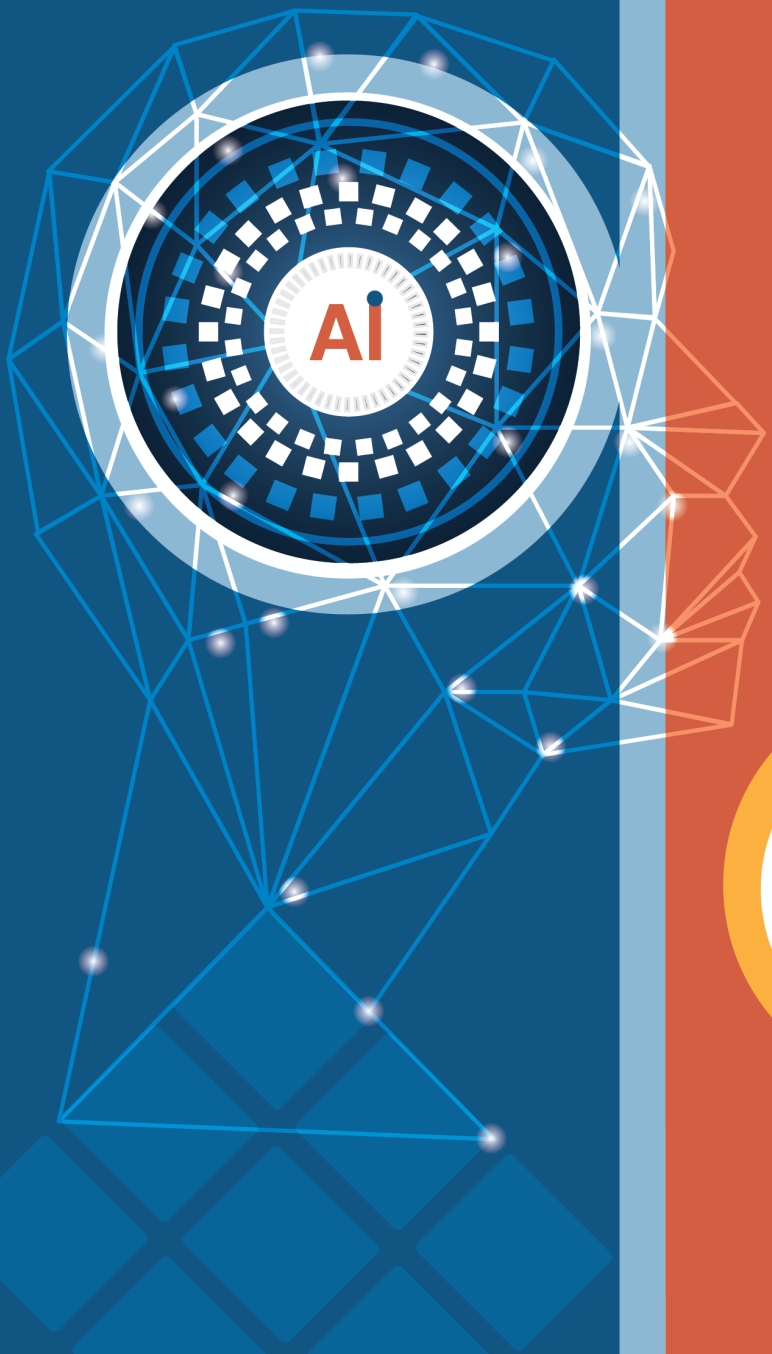
وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويـت

# دليل المعلم
# تقنية المعلومات

الصف العاشر - الجزء الثاني




PYTHON

# دليل المعلم
# تقنية المعلومات

## الصف العاشر


إشراف

أ/ منى سالم عوض سالم

أ/ ماجدة إبراهيم مزعل الدوسري

أ/ فاطمه خليف غنيم العنزي

أ/ إيمان عبد العزيز أحمد الفارسي

أ/ ساره ياسين عبدالله الامير

إخراج

أ/ سنيه محمد علي المؤمن

# المحـــــتوى

# المقـــدمة

عزيزي المعلم

يسعدنا أن نقدم لك هذا الدليل الشامل الذي يهدف إلى دعمك في تقديم مادة تقنية المعلومات لمتعلمي الصف العاشر (الجزء الثاني) بما في ذلك مفاهيم الذكاء الاصطناعي وبرمجة بايثون و بصورة فعالة ومبتكرة، نحن ندرك أن تقديم هذه المادة الحيوية يحتاج إلى أدوات ومعارف حديثة تتماشى مع تطورات العصر الرقمي واحتياجات المتعلمين.

يتضمن هذا الدليل استراتيجيات تعليمية متنوعة تتناسب مع الفئة العمرية للمتعلمين، مثل التعلم المباشر (المحاضرة)، التعلم التعاوني، المناقشة والحوار، التعلم القائم على حل المشكلات، والتعلم القائم على المشاريع، كما يوفر الدليل أنواعًا مختلفة من التقويمات، سواء التكوينية منها أو النهائية، لضمان تقييم شامل لتقدم المتعلمين ومساعدتهم على تحسين أدائهم بشكل مستمر.

لضمان تجربة تعليمية شاملة، يحتوي هذا الدليل على خطط مفصلة لكل وحدة دراسية تشمل أهداف الوحدة، الموضوعات الرئيسية، الأنشطة التعليمية، المواد والأدوات اللازمة، وأساليب التقويم. تم تصميم نتاجات التعلم بوضوح لكل وحدة لتمكين المتعلمين من معرفة ما يُتوقع منهم تحقيقه بنهاية كل درس.

فيما يخص برمجة بايثون، فإن الدليل يحتوي على موارد مفيدة لشرح المفاهيم البرمجية الأساسية، بما في ذلك أمثلة عملية وأنشطة تطبيقية لمساعدة المتعلمين على فهم البرمجة بطريقة سلسة وممتعة.
كما يتضمن الدليل قائمة شاملة بالمصطلحات الحاسوبية الهامة، بالإضافة إلى الحلول النموذجية لأوراق العمل لتيسير عملية التصحيح. يهدف هذا الدليل إلى تزويدك بالأدوات والمعارف اللازمة لتحسين عملية التعلم والتعليم، ودعمك في التحضير الفعال للدرس وتنفيذه بنجاح.

أما بالنسبة للذكاء الاصطناعي، إن الهدف من استخدامها هو تمكين المعلمين و المتعلمين من تطوير إنتاجيتهم و كذلك تقديم الحلول المستمرة لتطوير العملية التعليمية وزيادة فاعليتها وتطوير أنظمة وبرمجيات قادرة على محاكاة العقل البشري و تنفيذ مهام معقدة تكون قادرة على التعلم من التجارب و الاستنتاج والتنبؤ وحل المشكلات و غيرها من مهارات يتميز بها الانسان.

نحن على يقين بأن هذا الدليل سيكون مرشدك الأمثل في رحلتك التعليمية، ويساعدك على تقديم دروس ممتعة، تساهم في تطوير مهارات المتعلمين وتعزيز فهمهم للتكنولوجيا، الذكاء الاصطناعي، وبرمجة بايثون. نأمل أن تجد في هذا الدليل كل ما تحتاجه لتحقيق أفضل النتائج في تدريس منهج الصف العاشر

مع أطـيب التمـــنيات

# الوحدة الأولى : برمجة Python

# تصحيح الأخطاء والاستثناءات

# Debugging and Exceptions

01

## نتاجات التعلم: تصحيح الأخطاء والاستثناءات.

| | |
|---|---|
| 1 | التعرف على مفهوم الأخطاء والاستثناءات كعنصر أساسي في البرمجة. |
| 2 | التعرف على أنواع الأخطاء البرمجية في بايثون (Syntax Errors - Semantic Errors - Logic Errors ) |
| 3 | تمييز الأخطاء اللغوية (Syntax Errors) التي تظهر بسبب عدم اتباع قواعد اللغة البرمجية. |
| 4 | تمييز الأخطاء الدلالية (Semantic Errors): التي تظهر أثناء تشغيل البرنامج نتيجة ظروف غير متوقعة يكتشفها المفسر(Python Interpreter) . |
| 5 | تمييز الأخطاء المنطقية (Logic Errors): بأنها تظهر نتائج غير صحيحة أو غير متوقعة لا يتم اكتشافها من قبل المفسر (Python Interpreter) وتكتشف من قبل المبرمج. |
| 6 | التعرف على أنواع الاستثناءات ومنها(Value Error- NameError - Index Error - Type Error) |
| 7 | استخدام الاستثناءات (Exceptions) للتعامل مع الأخطاء. |
| 8 | تعلم كيفية استخدام التعليمة البرمجية 'try/except' للتعامل مع الأخطاء المتوقعة أثناء تنفيذ البرنامج واستمراريته دون توقف . |

## المصطلحات الحاسوبية: تصحيح الأخطاء والاستثناءات.

Debugging تصحيح الأخطاء


Syntax Errors الأخطاء اللغوية


Semantic Errors الأخطاء الدلالية

Logic Errors الأخطاء المنطقية

try/except تعليمة

ValueError

ملاحظة للمعلم

الاطلاع على تقنيات تصحيح الأخطاء الشائعة Python وسيتم عرض ابسطهم التي تتناسب مع الفئة العمرية للمتعلمين .

### الطباعة باستخدام print()

الأسلوب الأبسط ، حيث يمكنك وضع أمر print() بعد تعليمات برمجية لرؤية قيم المتغيرات أو التحقق من الخطوات

### نقاط التوقف (BreaKpoints)

هي ميزة في PyCharm تتيح لك إيقاف التعليمة البرمجية مؤقتًا عند سطر معين، للتمكن من فحص قيم المتغيرات أو التحقق من التعليمات البرمجية خطوة بخطوة.

### قراءة رسائل الخطأ في PyCharm

PyCharm يعرض رسائل خطأ مفصلة تساعدك على معرفة نوع الخطأ وموقعه

### كتابة comment قبل التعليمات البرمجية

لشرح الهدف من التعليمات البرمجية أو توضيح الأجزاء (المقاطع) البرمجية .

### التعامل مع الأخطاء باستخدام try/except

في حالة التوقع حدوث خطأ، يمكن استخدام **try/except** لمنع توقف البرنامج فجأة يساعد على التعامل مع الأخطاء وتوفير رسائل توضيحية للمستخدم.

وغيرها من التقنيات لتصحيح الأخطاء

# أمثلة اضافية: الأخطاء اللغوية Syntax Errors

مثـــال(1) :

الخطأ:

نسيان إغلاق القوس في دالة **print()** .

التعليمات البرمجية

input

```
1  x = 5
2  y = 10
3  print(x + y
```

Out put

```
  File "C:\Users\HP\Documents\pythonpart 1\1-1.py", line 3
    print(x + y
         ^
SyntaxError: '(' was never closed
```

مثـــال(2) :

الخطأ:

نسيان النقطتين **(:)** في الجمل الشرطية أو الحلقات **(Loops)**.

التعليمات البرمجية

input

```
1  if x > 5
2    print("x is greater than 5")
```

Out put

```
    if x > 5
            ^
SyntaxError: expected ':'
```

# أمثلة إضافية:

مثـــال(3) :

الخطأ في التعليمة البرمجية هو **Indentation Error**، ويحدث عندما يكون هناك مشكلة في المسافة البادئة (الإزاحة) في التعليمية البرمجية في**Python**.

التعليمات البرمجية

**Input**

```
if x > 5:
print("x is greater than 5")
```

**Out put**

```
    print("x is greater than 5")
    ^
IndentationError: expected an indented block after 'if' statement on line 1
```

# الأخطاء الدلالية:(Semantic Errors)

الأخطاء الدلالية (Semantic Errors)هي نوع من الأخطاء البرمجية التي تحدث عندما يكون التعليمات البرمجية مكتوبة بشكل صحيح من الناحية اللغوية (syntax) ولكن لا تقوم بالوظيفة المطلوبة بشكل صحيح بسبب ظرف معين على عكس الأخطاء اللغوية (Syntax Errors) التي تمنع تشغيل البرنامج من الأساس
الأخطاء الدلالية تكون أصعب في اكتشافها لأنها قد لا تؤدي إلى توقف البرنامج بل فقط تؤدي إلى نتائج خاطئة

من الضروري تنويع الأمثلة وتقديم سيناريوهات متعددة لإيصال مفهوم
الأخطاء الدلالية (Semantic Errors) بوضوح للمتعلم للتعرف على أهمية التفكير المنطقي السليم أثناء البرمجة، مما يسهل عليه اكتشاف وإصلاح الأخطاء في برامجه..

مثـــال(1) :

الخطأ الدلالي : بدلاً من استخدام عملية الجمع +، تم استخدام عملية الطرح -
النتيجة المتوقعة: يجب أن يكون result = x + y للحصول على النتيجة الصحيحة.

التعليمات البرمجية

Input

```
x = 10
y = 5
result = x - y   # الهدف هو حساب المجموع، لكن تمت عملية الطرح بدلاً من الجمع
print(result)
```

مثـــال(2) :

الخطأ الدلالي : جمع أطوال المستطيل (الطول و العرض ) بدلاً من عملية الضرب لحساب المساحة.
النتيجة المتوقعة : كان يجب أن يكون Area = width * height

التعليمات البرمجية

Input

```
width = 5
height = 10
area = width + height   # الهدف هو حساب المساحة ، ولكن تم جمع الأبعاد بدلاً من ضربها
print(area)
```

# تابع- الأخطاء الدلالية:(Semantic Errors)

مثـــال(3) :

**الخطأ الدلالي**:الشرط غير صحيح x<5 (وحسب البرنامج قيمة المتغير X يساوي 10 أي اكبر من 5 المذكورة في الشرط )، والبرنامج يطبع "x is greater than or equal to 5" وهو صحيح.

**الخطأ** : إذا كان الهدف هو التحقق مما إذا كان x أقل من 10، فيجب تعديل الشرط

**النتيجة المتوقعة**:إذا كان الهدف التحقق مما إذا كان x أقل من 10، يجب تعديل الشرط ليصبح

if x<:10

التعليمات البرمجية

Input

```
x = 10
if x < 5:
    print("x is less than 5")
else:
    print("x is greater than or equal to 5")
```

Out put

```
x is greater than or equal to 5

Process finished with exit code 0
```

مثـــال(4) :

مثال الكتاب ص 17

**الخطأ الدلالي**:عندما يقوم المستخدم بإدخال قيمة عدد صحيح للمتغير الأول والثاني تتم عملية القسمة بشكل صحيح ولكن عند ادخال قيمة 0 للمقسوم عليه .(var_y) يحاول البرنامج تنفيذ عملية القسمة var_x / var_y، فيؤدي إلى خطأ في البرنامج بسبب القسمة على الصفر.، وهو أمر غير مقبول رياضيًا .

**النتيجة المتوقعة** : إذا لم يتم التعامل المستخدم مع هذا الخطأ بشكل مناسب بإدخال اعداد صحيحة (سيقوم البرنامج بإصدار ZeroDivisionError ) وسيتوقف عن العمل .

التعليمات البرمجية

Input

```
var_x = int(input('Enter first number: '))
var_y = int(input('Enter second number: '))
var_z = var_x / var_y
print(var_z)
```

تابع مثــال(4) :

**Out put**

```
Enter first number: 6
Enter second number: 0
Traceback (most recent call last): Explain with AI
  File "C:\Users\HP\Documents\pythonpart 1\main.py", line 3, in <module>
    var_z = var_x / var_y
            ~~~~~~^~~~~~~
ZeroDivisionError: division by zero
```

### التأكيد على المتعلم

خطأ برمجي من نوع **ZeroDivisionError** نوع من الأخطاء (الاستثناءات) في بايثون عندما يحاول البرنامج قسمة عدد على الصفر

مثــال(5) :

**الخطأ الدلالي**: يفترض البرنامج يقوم بطباعة القيمة الأولية للمتغير count بشكل مستمر بدون توقف لأن المتغير لا يتم تحديثه، مما يؤدي إلى حلقة لا نهائية.

