل مع المصفوفات (Arrays) والعمليات الحسابية المتقدمة والخطية بسرعة عالية.
- **التركيز للمحاسبة والاقتصاد:**
  - **المصفوفة (array):** هيكل يخزن بيانات متجانسة (كلها أعداد مثلا) لإجراء عمليات رياضية عليها.
  - **المصفوفة المستطيلة (Matrix):** مصفوفة ذات بعدين (صفوف وأعمدة)، أساسية في الإحصاء والاقتصاد القياسي.

```python
import numpy as np

# إنشاء مصفوفة من قائمة
أسعار = np.array([100, 150, 200, 250])

# العمليات على المصفوفة تجري على كل العناصر تلقائيا
أسعار_بعد_الخصم = أسعار * 0.9  # جميع الأسعار تصبح أقل بـ 10%
print(أسعار_بعد_الخصم)  # [225 .180 .135 .90.]
```

2- مكتبة (Pandas) تحليل البيانات

- **لماذا نستخدمها؟** لتحليل البيانات الجدولية) مثل ملفات Excel أو (CSV والتعامل معها بسهولة.
  - **Series:** عمود واحد.
  - **DataFrame:** جدول كامل (الأهم بالنسبة لك).
- **العمليات الأساسية للاقتصاديين:**
  - **قراءة ملفات بيانات :**

```
df = pd.read_excel('ملف_مبيعات.xlsx').
```

  - **تصفح البيانات:**

df.info() (معلومات عن الأعمدة)، df.head() (أول 5 أسطر).

- **الوصول للأعمدة:**

  `df["المبيعات"]` أو `df.المبيعات`.

- **تصفية البيانات (filter):** عرض الأسطر التي تحقق شرطا.

```python
# عرض المبيعات التي أكبر من 1000
مبيعات_كبيرة = df[df["المبيعات"] > 1000]
```

العمليات الإحصائية:

`df["المبيعات"].mean()` (المتوسط)، `.sum()` (المجموع)، `.max()` (الأعلى)

- **التجميع (groupby):** تجميع البيانات حسب فئة معينة.

```python
# حساب إجمالي المبيعات لكل منطقة
المبيعات_حسب_المنطقة = df.groupby("المنطقة")["المبيعات"].sum()
```