**النتيجة المتوقعة**: كان يجب إضافة المتغير على ان يزيد بمعدل **1** في كل مره في الحلقة التكرارية **count += 1** ويتم تحديث المتغير في كل تكرار للحلقة.

### التعليمات البرمجية

**Input**

```
count = 0
while count < 5:
    print("Counting:", count)
    # count += 1  # تم نسيان من قبل المبرمج تحديث المتغير
```

**Out put**

```
Counting: 0
Counting: 0
Counting: 0
Counting: 0
Counting: 0
Counting: 0
```

بعد الضغط على أداة **Stop** أو **Ctrl+F2** تظهر الشاشة التالية:

**Out put**

```
  File "C:\Users\HP\Documents\pythonpart 1\main.py", line 3, in <module>
    print("Counting:", count)
KeyboardInterrupt
```

## الأخطاء المنطقية: (Logic Errors)

هي أخطاء لا يستطيع المفسر (Python Interpreter) اكتشافها وتكتشف من قبل المبرمج، مما يعني أن البرنامج يعمل دون مشاكل واضحة أو توقف، ولكنه يظهر نتائج غير متوقعة أو غير صحيحة. هذه الأخطاء تحدث بسبب نتيجة لخلل في منطق البرنامج أو في كيفية تنفيذ العمليات.

مثـــال(3) :
الكتاب ص 18

### الخطأ المنطقي (Logic Error)

البرنامج يعمل بشكل صحيح ، ولا يوجد أخطاء أثناء التنفيذ ولكنه لا يعطي النتيجة المتوقعة بسبب خطأ في المنطق المستخدم في هذه الحالة، تم استخدام صيغة لحساب محيط المستطيل بدلاً من مساحة المستطيل .

النتيجة المتوقعة: الهدف هو حساب المساحة، فيجب تعديل الصيغة لتصبح :

( var_area = var_l * var_w )

التعليمات البرمجية

Input

```
var_l = 5
var_w = 6
# المطلوب حساب المساحة و والعملية المحسوبه لقياس المحيط
var_area = 2* ( var_l + var_w)
# طباعة النتيجة
print('Area = ', var_area)
```

Out put

```
Area =  22

Process finished with exit code 0
```

# تصحيح الأخطاء

مثــال(4) : الكتاب ص 18

Input

```
x = input('Enter first number: ')     # تخزين المدخل في المتغير x
if x == 0                             # مقارنة x مع الصفر
print('x is equal to 0.')
else:
    print('x is not equal to 0.')
```

## الأخطاء المنطقية Logic Errors

- x = input('Enter first number: ')

لم يتم تحويل المدخل في المتغير X من نص الى قيمة عددية باستخدام أحد الدوال (int, float,.. ).

- في السطر الذي يحتوي على جملة الشرطية if x == 0، يتم مقارنة المتغير x مع الرقم 0 المشكلة هنا هي أن المدخل x يتم استدعاؤه باستخدام دالة input()، والتي تقوم بإرجاع القيمة كـ سلسلة نصية(String) ، حتى لو أدخل المستخدم رقمًا.

## الأخطاء اللغوية Syntax Errors

- عدم وجود : في عبارة if.
- عدم وجود مسافة بادئة في عبارة print الأولى.

# الاستثناءات (Exceptions)

الاستثناءات في البرمجة هي أخطاء قد تحدث أثناء تشغيل البرنامج نتيجة لأحداث غير متوقعة تساعدك الاستثناءات على التعامل مع الأخطاء بطريقة منظمة بحيث يستمر البرنامج في العمل دون توقف مفاجئ.

ملاحظة للمعلم

أنه يمكن استخدام التعليمات البرمجية **try/except** ، لتحديد الجزء الذي قد يحدث فيه خطأ، وللتأكد من استمرار عمل البرنامج.

## أمثلة على الأخطاء الشائعة (الاستثناءات):

يوجد العديد من أنواع الأخطاء ( الاستثناءات)التي قد تحدث أثناء تشغيل البرنامج، ومن أشهرها:

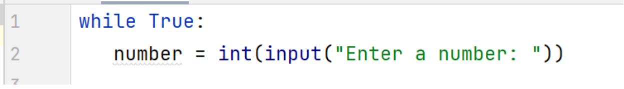
- **Type Error** يحدث عندما تحاول استخدام نوع غير صحيح من البيانات.
- **Index Error** يحدث عندما تحاول الوصول إلى عنصر غير موجود في قائمة.
- **NameError** يحدث عندما تحاول استخدام متغير لم يتم تعريفه.
- **Value Error** يحدث عندما تدخل قيمة غير صحيحة.

## كيفية استخدام try/except

- التعليمة البرمجية **try** تحتوي على التعليمات البرمجية الذي قد يحدث فيه خطأ.
- التعليمة البرمجية **except** تحتوي على التعليمات البرمجية الذي سيتم تنفيذه إذا حدث خطأ في الجزء الموجود داخل **try**.

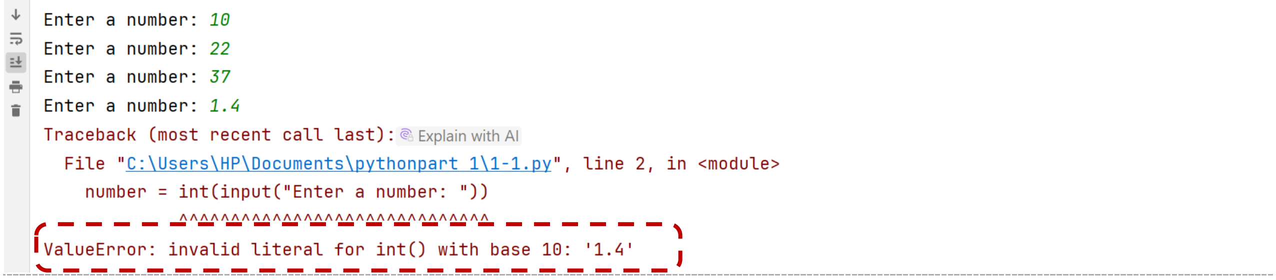
مثــال(5) : الكتاب ص 20

input



```
while True:
    number = int(input("Enter a number: "))
```

Out put



```
Enter a number: 10
Enter a number: 22
Enter a number: 37
Enter a number: 1.4
Traceback (most recent call last):
  File "C:\Users\HP\Documents\pythonpart 1\1-1.py", line 2, in <module>
    number = int(input("Enter a number: "))
             ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
ValueError: invalid literal for int() with base 10: '1.4'
```

## شرح المثال:

التعليمة البرمجية: while True تعني أن البرنامج سيستمر في تنفيذ التعليمات البرمجية الموجودة داخل الحلقة بشكل مستمر

- number = int(input("Enter a number: ")) .

- التعليمة input() تستقبل من المستخدم قيمة .

- الدالة int() تحويل القيمة المدخلة إلى عدد صحيح .

- إذا أدخل المستخدم قيمة صحيحة (مثل 10 أو 22) سيتم تخزينها في متغير number.

- إذا أدخل المستخدم قيمة غير صحيحة (مثل نص أو رقم عشري)، ظهور خطأ من نوع Value Error .

## استخدام while True: مع try/except:

للتعامل مع المدخلات الخاطئة ، استخدام try/except داخل الحلقة لكسر الحلقة عند إدخال رقم صحيح فقط .

**Input**

```
while True:
    try:
        number = int(input("Enter a number: "))
        print("You entered the number: ", number)

    except ValueError:
        print("Invalid number ! Please enter a valid integer.")
```

**Out put**

```
Enter a number: 10
You entered the number:  10
Enter a number: 55
You entered the number:  55
Enter a number: 1.5
Invalid number ! Please enter a valid integer.
Enter a number: 2
You entered the number:  2
Enter a number:
```


إليك فكرة لشرح break و continue بطريقة يمكن للمتعلمين فهمها بسهولة

Break الخروج من الحلقة

يمكنك تشبيه break بزر "التوقف" الذي يسمح لك بالخروج من الحلقة وإيقاف التكرار.

مثــال حياتي:

تخيل أنك في ملعب وتلعب لعبة، وتكرر نفس الحركة عدة مرات (هذه هي الحلقة).

لكن في لحظة معينة ، قررت التوقف عن اللعب والخروج من الملعب ( breaK )

تتوقف عن تكرار الحركة وتغادر اللعبة.

- breaK يستخدم لإنهاء الحلقة بالكامل والخروج منها.
- continue يستخدم لتجاوز التكرار الحالي والانتقال إلى التكرار التالي دون إنهاء الحلقة

## استخدام while True: مع try/except:

عند توظيف دالة ( breaK ) في مثال 5 ص 20

**Input**

```python
while True:
    try:
        number = int(input("Enter a number: "))
        print("You entered the number:", number)
        answer = input("Do you want to stop? (yes/no): ")
        if answer == "yes":
            print("Game over!")
            break  # هنا يتم كسر الحلقة والخروج منها
        else:
            print("We're still playing!")
    except ValueError:
        print("Invalid number! Please enter a valid integer.")
```

**Out put**

```
Enter a number: 6
You entered the number: 6
Do you want to stop? (yes/no): no
We're still playing!
Enter a number: 1.5
Invalid number! Please enter a valid integer.
Enter a number: 4
You entered the number: 4
Do you want to stop? (yes/no): yes
Game over!
```

# حل أوراق العمل -ورقة عمل 1 ص 21:

| رقم السطر | الخطأ | نوع الخطأ | التصحيح | ملاحظات |
|---|---|---|---|---|
| 2 | Rashed | Syntax Error | "Rashed" | عدم استخدام علامتي التنصيص |
| 9 | input(f"أدخل وزن {player}: ") | Syntax Error | weight = float(input(f"أدخل وزن {player}: ")) | التأكد من كتابة حرف f قبل علامة الاقتباس |
| 9 | اذا أدخل المستخدم نص بدلا من الرقم<br>float(input(...)) | Semantic Error | ادخال أرقام صحيحة أو عشرية فقط | استخدام try/except لمعالجة إدخال غير صالح |
| 17 | If weight >63: | Logical Error | If weight < 63: | تصحيح عامل المقارنة > |
| 19 | Elif 63<= weight < 90 | Logical Error | Elif 63 > weight < 90 | تصحيح عامل المقارنة <= |

البرنامج بعد معالجة وتصحيح الأخطاء :

Input

```
#إنشاء قائمة بأسماء اللاعبين
players=['Ahmed','Khaled','Ali',Rashed,'Majed','Zaher']

#إنشاء قائمة لتخزين الأوزان
weights = []

#الحلقة للحصول على أوزان اللاعبين
for player in players:

    weight = float(input(f'أدخل وزن {player}: '))
    weights.append(weight)

#حساب وطباعة النتائج
for i in range(len(players)):
    player = players[i]
    weight = weights[i]

    if weight > 63:
        print(f' وزن{player}: {weight} وزن الريشة- ')
    elif 63 ≤ weight < 90:
    print(f'وزن{player}: {weight} وزن الوسط- ')
    else:
        print(f'وزن{player}: {weight} وزن الثقيل- ')
```

## تابع :حل أوراق العمل -ورقة عمل 1 ص 21:

البرنامج بعد إضافة الاستثناءات لمعالجة الأخطاء **try/except** .

**Out put**

```
# قائمة بأسماء اللاعبين
players = ['Ahmed', 'Khaled', 'Ali', 'Rashed', 'Majed', 'Zaher']
# قائمة لتخزين الأوزان
weights = []
# حلقة للحصول على أوزان اللاعبين مع معالجة الأخطاء
for player in players:
    try:
        weight = float(input(f" أدخل وزن {player}: "))
        weights.append(weight)
    except ValueError:
        print("خطأ: يجب إدخال رقم صحيح أو عشري. سيتم تعيين وزن افتراضي لهذا اللاعب.")
        weights.append(0.0)  # تعيين وزن افتراضي في حال حدوث خطأ
# حساب وطباعة النتائج
for i in range(len(players)):
    player = players[i]
    weight = weights[i]
    if weight < 63:
        print(f" وزن {player}: {weight} وزن الريشة - ")
    elif 63 ≤ weight ≤ 90:
        print(f" وزن {player}: {weight} وزن الوسط - ")
    else:
        print(f" وزن {player}: {weight} وزن ثقيل - ")
```

## حل أوراق العمل -ورقة عمل 2 الكتاب22:

| رقم السطر | الخطأ | نوع الخطأ | التصحيح | ملاحظات |
|---|---|---|---|---|
| 9 | if input_number = "تم": | Syntax Error | if input_number == "تم": | معامل == يعني المقارنة بين عنصرين |
| 13 | تحويل الإدخال إلى عدد عشري | Value Error | إضافة الاستثناءات try/except | |
| 16 | كتابة اسم المتغيرSum<br>تميز بين الأحرف الكبيرة والصغيرة | Syntax Error | استبدال sum بـ Sum | |
| 16 | المسافة البادئة | Syntax Error | تعديل المسافة البادئة | |
| 19 | قسمة على صفر عند حساب average | | القائمة numbers_list فارغة (لم يتم إدخال أي أعداد) لذا فإن القسمة على صفر ستؤدي إلى خطأ | إضافة شرط للتحقق من أن القائمة تحتوي على عناصر قبل حساب المتوسط |

### حل ورقة العمل :

**Input**

```
1   # الإعلان عن قائمة فارغة لتخزين الأعداد
2   numbers_list = []
3   sum = 0
4   # استخدام حلقة للسماح للمستخدم بإدخال الأعداد
5   while True:
6       # ()input الحصول على إدخال المستخدم باستخدام دالة
7       input_number = input(" :(أدخل 'تم' للانتهاء) الرجاء إدخال عدد")
8       # التحقق إذا أدخل المستخدم كلمة 'تم' للخروج من الحلقة
9       if input_number == "تم":
10          break
11      else:
12          try:
13              # تحويل النص الذي أدخله المستخدم إلى عدد
14              input_number = float(input_number)
15              # إضافة العدد إلى القائمة
16              numbers_list.append(input_number)
17              sum += input_number  # جمع العدد إلى المجموع
18          except ValueError:
19              print("خطأ: يرجى إدخال رقم صحيح أو 'تم' لإنهاء الإدخال.")
20              continue
21  # حساب متوسط الأعداد
22  if len(numbers_list) > 0:
23      average = sum / len(numbers_list)
24  else:
25      average = 0
26  # طباعة عناصر القائمة
27  print("الأعداد التي قمت بإدخالها: ")
28  print(numbers_list)
29  # طباعة متوسط الأعداد
30  print("المتوسط الحسابي:", average)
31
```

مثال إضافي :

لبرنامج آلة حاسبة بسيط يستقبل من المستخدم رقمين ثم يحدد نوع العملية الحسابية ثم يطبع النتيجة بعد التنفيذ , يحتوي هذا البرنامج على الأنواع الثلاثة من الأخطاء:

- الأخطاء النحوية(Syntax Errors) .
- الأخطاء المنطقية(Logical Errors) .
- الأخطاء الدلالية (Semantic Errors) .

Input

```
# آلة حاسبة بسيطة تجمع، تطرح، تضرب، وتقسم رقمين

# الحصول على الأرقام من المستخدم
num1 = input(" :أدخل الرقم الأول")
num2 = input(" :أدخل الرقم الثاني")

# اختيار العملية الحسابية
operation = input(" :(/ ,* ,- ,+) أدخل العملية التي تريد القيام بها")

# تحويل الأرقام إلى float مع معالجة الأخطاء
try:
    num1 = float(num1)
    num2 = float(num2)
except ValueError
    print(".خطأ: يرجى إدخال أرقام صحيحة")
    exit()

# إجراء العملية الحسابية وطباعة النتيجة
if operation == "+":
    result = num1 - num2  # خطأ دلالي: استخدام الطرح بدلاً من الجمع
elif operation == "-":
    result = num1 + num2  # خطأ دلالي: استخدام الجمع بدلاً من الطرح
elif operation == "*":
    result = num1 * 2  # خطأ دلالي: يجب أن يكون num2 هو العدد المضروب
elif operation == "/":
    if num2 == 0:
        print("خطأ: لا يمكن القسمة على الصفر")
    else:
        result = num1 / num1  # خطأ دلالي: القسمة على نفس العدد بدلاً من num2
else:
    print("خطأ: عملية غير صالحة")

# طباعة النتيجة
print(":النتيجة", result
```

## حل المثـــال:

| رقم السطر | الخطأ | نوع الخطأ | التصحيح | ملاحظات |
|---|---|---|---|---|
| 14 | نسيان النقطتين :بعد except ValueError | الأخطاء النحوية (Syntax Errors) | except ValueError: | عدم كتابة النقطتين : بعد except ValueError |
| 34 | نسيان إغلاق القوس في نهاية جملة الطباعة<br>print("النتيجة:", result) | الأخطاء النحوية (Syntax Errors) | print("النتيجة:", result) | |
| 25 | القسمة على num2 بدون التحقق مما إذا كان يساوي صفرًا<br>elif operation:"/" ==<br>result = num1 / num2 | الأخطاء المنطقية (Logical Errors) | Elif operation == "/":<br>if num2 == 0:<br>print ("خطأ: لا يمكن القسمة على الصفر")<br>else:<br>result = num1 / num2 | إضافة شرط للتحقق مما إذا كان num2 يساوي صفرًا قبل إجراء القسمة |
| 31 | إذا كانت العملية المدخلة (operation) صالحة إذا أدخل المستخدم رمزًا غير صالح<br>else:<br>print("خطأ: عملية غير صالحة") | | اضافة شرط if إضافي قبل الطباعة للتحقق مما إذا كانت العملية صالحة<br>if operation not in:["/" ,"*" ,"-" ,"+"]<br>print("خطأ: عملية غير صالحة") | |
| 19 | في عملية الجمع والطرح يتم تنفيذ عملية خاطئة فعند اختيار عملية الجمع (+) ، يتم الطرح، وعند اختيار الطرح (-) ، يتم الجمع<br>if operation == "+":<br>result = num1 - num2<br>elif operation == "-":<br>result = num1 + num2 | الأخطاء الدلالية (Semantic Errors) | if operation:"+" ==<br>result = num1 + num2<br>elif operation :"-" ==<br>result = num1 - num2 | استبدال العمليات لتكون صحيحة |
| 24 | في num1 في عملية الضرب، يتم ضرب 2 بدلاً من num2<br>elif operation == "*":<br>result = num1 * 2 | | elif operation == "*":<br>result = num1 * num2 | استبدال 2 بـ num2 لإتمام عملية الضرب بشكل صحيح |

02

# الوحدة الأولى : برمجة Python

## المكتباتLibraries:

## نتاجات التعلم: المكتباتLibraries:

| | |
|---|---|
| 1 | التعرف على مفهوم المكتبات ووظيفتها. |
| 2 | فهم دور المكتبات البرمجية كأدوات لإعادة استخدام وتنظيم المقاطع البرمجية بما يسهم في بناء مشروعات برمجية بكفاءة عالية. |
| 3 | اتقان طرق استيراد المكتبات في Python ، واستخدام دوالها وأدواتها بفعالية لتوفير الوقت والجهد البرمجي. |
| 4 | التعرف على المكتبات القياسية المدمجة في Python ، وكيفية استخدامها لأداء الوظائف الأساسية مثل العمليات الرياضية وإدارة الملفات. |
| 5 | التعرف على فوائد استخدام المكتبات Libraries. |
| 6 | التعامل مع المكتبات الخارجية تثبيتها، تحديثها، وإدارتها من خلال بيئة التطوير . |
| 7 | التعرف على أقسام المكتبات في البايثون. |
| 8 | التعرف على طرق استيراد المكتبات بأشكالها المختلفة . |

## المصطلحات الحاسوبية: المكتبات.

المكتباتPython Libraries

تحليل البيانات (Data Analysis)

الذكاء الاصطناعي (Artificial Intelligence)

تعلم الآلة (Machine Learning)

التعلم العميق (Deep Learning)

الأمن السيبراني (Cybersecurity)

# المصطلحات الحاسوبية: المكتبات.

تطوير الألعاب (Gaming Development)

المتغيرات variables

الدوال functions

والفئات classes

مكتبة الدوال الرياضية NumPy

مكتبة تحليل البيانات Pandas

مكتبة التصميم الجرافيكي مكتبة Matplotlib

مكتبة تحليل الصور والفيديوهات OpenCV

مكتبة تعلم الآلة TensorFlow

مكتبة التعلم العميقPyTorch

المكتبات القياسيةStandard Libraries

المكتبات الخارجية Third-Party Libraries

# تابع -المصطلحات الحاسوبية: المكتبات.

المكتبات الخاصة Customize Libraries

استيراد المكتبات Import Libraries

توثيق المكتبات pydoc

مكتبة Random

استيراد import

العمليات الحسابية math

التثبيت :(Installation)

واجهة سطر الأوامر Terminal

لتثبيت مكتبة برمجية خارجية الأمر pip install

أداة مدمجة في بيئات تطوير

للبحث والتثبيت ( Python PacKage)

لإدارة المكتبات في المشروع Python Interpreter

امر لإلغاء تثبيت المكتبة Uninstall

## تعريف المكتباتLibraries:

هي عبارة عن مجموعة من التعليمات البرمجية المكتوبة مسبقًا والتي يمكن استخدامها لأداء مهام محددة يتم استدعاؤها حسب الحاجة لتوفير الوقت والجهد للمبرمج لكي لا يعيد كتابتها مرة أخرى .

**مثـــال**

عند الرغبة في كتابة برنامج لحساب المتوسط الحسابي لمجموعة من الأرقام بدون استخدام أي مكتبات مساعدة سيكون التعليمات البرمجية مشابهًا للتالي:

**Input**

```
num = [2, 4, 1, 5, 9]
n = len(num)
get_sum = sum(num)
Avrg = get_sum / n
print("Average = ", Avrg)
```

الآن لو أردنا لاستعانة بإحدى المكتبات التي توفر دالة جاهزة لحساب المتوسط سنجد العديد منها مثل مكتبة NumPy ، سنستورد مكتبة NumPy في بداية الكود البرمجي بعدها سنتمكن من كتابة الكود المكافئ للكود السابق لحساب المتوسط الحسابي بتعليمات أقل كما يلي:

**Out put**

```
import numpy as numpy  # NumPy استيراد مكتبة
# تعريف قائمة من الأرقام
num = [2, 4, 1, 5, 9]
# NumPy من مكتبة average حساب المتوسط باستخدام دالة
Avrg = numpy.average(num)
# طباعة المتوسط الحسابي
print("Average = ", Avrg)
```

**ملاحظة**

قبل استيراد المكتبة يجب تثبيت المكتبة أولا بإحدى الطرق التالية.

# التعامل مع المكتبات الخارجيةThird-Party Libraries

## تثبيت المكتبات الخارجيةInstall Third-Party Libraries

### الطريقة الأولى

1- فتح المشروع فيPyCharm

2- من الشريط أسفل نافذة البرنامج PyCharm اختيار تبويب Terminal

3- يتم كتابة الأمر `pip install (library name)`

**Terminal screen**

```
Terminal: Local
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

(.venv) PS C:\Users\HP\Documents\pythonProjectnstall> pip install NumPy
Collecting NumPy
  Obtaining dependency information for NumPy from https://files.pythonhosted.org/packages/a6/84/fa11dad3404b7634aaab50733581ce11e5350383311ea7a7010f464c0170/numpy-2.1.3-cp312-cp312-win_amd64.whl.metadata
  Downloading numpy-2.1.3-cp312-cp312-win_amd64.whl.metadata (60 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 60.8/60.8 kB 404.8 kB/s eta 0:00:00
Downloading numpy-2.1.3-cp312-cp312-win_amd64.whl (12.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.6/12.6 MB 25.2 MB/s eta 0:00:00
Installing collected packages: NumPy
Successfully installed NumPy-2.1.3

[notice] A new release of pip is available: 23.2.1 -> 24.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS C:\Users\HP\Documents\pythonProjectnstall>
Version Control  TODO  Problems  Terminal  Python Packages  Services
```

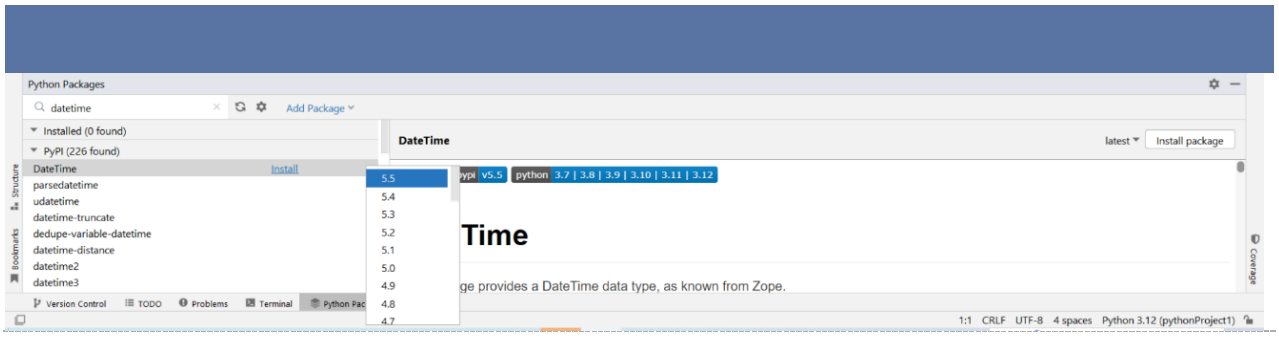
### الطريقة الثانية

1- فتح المشروع في **PyCharm** .

2- من الشريط أسفل نافذة **PyCharm** اختيار تبويب **Python PacKage**

3- البحث عن المكتبة المناسبة في خانة البحث.

4- بعد ظهور نتيجة البحث يتم تحديد اسم المكتبة واختيار الإصدار المناسب ثم تثبيتها .

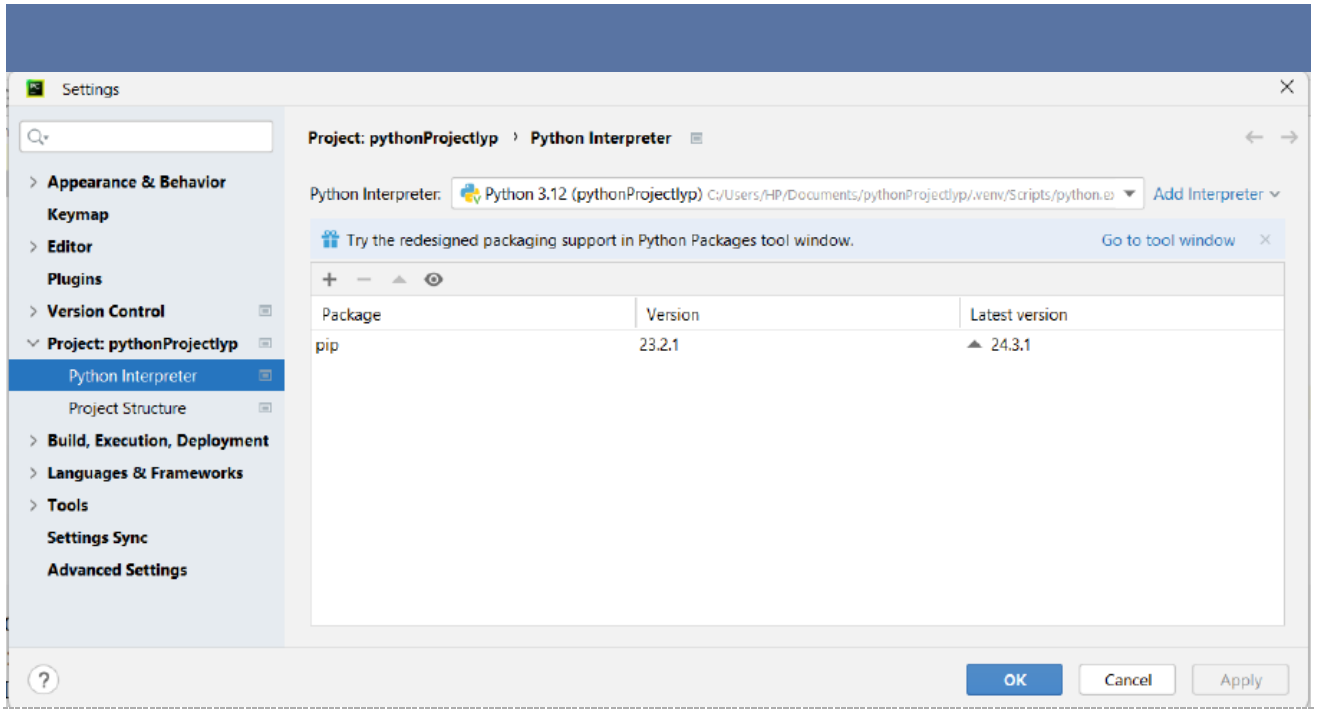


# تابع: التعامل مع المكتبات الخارجيةThird-Party Libraries

## الطريقة الثالثة

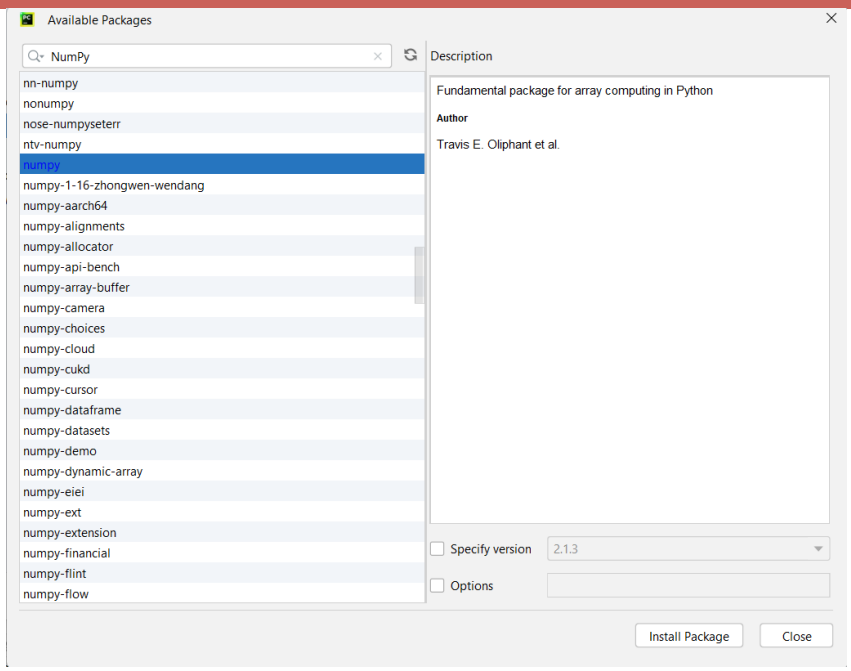
1- اختيار الأمر **Settings** من قائمة **File** .

2- اختيار أمر **Project**( **اسم المشروع** ) من القائمة الجانبية، ثم تحديد **Python Interpreter**.



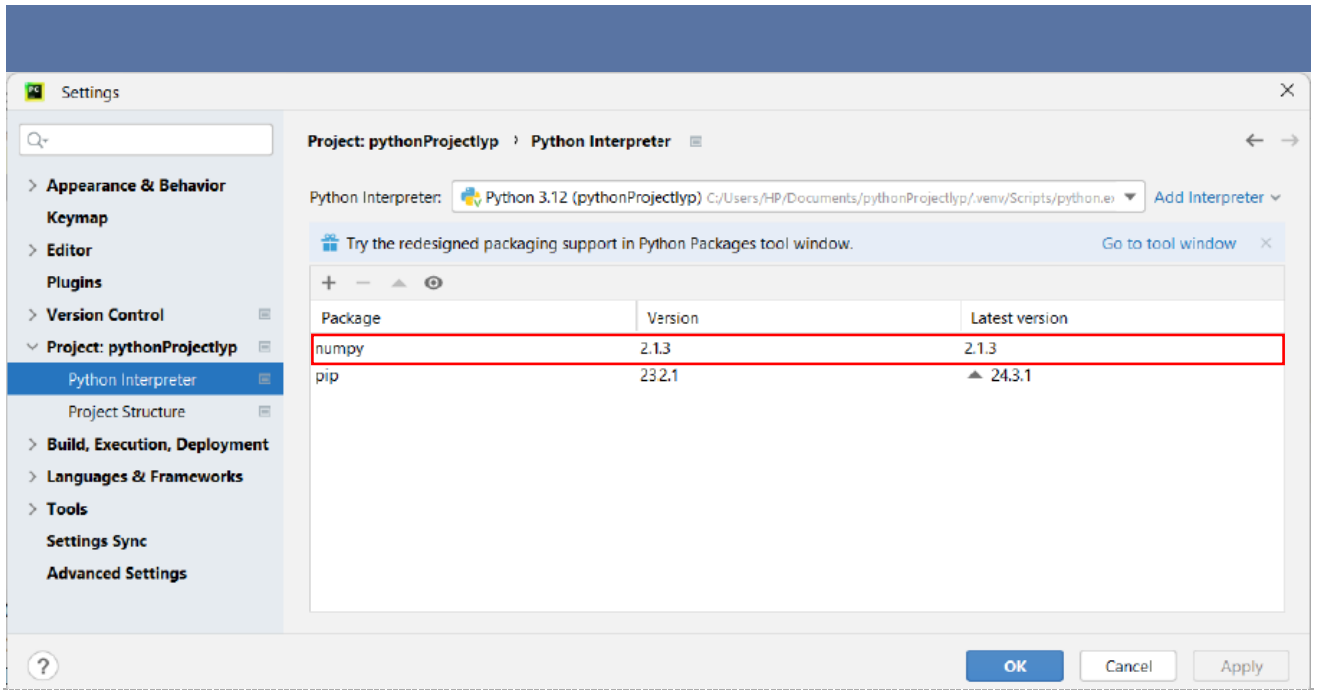

3. تحديد زر (+) في الجانب الأيمن لفتح نافذة **Available PacKages**

Available PacKages



## تابع: التعامل مع المكتبات الخارجيةThird-Party Libraries

4. كتابة اسم المكتبة "NumPy" في شريط البحث وستظهر في قائمة النتائج.
5. بمجرد العثور على المكتبة المطلوبة، يتم الضغط على . Install PacKage



# إلغاء تثبيت المكتبات الخارجيةuninstall Third-Party Libraries

## الطريقة الأولى :

1. من الشريط أسفل نافذة البرنامج **PyCharm** اختيار تبويب **Terminal**.
2. كتابة أمر إلغاء التثبيت الأمر **pip uninstall** متبوعًا باسم المكتبة التي تريد إلغاء تثبيتها

```
pip uninstall (library name)
```

مثال :

```
Terminal: Local
[notice] A new release of pip is available: 23.2.1 -> 24.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS C:\Users\HP\Documents\pythonProjectnstall> pip uninstall NumPy
Found existing installation: numpy 2.1.3
Uninstalling numpy-2.1.3:
  Would remove:
    c:\users\hp\documents\pythonprojectnstall\.venv\lib\site-packages\numpy-2.1.3-cp312-cp312-win_amd64.whl
    c:\users\hp\documents\pythonprojectnstall\.venv\lib\site-packages\numpy-2.1.3.dist-info\*
```

تأكيد إلغاء التثبيت اضغط **y** ثم **Enter** للتأكيد

```
Terminal: Local
    c:\users\hp\documents\pythonprojectnstall\.venv\lib\site-packages\numpy.libs\libscipy_openblas64_-c16e4918366c6bc1f1cd71e28ca36fc0
.dll
    c:\users\hp\documents\pythonprojectnstall\.venv\lib\site-packages\numpy.libs\msvcp140-d64049c6e3865410a7dda6a7e9f0c575.dll
    c:\users\hp\documents\pythonprojectnstall\.venv\lib\site-packages\numpy\*
    c:\users\hp\documents\pythonprojectnstall\.venv\scripts\f2py.exe
    c:\users\hp\documents\pythonprojectnstall\.venv\scripts\numpy-config.exe
Proceed (Y/n)? y
  Successfully uninstalled numpy-2.1.3
(.venv) PS C:\Users\HP\Documents\pythonProjectnstall>
```

## الطريقة الثانية :

1- اختيار الأمر **Settings** من قائمة **File**

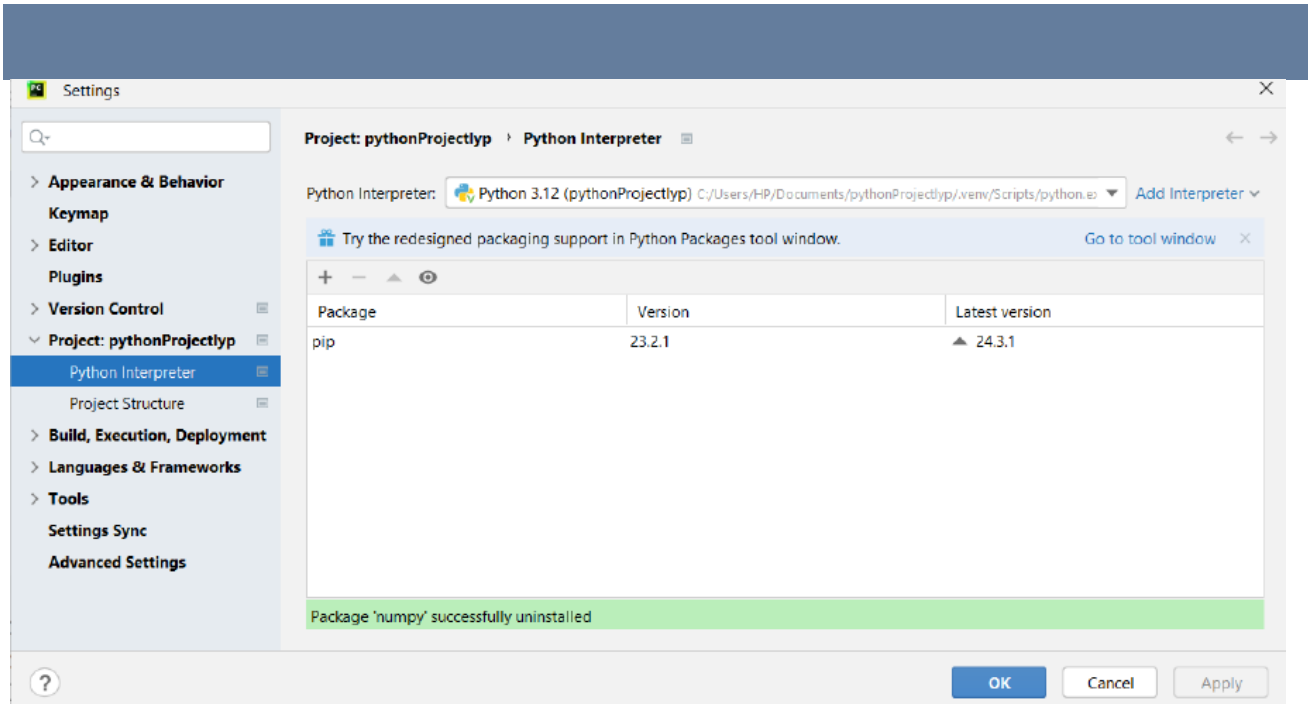
2- اختيار **Project**( **اسم المشروع** ) من القائمة الجانبية، ثم تحديد **Python Interpreter**

3- تظهر قائمة بجميع المكتبات المثبتة في المشروع يتم البحث عن المكتبة التي تريد إلغاء تثبيتها (مثل **NumPy**)

4- تحديد المكتبة، ثم الضغط على زر **Uninstall (-)** بجانب اسم المكتبة

5- تظهر رسالة تأكيد إلغاء التثبيت لإزالة المكتبة من المشروع.

## تابع -إلغاء تثبيت المكتبات الخارجيةuninstall Third-Party Libraries



# استيراد المكتباتImport Libraries

بعد تثبيت المكتبة باستخدام install لتفعيل المكتبة واستخدام الدوال والوظائف التي تحتوي عليها نحتاج إلى استيرادها عن طريق التعليمة برمجية .

import library name

## أولا : استيراد مكتبة كاملة

مثـــال 1 :

استيراد مكتبة **datetime** المخصصة للتعامل مع التاريخ والوقت.
في المثال التالي تم استيراد دالة التاريخ والوقت باستخدام الأمر **import**.
ثم أدرجنا التعليمات البرمجية التي تعرض كيفية الوصول إلى مكونات التاريخ (اليوم، الشهر، السنة)
باستخدام مكتبة **datetime**، مما يسهل التعامل مع التواريخ في البرامج.

ملاحظة : يجب ادراج اسم المكتبة أولا متبوعا باسم الدالة .

input

```
import datetime

print(datetime.date.today())          # طباعة التاريخ الكامل
print(datetime.date.today().day)      # طباعة اليوم
print(datetime.date.today().month)    # طباعة الشهر
print(datetime.date.today().year)     # طباعة السنة
```

Output

```
C:\Users\HP\Documents\pythonProject
2024-11-10
10
11
2024

Process finished with exit code 0
```

- عرض الوقت والتاريخ الحالي

input

```
import datetime

# الحصول على التاريخ والوقت الحالي
current_datetime = datetime.datetime.now()

print("التاريخ والوقت الحالي:", current_datetime)
```

Output

```
C:\Users\HP\Documents\pythonProjectlyp\.venv\Scripts
11:33:03.347206 10-11-2024 :التاريخ والوقت الحالي
```

# تابع: استيراد المكتباتImport Libraries

مثـــال 2 :

**استيراد مكتبة.**

مكتبة **random** من المكتبات المدمجة في **Python** ، وتساعد على العمل مع الأعداد العشوائية لتوليد عدد صحيح عشوائي بين **1 و10** سنتناول بعض دوالها بالامثلة التوضيحية :

## Random

توليد رقم عشوائي بين 0 و1

```
Import Random
Random-num=random.random()
Print (Random-num)
```

## Randint

الدالة **random.randint** هي دالة مخصصة لتوليد أعداد صحيحة بشكل عشوائي ضمن مجال محدد. وهي جزء من مكتبة **random** .

مثـــال 1 : برنامج "رمي النرد" افتراضيًا والحصول على رقم بين 1 و 6

**input**

```
import random

# توليد رقم عشوائي بين 1 و 6 (يمثل رمي النرد)
nrd = random.randint( a: 1,  b: 6)
print(":نتيجة رمي النرد", nrd)
```

**Output**

```
C:\Users\HP\Documents\pythonProjectlyp\.
5 :نتيجة رمي النرد

Process finished with exit code 0
```

مثـــال2 : توليد رقم عشوائي بين 1 و 100

**input**

```
import random

# توليد عدد عشوائي بين 1 و 100
random_number = random.randint( a: 1,  b: 100)

print(":العدد العشوائي هو", random_number)
```

# تابع: استيراد المكتباتImport Libraries

تابع -مثـــال2 : توليد رقم عشوائي بين 1 و 100

**Output**

```
C:\Users\HP\Documents\pythonProjectlyp\.
العدد العشوائي هو: 73

Process finished with exit code 0
```

مثـــال الكتاب ص 31 توليد رقم عشوائي بين 1 و 100.

**Input**

```
import random

random_num = random.randint( a: 1,  b: 10)  # توليد عدد عشوائي بين 1 و 10
print(f"The random integer is = {random_num}")  # طباعة العدد العشوائي

```

**Output**

```
C:\Users\HP\Documents\pythonProjectly
The random integer is = 9

Process finished with exit code 0
```

شرح المثـــال :

في البرنامج يتم **استيراد مكتبة** :random بكتابة التعليمة البرمجية :import random

**توليد عدد عشوائي باستخدام** : randint

- نستخدم الدالة randint(1, 10) من مكتبة random لتوليد عدد صحيح عشوائي بين 1 و10.
- يتم تخزين العدد العشوائي الناتج في المتغير random_num.

**طباعة العدد العشوائي باستخدام**: f-string

- print(f"The random integer is = {random_num}")

f-string لإدراج قيمة random_num داخل النص


- f-string هي طريقة سهلة ومباشرة لإدراج القيم المتغيرة داخل النصوص ،
- لإنشاء f-string، ادراج الحرف f قبل علامات التنصيص " " أو ' '. داخل النص ثم يمكنك وضع القيم المتغيرة بين {} لتظهر كجزء من النص.

## تابع: استيراد المكتباتImport Libraries


عند استدعاء مكتبة كاملة في Python باستخدام الأمر import، يجب **كتابة اسم المكتبة** متبوعًا بنقطة .قبل استخدام أي دالة من دوالها **تكتب للتأكد أنها موجودة في تلك المكتبة**.

### ثانيا : استيراد عنصر محدد (التعليمات البرمجية الدوال functions ، والفئات classes ، والمتغيرات variables ) من مكتبة محددة

يمكن استيراد عنصر محدد بدلاً من استيراد المكتبة كاملة مما يجعل التعليمة البرمجية أبسط وذات كفاءة حيث تستورد فقط ما يتم احتياجه من المكتبة.

مثـــال 1 : مثال الكتاب ص 31 توليد رقم عشوائي بين 1 و 100
استيراد الدالة asctime فقط، من مكتبة time ، والتعامل معها.

Input

```
# time استيراد دالة asctime فقط من مكتبة
from time import asctime

# ()asctime طباعة التاريخ والوقت الحالي باستخدام الدالة
print(f"The current date and time is = {asctime()}")
```

Output

```
C:\Users\HP\Documents\pythonProjectlyp\.venv\Scripts\python.exe
The current date and time is = Mon Nov 11 08:30:20 2024

Process finished with exit code 0
```

نستورد الدالة **asctime** فقط من مكتبة **time**، ولا نستورد المكتبة كاملة.
مما يسمح لنا استخدام **asctime()** مباشرة دون الحاجة لكتابة **time.asctime()**، مما يجعل الكود أكثر اختصارًا.

# تابع: استيراد المكتباتImport Libraries

مثـــال 2 :

استدعاء اكثر من دالة من نفس المكتبة.

شرح المثال 2 كتاب الطالب ص32 .

استدعاء دالتين محددتين(**randint** و **randrange** )من مكتبة random لتوليد أعداد صحيحة عشوائية.

**Input**

```
# random استيراد الدالتين randint و randrange فقط من مكتبة
from random import randint, randrange

random_num1 = randint( a: 1,  b: 10)  # توليد عدد صحيح عشوائي بين 1 و 10
random_num2 = randrange(10)   # توليد عدد صحيح عشوائي من 0 إلى 9

print("العدد العشوائي بين 1 و 10 هو:", random_num1)
print("العدد العشوائي بين 0 و 9 هو:", random_num2)
```

**Output**

```
C:\Users\HP\Documents\pythonProjectlyp
العدد العشوائي بين 1 و 10 هو: 4
العدد العشوائي بين 0 و 9 هو: 2

Process finished with exit code 0
```

لم نستورد المكتبة كاملة، بل نستورد الدالتين اللتين نحتاجهما فقط، مما يجعل التعليمات البرمجية كثر وضوحًا وكفاءة

## الفرق بين الدالة randint و randrange

| الدالةrandint (start, end) | الدالة randrange (end) |
|---|---|
| random_num1 = randint(1, 10)<br><br>تُعيد عددًا صحيحًا عشوائيًا بين القيمتين **start** و **end** ، (يشمل العددين **1** و**10**)<br><br>النتيجة قد تكون أي رقم من الأرقام التالية: 1، 2، 3، ...، 10. | random_num2 = randrange(10)<br><br>**randrange**(10)ستُعيد عددًا عشوائيًا بين 0 والقيمة داخل القوس 9 (يشمل 0 ويستثني 10)<br><br>النتيجة قد تكون أي رقم من الأرقام التالية: 0، 1، 2، ...، 9. |

## ثالثا : استيراد المكتبة والتعامل معها باسم مختصر

Import library name as shortcut name

يمكن التعبير عن اسم المكتبة بشكل مختصر لسهولة التعامل معها

مـثال: التعبير عن اسم المكتبة **datetime** باسم مختصر **dt**.

Input

```
import datetime as dt

print(dt.date.today())          # طباعة التاريخ الكامل
print(dt.date.today().day)      # طباعة اليوم
print(dt.date.today().month)    # طباعة الشهر
print(dt.date.today().year)     # طباعة السنة
```

## رابعا : استيراد جميع الكائنات(مثل الدوال، المتغيرات، الفئات) من مكتبة معينة.

استيراد جميع الكائنات (مثل الدوال، الفئات، التعاون) من المكتبة مثال:

datetime باستخدام الرمز ( * ) ثم استخدمنا **date.today()** للحصول على تاريخ مع تفاصيله (اليوم، الشهر، السنة) وطباعة كل منها بشكل منفصل.

From library name import *

Input

```
from datetime import *  # استيراد جميع العناصر من مكتبة datetime

print(date.today())           # طباعة التاريخ الكامل بصيغة YYYY-MM-DD
print(date.today().day)       # طباعة اليوم فقط كعدد صحيح
print(date.today().month)     # طباعة الشهر فقط كعدد صحيح
print(date.today().year)      # طباعة السنة فقط كعدد صحيح
```

Output

```
C:\Users\HP\Documents\pythonProjectlyp\
2024-11-12
12
11
2024

Process finished with exit code 0
```

## ثالثا : استيراد المكتبة والتعامل معها باسم مختصر

مثـــال إضافي : From Random import *

Input

```
# random استيراد جميع العناصر من مكتبة
from random import *

# استخدام الدالة randint لتوليد عدد صحيح عشوائي بين 1 و 10 مباشرةً
random_num1 = randint( a: 1,  b: 10)
print(":العدد العشوائي بين 1 و 10 هو", random_num1)

random_num2 = randrange(10)    # توليد عدد صحيح عشوائي من 0 إلى 9
print(":العدد العشوائي بين 0 و 9 هو", random_num2)

# اختيار حرف عشوائي من السلسلة النصية
random_char = choice("python")
print(":الحرف العشوائي المختار هو", random_char)

# استخدام الدالة random لتوليد عدد عشري عشوائي بين 0 و 1
random_num2 = random()
print(":العدد العشري العشوائي بين 0 و 1 هو", random_num2)
```

Output

```
C:\Users\HP\Documents\pythonProjectlyp\.venv\Scripts\python.exe
العدد العشوائي بين 1 و 10 هو: 10
العدد العشوائي بين 0 و 9 هو: 6
الحرف العشوائي المختار هو: t
العدد العشري العشوائي بين 0 و 1 هو: 0.8845971067438608

Process finished with exit code 0
```

1. استيراد جميع العناصر من مكتبة: random
   from random import *:
   - القصد استيراد جميع العناصر (الدوال والمتغيرات والفئات) من مكتبة random.
2. توليد عدد صحيح عشوائي بين 1 و 10 باستخدام: randint
   - random_num1 = randint(1, 10):
   - الدالة randint(start, end) تعرض عددًا صحيحًا عشوائيًا شامل القيمتين.
   - كل مرة تُشغّل فيها البرنامج، نحصل على عدد مختلف.
3. توليد عدد صحيح عشوائي بين 0 و 9 باستخدام: randrange
   - random_num2 = randrange(10):
   - الدالة randrange(end) تعرض عددًا صحيحًا عشوائيًا ولا يشمل العدد الأخير المذكور(end)
   - وهي دالة تفيد في عرض أعداد ضمن نطاق يبدأ من 0.

4. اختيار حرف عشوائي من السلسلة النصية - sequence باستخدام: choice
   - random_char = choice("python"):
     - الدالة choice(sequence) تختار عنصرًا عشوائيًا من sequence.
     - في هذا المثال، السلسلة النصية "python"تتكون من الأحرف: p, y, t, h, o, n
     - choice("python") يختار حرفًا عشوائيًا من هذه السلسلة النصية.
5. توليد عدد عشري عشوائي بين 0 و1 باستخدام: random
   - random_num3 = random():
     - الدالة random () تعرض عددًا عشريًا عشوائيًا (float) بين 0 و1 (يشمل 0 ولكنه لا يشمل 1).


الرمز( * AsterisK ) في لاستيراد والتعامل كامل محتوى المكتبة

# توثيق المكتبات

توثيق المكتبات هو دليل تفصيلي يوضح كيفية استخدام محتويات المكتبة بشكل صحيح، بما فيها ( الدوال، المتغيرات، والفئات)

يساعد التوثيق المطورين على فهم وظائف المكتبة وطرق استخدامها مما يسهل عليهم العمل مع مكتبات Python المختلفة.

## كيفية توثيق المكتبات:

1. استخدام أداة Terminal
   - الحصول على ملف توثيقي لأي مكتبة عن طريق استخدام الأمر ( pydoc )متبوعًا باسم المكتبة.
   - مثال :للحصول على توثيق مكتبة random.

     اكتب الأمر التالي في: Terminal ( pydoc random )

```
Terminal:  Local  +
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

(.venv) PS C:\Users\HP\Documents\pythonProjectlyp> pydoc random
```

   - سيظهر التوثيق الكامل لمكتبة random ، مع شرح لكل دالة ووظيفتها .

```
Terminal:  Local  +
Help on module random:

NAME
    random - Random variable generators.

MODULE REFERENCE
    https://docs.python.org/3.12/library/random.html

    The following documentation is automatically generated from the Python
    source files.  It may be incomplete, incorrect or include features that
    are considered implementation detail and may vary between Python
    implementations.  When in doubt, consult the module reference at the
    location listed above.

-- More  --
```

# تابع :توثيق المكتبات

يمكنك الضغط على مفتاح Ctrl والضغط على اسم المكتبة باستخدام الزر الأيسر للفأرة
أو الضغط Enter بجانب -- More -- في أداة terminal

```
Terminal:  Local  +
    The following documentation is automatically generated from the Python
    source files.  It may be incomplete, incorrect or include features that
    are considered implementation detail and may vary between Python
    implementations.  When in doubt, consult the module reference at the
    location listed above.

DESCRIPTION
        bytes
        -----
               uniform bytes (values between 0 and 255)

        integers
        --------
               uniform within range

        sequences
        ---------
               pick random element
               pick random sample
               pick weighted random sample
               generate random permutation

        distributions on the real line:
        ------------------------------
               uniform
               triangular
        distributions on the real line:
        ------------------------------
               uniform
               triangular
               normal (Gaussian)
               lognormal
               negative exponential
               gamma
               beta
               pareto
               Weibull

        distributions on the circle (angles 0 to 2pi)
        ---------------------------------------------
               circular uniform
               von Mises

        discrete distributions
        ----------------------
               binomial


    General notes on the underlying Mersenne Twister core generator:

    * The period is 2**19937-1.
    * It is one of the most extensively tested generators in existence.
```

# تابع :توثيق المكتبات

ورقة عمل 1 ص 34

اسم البرنامج: توليد كلمة مرور قوية **Strong Password**

مشكلة البرنامج: توليد كلمة مرور مركبة من الأرقام، الأحرف، والرموز.

المطلوب: استدعاء الملف **gen_pasword_library.py** من مجلد أوراق العمل.

- سيظهر التوثيق الكامل لمكتبة **random** ، مع شرح لكل دالة ووظيفتها .

Output

```python
# مشروع توليد كلمة مرور قوية
# استيراد مكتبة random لتوليد عناصر عشوائية
import random

# قوائم تحتوي على الأحرف، الأرقام، والرموز التي سيتم استخدامها في كلمة المرور
letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p']
numbers = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']
symbols = ['-', '@', '&', '*', '-']

# طباعة رسالة ترحيبية للمستخدم
print("Welcome to the PyPassword Generator!")

# استقبال عدد الأحرف والرموز والأرقام المطلوبة من المستخدم
gen_letters = int(input("How many letters would you like in your password?: "))
gen_symbols = int(input("How many symbols would you like?: "))
gen_numbers = int(input("How many numbers would you like?: "))

# تهيئة متغير كلمة المرور ليكون فارغًا، وسنقوم بإضافة الأحرف العشوائية إليه لاحقًا
password = ""

# إضافة الأحرف العشوائية إلى كلمة المرور بناءً على العدد المطلوب
for char in range(1, gen_letters + 1):
    password += random.choice(letters)  # اختيار حرف عشوائي من قائمة الأحرف وإضافته إلى كلمة المرور

# إضافة الرموز العشوائية إلى كلمة المرور بناءً على العدد المطلوب
for char in range(1, gen_symbols + 1):
    password += random.choice(symbols)  # اختيار رمز عشوائي من قائمة الرموز وإضافته إلى كلمة المرور

# إضافة الأرقام العشوائية إلى كلمة المرور بناءً على العدد المطلوب
for char in range(1, gen_numbers + 1):
```


# إضافة الأرقام العشوائية إلى كلمة المرور بناءً على العدد المطلوب
for char in range(1, gen_numbers + 1):


```python
    password += random.choice(numbers)  # اختيار رقم عشوائي من قائمة الأرقام وإضافته إلى كلمة المرور

# طباعة كلمة المرور النهائية للمستخدم
print("Password is:", password)
```

# تابع :توثيق المكتبات


- التأكيد على استيراد مكتبة random، وكونها توفر دوالًا لتوليد عناصر عشوائية مثل .choice()
- القوائم تحتوي على عناصر سيتم اختيارها عشوائيًا لإنشاء كلمة المرور
  - ( letters ) تحتوي أحرف.
  - ( numbers) تحتوي أرقام.
  - ( symbols) تحتوي رموز.

البرنامج يطلب من المستخدم تحديد عدد كل من الأحرف والرموز و الأرقام التي تكون كلمة المرور.

input() : ستخدم للحصول على المدخلات.

و int() تحولها إلى عدد صحيح.

password ="" تهيئة متغير .

- حلقة لإضافة( letters ) أحرف - ( numbers) أرقام -( symbols) رموز ) العشوائية بناءً على العدد الذي تم تحديده من المستخدم
- يختار حرفا ورمزا ورقما عشوائيًا من قائمات الثلاث ويضيفه إلى .password

Input

```
for char in range(1, gen_letters + 1):
    password += random.choice(letters)

for char in range(1, gen_symbols + 1):
    password += random.choice(symbols)

for char in range(1, gen_numbers + 1):
    password += random.choice(numbers)
```

Output

```
C:\Users\HP\Documents\pythonProjectlyp\.venv\Scripts\python.exe
Welcome to the PyPassword Generator!
How many letters would you like in your password?: 3
How many symbols would you like?: 2
How many numbers would you like?: 2
Password is: kmn@@63

Process finished with exit code 0
```

# تابع :توثيق المكتبات

## تابع ورقة عمل (١) :

طور البرنامج ليتم توليد كلمة مرور مركبة لا يقل عدد عناصرها عن **8**، ولا يزيد عن **12**.

Input

```python
# مشروع توليد كلمة مرور قوية
# استيراد مكتبة random لتوليد عناصر عشوائية
import random

# قوائم تحتوي على الأحرف، الأرقام، والرموز التي سيتم استخدامها في كلمة المرور
letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p']
numbers = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']
symbols = ['-', '@', '&', '*', '-']

# طباعة رسالة ترحيبية للمستخدم
print("Welcome to the PyPassword Generator!")

# تكرار إدخال عدد الأحرف والرموز والأرقام حتى يكون العدد الكلي بين 8 و 12
while True:
    # استقبال عدد الأحرف، الرموز، والأرقام من المستخدم
    gen_letters = int(input("How many letters would you like in your password?: "))
    gen_symbols = int(input("How many symbols would you like?: "))
    gen_numbers = int(input("How many numbers would you like?: "))

    # حساب العدد الكلي للعناصر
    total_characters = gen_letters + gen_symbols + gen_numbers

    # التحقق من أن العدد الكلي بين 8 و 12
    if 8 ≤ total_characters ≤ 12:
        break  # إذا كان العدد الكلي مناسبًا، نخرج من الحلقة
    else:
        print("The total number of characters must be between 8 and 12. Please try again.")

# تهيئة متغير كلمة المرور ليكون فارغًا، وسنقوم بإضافة الأحرف العشوائية إليه لاحقًا
password = ""

# إضافة الأحرف العشوائية إلى كلمة المرور بناءً على العدد المطلوب
for char in range(gen_letters):
    password += random.choice(letters)  # اختيار حرف عشوائي من قائمة الأحرف وإضافته إلى كلمة المرور

# إضافة الرموز العشوائية إلى كلمة المرور بناءً على العدد المطلوب
for char in range(gen_symbols):
    password += random.choice(symbols)  # اختيار رمز عشوائي من قائمة الرموز وإضافته إلى كلمة المرور

# إضافة الأرقام العشوائية إلى كلمة المرور بناءً على العدد المطلوب
for char in range(gen_numbers):
    password += random.choice(numbers)  # اختيار رقم عشوائي من قائمة الأرقام وإضافته إلى كلمة المرور

# طباعة كلمة المرور النهائية للمستخدم
print("Password is:", password)
```

أضفنا حلقة **while** لتكرار المدخلات عدد عناصر (أحرف + رموز + أرقام) يتراوح بين **8 و 12**.

إذا كان العدد الكلي للعناصر خارج النطاق (أقل من 8 أو أكثر من 12) يظهر رسالة للمستخدم ويطلب إعادة الإدخال.

```python
# حساب العدد الكلي للعناصر
total_characters = gen_letters + gen_symbols + gen_numbers
```

# تابع :توثيق المكتبات

ورقة عمل 2
ص 35

اسم البرنامج: بنك الأسئلة

مشكلة البرنامج: توظيف المكتبات لإظهار أسئلة عشوائية، واستقبال الأجوبة، وحساب النتيجة النهائية.

المطلوب: استدعاء الملف questions_bank.py من مجلد أوراق العمل.

Input

```python
# برنامج بنك الأسئلة
# هذا البرنامج يستخدم مكتبة random لاختيار أسئلة عشوائية من قائمة أسئلة، ثم يعرضها للمستخدم، يستقبل الإجابات، ويحسب النتيجة النهائية

# استيراد المكتبات اللازمة
import random  # مكتبة random لتوليد أرقام عشوائية
import datetime  # مكتبة datetime لعرض التاريخ الحالي
# قائمة الأسئلة، تحتوي على أسماء بعض الدوال في Python
questions = ["str()", "int()", "input()", "print()", "sum()", "del()", "exit()"]
# قائمة الإجابات، كل عنصر يمثل معنى الدالة في نفس فهرس السؤال المقابل في قائمة questions
answers = [ "Convert value to string", "Convert value to integer", "Enter value from user",
    "Display values on screen","Sum an integer values", "Delete variables", "Exit program" ]
# تهيئة درجة المستخدم لتكون صفر في البداية
score = 0
# عرض التاريخ الحالي للمستخدم في بداية البرنامج
print(f'Today Date: {datetime.datetime.today()}')

# حلقة لتكرار الأسئلة العشوائية حتى يتبقى سؤالان فقط في قائمة الأسئلة
while len(questions) > 2:
    # اختيار سؤال عشوائي من القائمة باستخدام فهرس عشوائي
    random_index = random.randrange(len(questions))
    user_question = questions[random_index]  # تخزين السؤال المختار
    # عرض السؤال المختار للمستخدم
    print(f"Question: What is the meaning of: {questions[random_index]} function")
    # عرض الخيارات المتاحة للمستخدم للإجابة عليها
    for i in range(len(answers)):
        print(f"{i+1}- {answers[i]}")  # طباعة كل خيار مع رقم تسلسلي
    # استقبال إجابة المستخدم وتحويلها إلى عدد صحيح
    user_answer = int(input("Choose the correct answer: "))
    # التحقق مما إذا كانت الإجابة صحيحة
    if (user_answer - 1) == random_index:
        score += 1  # زيادة درجة المستخدم بمقدار 1 إذا كانت الإجابة صحيحة
        print(f"Correct answer\n{'=' * 50}")  # طباعة رسالة توضح أن الإجابة صحيحة
    else:
        print(f"Wrong answer\n{'=' * 50}")  # طباعة رسالة توضح أن الإجابة خاطئة
    # إزالة السؤال والإجابة المقابلة له من القوائم لضمان عدم تكرار السؤال
    questions.pop(random_index)
    answers.pop(random_index)
# عرض النتيجة النهائية للمستخدم بعد انتهاء الأسئلة
print(f"Your score is: {score}")
```

- استيراد مكتبة **random** لتوليد أرقام عشوائية لاختيار سؤال عشوائي من القائمة.
- استيراد مكتبة **datetime** لعرض التاريخ الحالي عند بدء البرنامج.

## تعريف قائمة الأسئلة والإجابات:

- **questions** أسماء الدوال في **Python** ، والتي سيتم عرضها للمستخدم كجزء من الأسئلة.
- **answers** تفسيرات كل دالة (كل عنصر يطابق السؤال **questions** ) بنفس الترتيب.
- حلقة **while** لتكرار الأسئلة عشوائيًا حتى يتبقى سؤالان فقط في القائمة كل مرة يتم اختيار سؤال عشوائي.

# تابع :توثيق المكتبات

```
# حلقة لتكرار الأسئلة العشوائية حتى يتبقى سؤالان فقط في قائمة الأسئلة
while len(questions) > 2:
```

- عرض السؤال العشوائي باستخدام .questions[random_index]

```
    # اختيار سؤال عشوائي من القائمة باستخدام فهرس عشوائي
    random_index = random.randrange(len(questions))
    user_question = questions[random_index]  # تخزين السؤال المختار
```

- بعد ذلك، يتم عرض جميع الخيارات المتاحة للمستخدم باستخدام حلقة .for

```
    # عرض السؤال المختار للمستخدم
    print(f"Question: What is the meaning of: {questions[random_index]} function")
```

- يتم استقبال الإجابة من المستخدم ويحولها إلى int()

```
    # استقبال إجابة المستخدم وتحويلها إلى عدد صحيح
    user_answer = int(input("Choose the correct answer: "))
```

- إذا كانت الإجابة صحيحة، يتم زيادة score بمقدار (1) ويعرض رسالة"Correct answer" .
- إذا كانت خاطئة، يعرض رسالة"Wrong answer" .

ومن ثم حذف السؤال والإجابة من القائمة:

```
    # التحقق مما إذا كانت الإجابة صحيحة
    if (user_answer - 1) == random_index:
        score += 1  # زيادة درجة المستخدم بمقدار 1 إذا كانت الإجابة صحيحة
        print(f"Correct answer\n{'=' * 50}")  # طباعة رسالة توضح أن الإجابة صحيحة
    else:
        print(f"Wrong answer\n{'=' * 50}")  # طباعة رسالة توضح أن الإجابة خاطئة
    # إزالة السؤال والإجابة المقابلة له من القوائم لضمان عدم تكرار السؤال
    questions.pop(random_index)
    answers.pop(random_index)
```

## بعد دراسة التعليمات البرمجية واستكمال البيانات المطلوبة :

اسم المكتبات المستخدمة:

random: تُستخدم لتوليد أرقام عشوائية.

datetime: تُستخدم لعرض التاريخ الحالي.

## اسم الدوال المستخدمة ووظيفة كل منها:

1. input(): وظيفتها: تُستخدم لاستقبال إجابة المستخدم كمدخل نصي.
2. print() وظيفتها: تُستخدم لطباعة النصوص والرسائل على الشاشة.
3. len(): وظيفتها : تعد عدد العناصر في القائمة وعرفة عدد الأسئلة المتبقية في قائمة .questions.
4. int(): وظيفتها: تُحول المدخل النصي إلى عدد صحيح.
5. random.randrange() هي دالة في مكتبة random تُستخدم لتوليد عدد صحيح عشوائي ضمن نطاق محدد.
6. datetime.datetime() هي دالة في مكتبة datetime تُستخدم لعرض التاريخ والوقت.

# تابع :توثيق المكتبات- أمثلة إضافية :

يمكن للمعلم اختيار استكمال البرنامج الحالي وتوضيح تفاصيله، أو إنشاء برنامج جديد من البداية استنادًا إلى الخوارزمية، حسب مستوى واحتياجات المتعلمين.

## مثـــال ا:

### برنامج لعبة "اختيار رقم عشوائي "

فكرة البرنامج

البرنامج يختار رقم عشوائي بين 1 و20، ويطلب من المستخدم محاولة تخمين الرقم .بعد كل تخمين، يعرض البرنامج تلميحات للمستخدم حول ما إذا كان الرقم الذي أدخله أكبر أو أصغر من الرقم الصحيح.

في حالة إذا نجح المستخدم في تخمين الرقم، يظهر البرنامج رسالة تهنئة وعدد المحاولات التي قام بها.

خوارزمية البرنامج

1. استيراد مكتبة random لتوليد الأرقام العشوائية بين 1 و 20 ، (1, 20)وتخزينه في متغير .target_number
2. متغير (attempts = 0) لتتبع عدد محاولات المستخدم.
3. طباعة رسالة توضح للمستخدم أن هناك رقمًا بين 1 و 20، وعليه تخمينه.
4. بدء حلقة التخمين تستمر حتى يُدخل المستخدم الرقم الصحيح.
5. استقبل تخمين من المستخدم باستخدام input، وتحويله إلى عدد صحيح.
6. زيادة المتغير attempts بمقدار 1 لتسجيل عدد المحاولات.

التحقق إذا كان التخمين

- أقل من الرقم العشوائي:(target_number) - طباعة رسالة."Too low! Try a higher number."
- إذا كان التخمين أكبر من الرقم العشوائي: طباعة رسالة."Too high! Try a lower number."
- إذا كان التخمين مساويًا للرقم العشوائي:
  - طباعة رسالة تهنئة تحتوي على الرقم الصحيح وعدد المحاولات.
  - إنهاء الحلقة باستخدام .breaK
- ينتهي البرنامج عند تخمين المستخدم للرقم الصحيح.

## تابع :توثيق المكتبات- أمثلة إضافية :

### Input

```python
import random  # استيراد مكتبة random لتوليد رقم عشوائي

# توليد رقم عشوائي بين 1 و 20
target_number = random.randint(a: 1,  b: 20)

# تهيئة عدد المحاولات
attempts = 0

# رسالة ترحيبية
print("Welcome to the Guessing Game!")
print("I have chosen a number between 1 and 20. Can you guess it?")

# حلقة التخمين
while True:
    # استقبال تخمين المستخدم وزيادة عدد المحاولات
    user_guess = int(input("Enter your guess: "))
    attempts += 1

    # التحقق من التخمين وتقديم تلميحات
    if user_guess < target_number:
        print("Too low! Try a higher number.")
    elif user_guess > target_number:
        print("Too high! Try a lower number.")
    else:
        # إذا كان التخمين صحيحًا
        print(f"Congratulations! You guessed the number {target_number} correctly in {attempts} attempts.")
        break  # الخروج من الحلقة عند التخمين الصحيح
```

### Output

```
C:\Users\HP\Documents\pythonProjectlyp\.venv\Scripts\python.exe C:\Use
Welcome to the Guessing Game!
I have chosen a number between 1 and 20. Can you guess it?
Enter your guess: 5
Too low! Try a higher number.
Enter your guess: 7
Too low! Try a higher number.
Enter your guess: 17
Congratulations! You guessed the number 17 correctly in 3 attempts.

Process finished with exit code 0
```

# تابع :توثيق المكتبات- أمثلة إضافية :

## مثـــال 2:

### برنامج " بنك الأسئلة "

### فكرة البرنامج

البرنامج يستخدم مكتبة random لتوليد أسئلة عشوائية من قائمة أسئلة ويعرضها للمستخدم، ثم يستقبل الأجوبة منه ويحسب النتيجة النهائية بناءً على الإجابات الصحيحة.

البرنامج يتعامل مع قائمة من الأسئلة والإجابات ويرتب الأجوبة لكل سؤال، ثم يعطي المستخدم النتيجة في النهاية.

### الخوارزمية

1. استيراد مكتبة **random** لتوليد أرقام عشوائية.
2. استيراد مكتبة **datetime** لعرض التاريخ الحالي.
3. تعريف قائمة **questions** التي تحتوي على الأسئلة.
4. تعريف قائمة **answers** التي تحتوي على الإجابات الصحيحة، يجب أن تتطابق كل عنصر في القائمة مع السؤال في نفس فهرس قائمة **questions**.
5. المتغير **score** ليبدأ من الصفر، وذلك حتى يتم تسجيل عدد الإجابات الصحيحة.
6. عرض التاريخ والوقت الحالي للمستخدم باستخدام **datetime.datetime.today()**.

حلقة الأسئلة العشوائية:

تكرار الخطوات التالية طالما أن عدد الأسئلة المتبقية في قائمة questionsأكبر من 2:

تحديد فهرس عشوائي:

مستخدمة التعليمة التالية: **random.randrange(len(questions))** لتوليد فهرس عشوائي لاختيار سؤال عشوائي من قائمة **questions** وإجابته من قائمة **answers**.

تخزين الفهرس العشوائي في المتغير **random_index**.

عرض السؤال للمستخدم:

عرض جميع الخيارات في قائمة **answers** لـ لمستخدم، مع ترقيم كل خيار ليختار الإجابة الصحيحة بسهولة.

استقبال إجابة المستخدم الذي يمثل إجابته، ثم تحويل المدخل إلى عدد صحيح **int ()**.

التحقق من صحة الإجابة:

إذا كانت إجابة المستخدم صحيحة

في حالة المساواة **(user_answer - 1) == random_index**

زيادة **score** بمقدار 1 يتم عرض رسالة **"Correct answer"** للمستخدم.

إذا كانت الإجابة خاطئة، يتم عرض رسالة **"Wrong answer"** للمستخدم.

7. حذف السؤال والإجابة المقابلة له من القائمتين **questions** و **answers** باستخدام pop()، لضمان عدم تكرار السؤال في الدورة القادمة.
8. بعد انتهاء الأسئلة، عرض النتيجة النهائية للمستخدم بطباعة عدد الإجابات الصحيحة في المتغير **score**.

## تابع :توثيق المكتبات- أمثلة إضافية :

Input

```python
# برنامج بنك الأسئلة
# هذا البرنامج يستخدم مكتبة random لاختيار أسئلة عشوائية من قائمة أسئلة، ثم يعرضها للمستخدم، يستقبل الإجابات، ويحسب النتيجة النهائية

# استيراد المكتبات اللازمة
import random  # مكتبة random لتوليد أرقام عشوائية
import datetime  # مكتبة datetime لعرض التاريخ الحالي
# قائمة الأسئلة، تحتوي على أسماء بعض الدوال في Python
questions = ["str()", "int()", "input()", "print()", "float()", "random()", "exit()"]
# قائمة الإجابات، كل عنصر يمثل معنى الدالة في نفس فهرس السؤال المقابل في قائمة questions
answers = [ "Convert value to string", "Convert value to integer", "Enter value from user",
    "Display values on screen","Convert value to a decimal number",
            "generate random numbers", "Exit program" ]
# تهيئة درجة المستخدم لتكون صفر في البداية
score = 0
# عرض التاريخ الحالي للمستخدم في بداية البرنامج
print(f'Today Date: {datetime.datetime.today()}')

# حلقة لتكرار الأسئلة العشوائية حتى يتبقى سؤالان فقط في قائمة الأسئلة
while len(questions) > 2:
    # اختيار سؤال عشوائي من القائمة باستخدام فهرس عشوائي
    random_index = random.randrange(len(questions))
    user_question = questions[random_index]  # تخزين السؤال المختار
    # عرض السؤال المختار للمستخدم
    print(f"Question: What is the meaning of: {questions[random_index]} function")
    # عرض الخيارات المتاحة للمستخدم للإجابة عليها
    for i in range(len(answers)):
        print(f"{i+1}- {answers[i]}")  # طباعة كل خيار مع رقم تسلسلي
    # استقبال إجابة المستخدم وتحويلها إلى عدد صحيح
    user_answer = int(input("Choose the correct answer: "))
    # التحقق مما إذا كانت الإجابة صحيحة
    if (user_answer - 1) == random_index:
        score += 1  # زيادة درجة المستخدم بمقدار 1 إذا كانت الإجابة صحيحة
        print(f"Correct answer\n{'=' * 50}")  # طباعة رسالة توضح أن الإجابة صحيحة
    else:
        print(f"Wrong answer\n{'=' * 50}")  # طباعة رسالة توضح أن الإجابة خاطئة
    # إزالة السؤال والإجابة المقابلة له من القوائم لضمان عدم تكرار السؤال
    questions.pop(random_index)
    answers.pop(random_index)
# عرض النتيجة النهائية للمستخدم بعد انتهاء الأسئلة
print(f"Your score is: {score}")
```

## تابع :توثيق المكتبات- أمثلة إضافية :

Output

```
1-2w
C:\Users\HP\Documents\pythonProjectlyp\.venv\Scripts\python.exe C:
Today Date: 2024-11-12 16:44:31.202774
Question: What is the meaning of: str() function
1- Convert value to string
2- Convert value to integer
3- Enter value from user
4- Display values on screen
5- Convert value to a decimal number
6- generate random numbers
7- Exit program
Choose the correct answer: 1
Correct answer
==================================================
Question: What is the meaning of: random() function
1- Convert value to integer
2- Enter value from user
3- Display values on screen
4- Convert value to a decimal number
5- generate random numbers
6- Exit program
Choose the correct answer: 5
Correct answer
==================================================
Question: What is the meaning of: exit() function
1- Convert value to integer
2- Enter value from user
3- Display values on screen
4- Convert value to a decimal number
5- Exit program
Choose the correct answer: 4
Wrong answer
==================================================
Question: What is the meaning of: float() function
1- Convert value to integer
2- Enter value from user
3- Display values on screen
4- Convert value to a decimal number
Choose the correct answer: 4
Correct answer
==================================================
Question: What is the meaning of: int() function
1- Convert value to integer
2- Enter value from user
3- Display values on screen
Choose the correct answer: 1
Correct answer
==================================================
Your score is: 4

Process finished with exit code 0
```

# تابع :توثيق المكتبات- أمثلة إضافية :

## مثـــال 3:

### فكرة برنامج: لعبة أسئلة

لعبة أسئلة تفاعلية يتم فيها طرح مجموعة من الأسئلة العشوائية على المستخدم، ويتم حساب النقاط بناءً على الإجابات الصحيحة يمكن للمستخدم الحصول على تغذية راجعة بعد كل إجابة لمعرفة ما إذا كانت صحيحة أو خاطئة، وفي نهاية اللعبة يتم عرض النتيجة النهائية.

### للمعلم حريه عرض المثال

التأكيد على اعداد خوارزمية بالمثال الإضافي السابق .

Input

```python
import random  # استيراد مكتبة random لاختيار الأسئلة عشوائيًا

# قائمة تحتوي على الأسئلة والأجوبة الصحيحة
questions = [
    {"question": "What is the capital of France?", "options": ["Paris", "London", "Rome",
                                                                "Berlin"], "answer": 1},
    {"question": "What is 5 + 7?", "options": ["10", "11", "12", "13"], "answer": 3},
    {"question": "Which planet is known as the Red Planet?", "options": ["Earth", "Mars",
                                                                          "Jupiter", "Saturn"],
     "answer": 2},
    {"question": "What is the largest ocean on Earth?", "options": ["Atlantic", "Indian",
                                                                      "Arctic", "Pacific"],
     "answer": 4},
    {"question": "Who wrote 'To Be, or Not to Be'?", "options": ["Shakespeare", "Dickens",
                                                                   "Hemingway", "Austen"],
     "answer": 1}
]
# تهيئة درجة المستخدم
score = 0

# رسالة ترحيبية
print("Welcome to the Quiz Game!")
print("Answer the following questions by choosing the correct option number.")

# اختيار 3 أسئلة عشوائية من القائمة
quiz_questions = random.sample(questions, k: 3)

# طرح الأسئلة المختارة على المستخدم
for question_data in quiz_questions:
    print("\n" + question_data["question"])  # عرض السؤال
    for i, option in enumerate(question_data["options"], 1):  # عرض الخيارات
        print(f"{i}. {option}")

    # استقبال إجابة المستخدم
    user_answer = int(input("Enter the option number of your answer: "))

    # التحقق من صحة الإجابة
    if user_answer == question_data["answer"]:
        print("Correct!")
        score += 1  # زيادة النقاط إذا كانت الإجابة صحيحة
    else:
        print("Wrong! The correct answer was:", question_data["options"][question_data["answer"] - 1])

# عرض النتيجة النهائية
print(f"\nYour final score is: {score} out of {len(quiz_questions)}")
```

## تابع :توثيق المكتبات- أمثلة إضافية :

Output

```
Welcome to the Quiz Game!
Answer the following questions by choosing the correct option number.

What is the capital of France?
1. Paris
2. London
3. Rome
4. Berlin
Enter the option number of your answer: 1
Correct!

Which planet is known as the Red Planet?
1. Earth
2. Mars
3. Jupiter
4. Saturn
Enter the option number of your answer: 2
Correct!

What is the largest ocean on Earth?
1. Atlantic
2. Indian
3. Arctic
4. Pacific
Enter the option number of your answer: 1
Wrong! The correct answer was: Pacific

Your final score is: 2 out of 3
```

# الوحدة الأولى: برمجة Python

# واجهة المستخدم الرسومية GUI

# Graphic User Interface

03

# واجهة المستخدم الرسوميةGUI - Graphic User Interface

## نتاجات التعلم " واجهة المستخدم الرسومية (GUI)"

| | |
|---|---|
| 1 | التعرف على مفهوم واجهة المستخدم الرسومية **(Graphical User Interface) (GUI)**: ودورها في تسهيل التفاعل بين المستخدم والتطبيقات. |
| 2 | التعرف على العناصر الأساسية لواجهة المستخدم الرسومية:<br>القدرة على التعامل مع العناصر الأساسية مثل النوافذ، الأزرار، القوائم، وعناصر الإدخال، وفهم كيفية تفاعل المستخدم معها. |
| 3 | فهم دور واجهة المستخدم الرسومية في تحسين تجربة الاستخدام ، مما يجعل التطبيقات أكثر سهولة. |
| 4 | القدرة على تصميم واجهة رسومية بسيطة. |
| 5 | فهم كيفية استخدام واجهة المستخدم الرسومية في معالجة وعرض المدخلات. |
| 6 | التعرف على تطبيقات وأمثلة عملية لواجهات المستخدم الرسومية:<br>- التعرف على بعض التطبيقات الشائعة التي تعتمد على واجهات المستخدم الرسومية. |
| 7 | التعرف على مزايا الواجهة الرسومية (GUI) مثل سهولة الاستخدام وسرعة التعلم وجاذبية التصميم |
| 8 | التعرف على أمثلة لبعض مكتبات تطوير واجهة المستخدم الرسومية في Python مثل **TKinter**، **PyQt**، و **Kivy** . |
| 9 | دراسة خطوات تصميم الواجهة الرسومية بدءًا من التخطيط ووضع المخطط وصولاً إلى كتابة الكود وتصدير البرنامج. |
| 10 | إضافة مكتبة **ICT_KW** إلى المشروع واستدعاؤها. |
| 11 | تعلم كيفية دمج مكتبة **ICT_KW** في المشروع واستدعاء الدوال الخاصة بها بفعالية . |
| 12 | التعامل مع عناصر واجهة الرسومية (**Widgets**) المتوفرة في مكتبة **ICT_KW**: |
| 13 | اكتساب مهارات ضبط خصائص العناصر الرسومية (**Widgets**) من مكتبة **ICT_KW** بما يناسب مع تصميم التطبيق. |
| 14 | تنفيذ مشاريع عملية تبرز فائدة توظيف الواجهة الرسومية في تبسيط تفاعل المستخدم مع التطبيقات. |
| 15 | تحفيز المتعلم للبحث عن مكتبات Python المساعدة في تصميم واجهات رسومية متقدمة وتحسين مهاراته. |

## المصطلحات الحاسوبية: واجهة المستخدم الرسومية (GUI)".

واجهة المستخدم الرسومية (GUI)
Graphical User Interface

الوحداتModules

الدوال Functions

البيانات غير الأولية Non-Primitive Data

أنظمة التشغيل Operating Systems

متصفحات الإنترنت Web Browser

مشغلات الوسائط Media Players

تطبيقات الهواتف الذكية Mobile Applications

حزمة تطبيقات Office Suites

الألعابGames.

عناصر Widgets

PyQt مكتبة لإنشاء واجهات المستخدم الرسومية (GUI) في Python.

مكتبة wxPython لتصميم واجهات البرامج.

مكتبة Kivy تطوير تطبيقات الأجهزة المحمولة وبرامج التطبيقات متعددة اللمس

## تابع- المصطلحات الحاسوبية: واجهة المستخدم الرسومية (GUI)".

مكتبة **Flask/Django**. لإنشاء تطبيقات الويب

مكتبة **PySide** لإنشاء تطبيقات سطح المكتب

الموقع **Location**

عنوان **Label**

الصورة **Image**

# ما هي واجهة المستخدم الرسومية (GUI)؟

واجهة المستخدم الرسومية (GUI)، أو Graphical User Interface، هي طريقة تفاعلية تسمح للمستخدمين بالتفاعل مع التطبيقات عبر "عناصر رسومية مرئية" .

ملاحظة للمعلم

التأكيد على ماهية الواجهة الرسومية بأنها ترجمة لكتابة أوامر نصية، تمكن المستخدم من التعامل مع المكونات الرسومية بدل الأكواد البرمجية

## نشاط : يوضح أهمية واجهات المستخدم الرسومية(GUI)

توضيح أهمية واجهات المستخدم الرسومية من خلال أمثلة عملية وتطبيقات تفاعلية تُظهر كيف تجعل GUI التطبيقات أكثر سهولة وجاذبية .

1. يبدأ المعلم بتقديم تعريف لواجهة المستخدم الرسومية(GUI) ، وشرح أنها واجهة تعتمد على عناصر مرئية يمكن للمستخدمين التفاعل معها مباشرة، مثل النوافذ، الأزرار، والقوائم.
2. يقارن المعلم عن مدى تفضيل المتعلمين بين استخدام واجهة رسومية أو كتابة الأوامر يدويًا، ويشجعهم على التفكير في سبب اختيارهم للواجهة الرسومية.
3. يعرض المعلم صورًا أو نماذج لتطبيقات مشهورة تعتمد علىGUI ، مثل Microsoft teams وحزمة التطبيقات المكتبية(Word) يوضح المعلم كيف تجعل هذه الواجهات البرامج أكثر سهولة وسرعة في الاستخدام.
4. من خلال عرض فيديو مختصر يوضح كيفية استخدام واجهة سطر الأوامر (Command Line) لأداء مهمة ما وآخر يستخدم واجهة رسومية (GUI) للقيام بنفس المهمة.

## مكونات واجهة المستخدم الرسوميةGUI

- النوافذ :إطار يظهر للمستخدم يحوي جميع مكونات التطبيق.
- الأزرار : عنصر يمكن للمستخدم الضغط عليه لتنفيذ أمر معين.
- مربع نص (حقل ادخال) : عنصر عرض وإدخال القيم ( نص أو أرقام) من المستخدم
- القوائم : مجموعة من الخيارات التي يمكن للمستخدم الاختيار منها.

# تابع- واجهة المستخدم الرسومية (GUI)؟

## نشاط : يوضح أهمية مكونات واجهة المستخدم الرسومية(GUI)

- عرض صور لواجهات مستخدم رسومية من تطبيقات معروفة، مثل تطبيق الآلة الحاسبة على الهواتف ، الحاسوب، أو برنامج إدخال الرقم المدني.
- استخدام عناصر رسومية مرئية وكيفية التعامل مع كل مكون في التفاعل مع البرنامج.

## نشاط: تصميم واجهة رسومية بسيطة

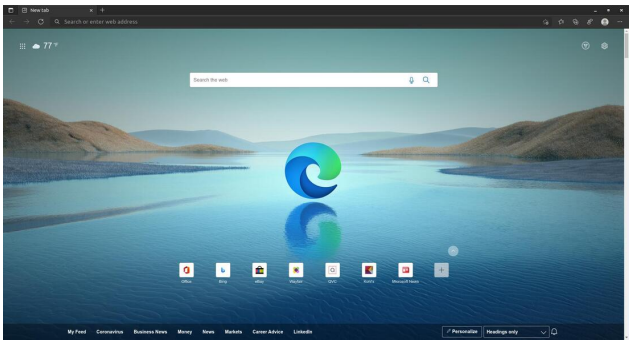
- يتم تقسيم المتعلمين إلى مجموعات صغيرة، بحيث يتعاون كل فريق على تصميم واجهة بسيطة لتطبيق معين مثل :
  - آلة حاسبة بسيطة.
  - نموذج إدخال بيانات (مثل إدخال اسم وعمر وعنوان).
  - برنامج للاستعلام عن تاريخ الميلاد بناءً على إدخال رقم مدني.
- الطلب من كل مجموعة تحديد المكونات التي تحتاجها، مثل:
  - نافذة رئيسية تحتوي على جميع العناصر.
  - حقول إدخال للنصوص أو الأرقام.
  - أزرار لتنفيذ العمليات.

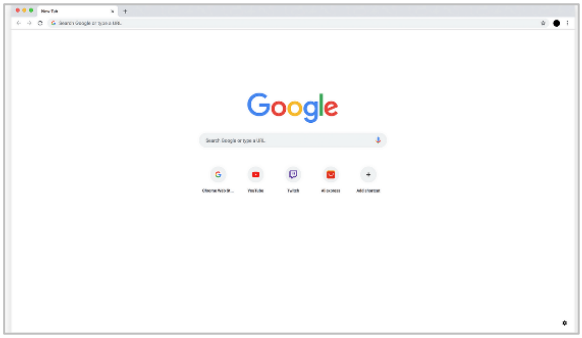
(يمكن لكل مجموعة رسم واجهة البرنامج على ورقة أو استخدام أداة تصميم واجهات رسومية بسيطة )

- بعد الانتهاء، تقوم كل مجموعة بعرض تصميمها أمام باقي المتعلمين وشرح كيف تعمل الواجهة.
- توجيه المجموعات لتناقش فيما بينها كيفية تحسين التصاميم بناءً على ملاحظات الجميع.

أبرز أمثلة واجهات المستخدم الرسومية( أنظمة تشغيل- متصفحات انترنت وغيرها)


## تابع- أبرز أمثلة واجهات المستخدم الرسومية






توضيح مثـــال الكـتاب ص 42

برنامج بسيط يطلب من المستخدم إدخال الرقم المدني ليستخرج منه تاريخ الميلاد، عند تشغيل البرنامج **تظهر النافذة التالية**:


### أنواع العناصر في واجهة المستخدم الرسومية

-النافذة الرئيسية: هي الواجهة الأساسية أو الاطار المتضمن لجميع العناصر الأخرى في البرنامج، تُستخدم لعرض واجهة التطبيق بالكامل.

- عنصر إدخال القيم: يُستخدم هذا العنصر لإدخال بيانات نصية أو رقمية يحتاجها البرنامج للقيام بوظيفة معينة.، مثل حقل إدخال الرقم المدني في هذا المثال.

- عنصر لتنفيذ أمر عند الضغط عليه: زر يتفاعل معه المستخدم للقيام بعملية معينة عند الضغط عليه. في هذا المثال، الزر مكتوب عليه "اعرض تاريخ ميلادك" وعند الضغط عليه يعرض تاريخ الميلاد بناءً على الرقم المدني المدخل.

-عنصر لعرض القيم: عرض النتيجة أو القيمة الناتجة بناءً على البيانات المدخلة، في هذا المثال، يظهر تاريخ الميلاد المستخرج من الرقم المدني ، ليتمكن المستخدم رؤية النتيجة مباشرة.

# أهمية الواجهات الرسوميةGUI

نشاط: توضيح أهمية الواجهة الرسومية

- يُقسم الطلاب إلى مجموعات ويُطلب من مجموعة استخدام واجهة سطر الأوامر (**Command Line**) لأداء مهمة بسيطة مثل إنشاء ملف جديد، بينما تستخدم مجموعة أخرى واجهة رسومية (**GUI**) للقيام بنفس المهمة.
- بعد التجربة، يتم النقاش حول الفروقات وأيهما كان أكثر سهولة وسرعة.

- شرح أهمية GUI بناءً على النقاط التالية:

- **سهولة الاستخدام** :يعرض المعلم كيف يمكن للمستخدمين التفاعل مع البرامج بسهولة عبر الضغط على الأزرار بدلاً من كتابة الأوامر.
- **تعزيز الإنتاجية** :يوضح أن GUI توفر عناصر مرئية مثل الأزرار وأيقونات الملفات، مما يسهم في تسهيل وسرعة أداء المهام.
- **سرعة التعلم** :يشرح المعلم كيف يمكن للمتعلمين الجدد تعلم استخدام التطبيقات ذات الواجهة الرسومية دون الحاجة لتدريب مكثف.
- **جاذبية التصميم** :يعرض أمثلة على تصاميم واجهات رسومية جذابة (مثل تطبيقات الهواتف)، ويشرح كيف تساهم الألوان والأيقونات في تحسين تجربة المستخدم وجذب الانتباه.
- **التفاعل الفوري** :يُبرز المعلم مدى استجابة الواجهات الرسومية بسرعة لأفعال المستخدم، مما يسهل فهم تأثير أفعاله عبر تلميحات بصرية مثل تغيير الألوان عند الضغط على زر.
- **إمكانية الوصول** :يناقش المعلم كيف يمكن تخصيص الواجهات لتلائم ذوي الاحتياجات الخاصة، مما يجعل التكنولوجيا شاملة للجميع.
- **دعم المهام المتعددة** :يعرض كيفية استخدام **GUI** لفتح عدة نوافذ وتنظيمها، مما يتيح للمستخدمين التنقل بسهولة بين المهام.

- يلخص المعلم الدرس بتوضيح أهمية GUI في الحياة اليومية ودورها في تحسين تجربة المستخدم وسهولة التفاعل.

# تابع-أهمية الواجهات الرسوميةGUI

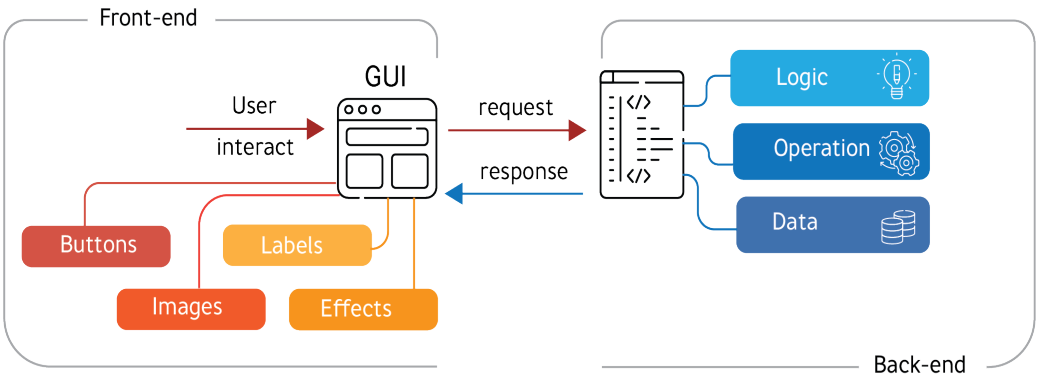
تدريب ص 45 من الكتاب: مستعينًا بشبكة الانترنت، ابحث عن اثنين من المكتبات السابقة، واستكمل الجدول التالي:

| م | اسم المكتبة | الاستخدام |
|---|---|---|
| 1 | TKinter | مكتبة مدمجة في **Python** لإنشاء واجهات رسومية بسيطة، توفر مجموعة من الأدوات مثل الأزرار، القوائم، وحقول الإدخال. |
| 2 | PyQt | مكتبة قوية تُستخدم لتطوير تطبيقات متقدمة ذات واجهات رسومية جذابة. تعتمد على مكتبة **Qt** وتوفر العديد من الأدوات الرسومية المتقدمة. |
| 3 | PySide | مكتبة مشابهة لـ **PyQt** لكنها مفتوحة المصدر وتابعة لـ **Qt**. تُستخدم أيضًا لتطوير تطبيقات متعددة المنصات توفر أدوات مرنة وقوية لإنشاء تطبيقات **GUI**. |
| 4 | FlasK/Django | مكتبات خاصة بتطوير تطبيقات الويب، وتُستخدم لإنشاء واجهات رسومية في متصفحات الإنترنت **FlasK**. مكتبة خفيفة ومناسبة للمشاريع الصغيرة، بينما **Django** إطار عمل شامل للمشاريع الكبيرة والمعقدة. |
| 5 | Kivy | مكتبة تُستخدم لتطوير تطبيقات متعددة المنصات تعمل على أجهزة الحاسوب والأجهزة المحمولة. تدعم اللمس المتعدد، ما يجعلها مناسبة لتطبيقات الأجهزة الذكية |
| 6 | wxPython | مكتبة تُستخدم لإنشاء تطبيقات بواجهة رسومية تعمل على أنظمة التشغيل المختلفة.تتميز بتوفير أدوات رسومية مرنة وتدعم العديد من المزايا، وتناسب التطبيقات الاحترافية. |

# توظيف المكتبات في الواجهة الرسومية

شرح مخطط الواجهة الرسومية للتوضيح للمتعلمين كيفية عمل الواجهتين معًا،

- مكونات الواجهة الرسومية (GUI) وعملية التفاعل بين Front-end وBack-endg




- الواجهة الرسومية (GUI) هي الجزء المرئي الذي يتفاعل معه المستخدم، حيث تحتوي على عناصر مرئية مثل الأزرار، الصور، النصوص، والتأثيرات البصرية.
- يُعرف هذا الجزء بالـ Front-end، وهو الواجهة الأمامية التي يتفاعل معها المستخدم مباشرة.

## مكونات الواجهة الأمامية:(Front-end)

- **Buttons – الأزرار** : عناصر تفاعلية يضغط عليها المستخدم لتنفيذ أوامر معينة.
- **Labels - التسميات** : تستخدم لعرض نصوص أو معلومات توضيحية ثابتة.
- **Images - الصور** : عناصر رسومية تستخدم لعرض الصور التي قد تكون لأغراض جمالية أو تعليمية.
- **Effects - التأثيرات** : تأثيرات مرئية مثل تغير الألوان عند الضغط ، مما يجعل تجربة الاستخدام أكثر جاذبية.

ملاحظة للمعلم

يطلق على ما سبق من الأدوات **widgets** وهي عبارة عن عناصر التحكم التي نضعها على النافذة ليتمكن المستخدم من التفاعل معها .

## كيفية التفاعل بين المستخدم والواجهة الرسومية:

- عندما يتفاعل المستخدم مع عناصر الواجهة (**مثل الضغط على زر** )، يتم إرسال طلب (**request**) إلى النظام لتنفيذ وظيفة معينة.

## الواجهة الخلفية:(BacK-end)

- بمجرد وصول الطلب إلى النظام، يتم التعامل معه في **الواجهة الخلفية(BacK-end)** ، التي تحتوي على:
  - **Logic - المنطق** : الجزء الذي يحدد القواعد والخطوات لتنفيذ العمليات.
  - **Operation - العمليات:** العمليات التي يتم تنفيذها استجابة لطلب المستخدم، مثل إجراء عملية حسابية أو استرجاع معلومات.
  - **Data - البيانات** : تخزين واسترجاع البيانات الضرورية، مثل استرجاع بيانات مستخدم معين.

### استجابة النظام وإرسال الرد:(response)

- بعد تنفيذ العمليات المطلوبة، يقوم النظام بإرسال رد (response) للواجهة الأمامية(Front-end) ، ويُظهر النتيجة أو المعلومات المطلوبة للمستخدم في الواجهة الرسومية.

### خطوات إنشاء الواجهة الرسومية:

- **تصور الشكل النهائي** :يرسم مخطط واجهة البرنامج الرسومية يوضح أماكن العناصر الرسومية التي سوف يتعامل معها المستخدمون.
- **تحديد العناصر** :(Widgets) ، اختيار الأدوات الرسومية المناسبة (الأزرار وحقول الإدخال) لبناء المخطط الرسومي ، ورسم مخطط هيكلي يوضح العناصر وخصائصها.
- **كتابة التعليمات البرمجية المناسبة** لتصميم الواجهة الرسومية من خلال النافذة الرئيسية ،window ،ومجموعة من العناصر widgets ، وضبط خصائص كل عنصر properties مثل حجم / لون / نص مستعيناً بأحد المكتبات المتخصصة في التصميم الرسومي.
- **تصميم الاستجابة للأحداث (BacK-end)**.
- **اختبار البرنامج من حيث:** التأكد من استجابة الأزرار والقوائم ب- معالجة البيانات والأخطاء.
- تصدير البرامج بإنشاء ملف تنفيذي.


التأكيد على المتعلمين أنه من الأمور الهامة التأكد من عمل البرنامج بشكل صحيح يتبعه تصديره كملف تنفيذي ليسهل تشغيله دون الحاجة إلى بيئة تطوير.

# مكتبة ICT_KW

مكتبة خاصة مبنية على البرمجة الشيئية OOP (Object Oriented Programming ) ، تم تصميمها لتسهيل إنشاء واجهات رسومية باستخدام لغة Python مبنية على مجموعة من المكتبات مثل **TKinter** و **Pillow** وغيرها من المكتبات المساندة.

## البرمجة الشيئيةObject Oriented Programming(OOP):

تعَدّ البرمجة كائنية التوجه- **Object Oriented Programming** أو **OOP** اختصارًا- نمطًا من أنماط البرمجة التي يُكتَب فيها البرنامج على صورة كائنات تحتوي على خاصيات ومهام -أي دوال- فيمكن مثلًا التفكير في السيارة على أساس كائن، إذ يكون لون السيارة ولون إطاراتها واسم الشركة المصنعة للسيارة خاصيات لها؛ أما مهمة حركة السيارة ومهمة إنقاص سرعة السيارة ومهمة زيادة سرعة السيارة، فهي دوال.
كما يمكن التفكير أيضًا في الأقلام على أنها كائنات، يكون لون القلم واسم الشركة المصنعة له خاصيات؛ أما مهمة الكتابة فهي دالة، كذلك الأشخاص يمكن التفكير فيهم من وجهة نظر البرمجة الكائنية على أساس كائنات لها خاصيات ودوال، إذ يكون اسم الشخص ولون بشرته وطول قامته خاصيات؛ أما مهمة المشي ومهمة الجري ومهمة الأكل فهي دوال، وكذلك مهمة النوم والجلوس والعمل، ...إلخ فهي كلها دوال موجودة في الكائن.

### نشاط

### بناء مشروع رسومي «الوقت والتاريخ »

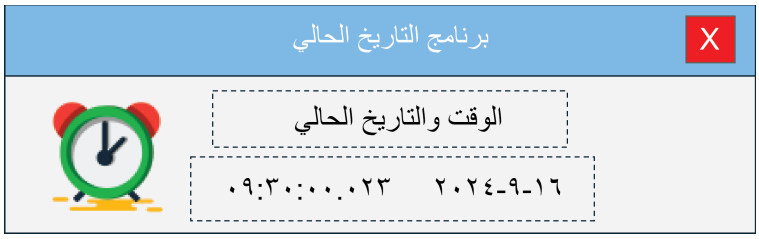
التعرف على كيفية استخدام مكتبة **ICT_KW** وتصميم النافذة الرئيسية **window** ، ثم إضافة مجموعة من العناصر إليها مثل العنوان **Label** ، والصورة **Image**. , يمر المشروع بعدة خطوات كالتالي سنتناول توضيحها تدريجيا

# تابع - مكتبة ICT_KW

## الخطوة الأولى : تصميم الواجهة الرسومية

### أولا : رسم واجهة المستخدم الرسومية

تصور الشكل النهائي برسم مخطط للواجهة المطلوبة

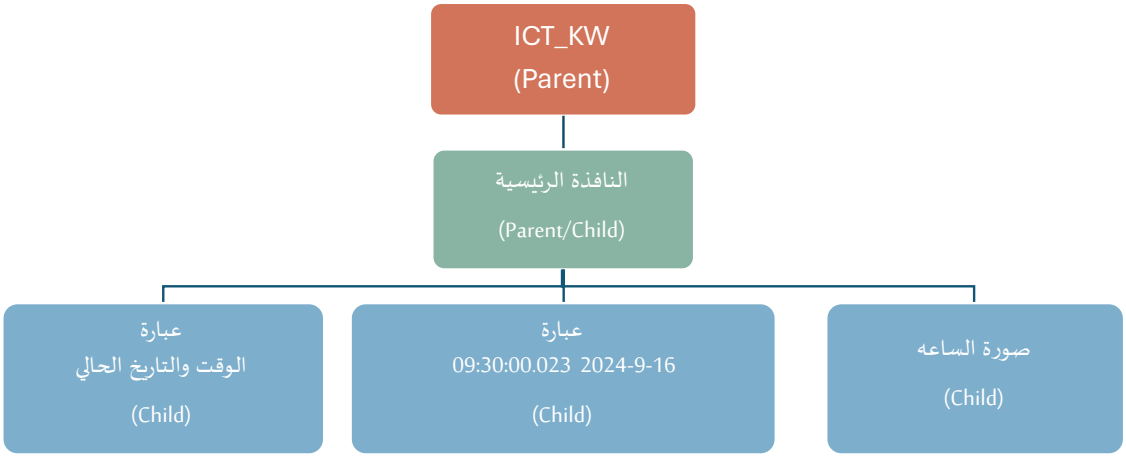



| | |
|---|---|
| تشجيع المتعلمين على تخطيط العناصر التي يريدون إضافتها وتحديد أماكنها، مما يسهل عليهم فهم التصميم وتنفيذه لاحقًا | |
| استخدام (ورق وأقلام )أو يمكن استخدام برامج التصميم البسيطة مثل **PowerPoint** أو **Draw.io** أو حتى تطبيقات الرسم على الأجهزة اللوحية | |
| يمكنك عرض مثال مخطط النهائي بسيط للطلاب كمرجع. | |
| تعريف المتعلمين بأهمية رسم مخطط للواجهة قبل البدء بالبرمجة، و تحديد العناصر الأساسية(مثل العناوين، الصور، الأزرار) ووضع تصور واضح للبرنامج. | |
| مناقشة تصاميم المتعلمين ، وتشجيع على تقديم اقتراحات لتحسين المخططات وتوزيع العناصر بطريقة مناسبة. | |

### ثانيا : تحديد العناصر(Widgets) :

التأكيد أنه بعد تحديد العناصر الأساسية المطلوبة وتوضيح خصائصها.

أهمية رسم مخطط هيكلي يهدف ايضاح العلاقة بين هذه العناصر وكيفية تنظيمها.



# تابع - مكتبة ICT_KW

في المخطط الهيكلي السابق، نرى التنظيم الهرمي للعناصر باستخدام مكتبة ICT_KW. المخطط يوضح العلاقات بين العناصر الرئيسية والعناصر الفرعية، كالتالي:

1. ICT_KW (Parent):

المكتبة المستخدمة لإنشاء الواجهة الرسومية تعتبر هذه المكتبة "الأساس" أو "Parent" لجميع العناصر التي سنستخدمها .

2. النافذة الرئيسية:(Parent/Child)

النافذة الرئيسية هي العنصر الذي يحتوي على جميع العناصر المتضمنة هي في ذاتها "Parent" للعناصر ، وفي الوقت نفسه "Child" للمكتبة ICT_KW.

- تحتوي النافذة على العنوان الرئيسي وتعرض الوقت والتاريخ وصورة الساعة.

3. العناصر الفرعية:(Children)

النافذة الرئيسية تحتوي على ثلاثة عناصر أساسية، وكل منها يعتبر "Child" للنافذة الرئيسية

- عبارة "الوقت والتاريخ الحالي:"
  - عبارة نصية توضح الهدف من البرنامج تظهر كعنوان أو وصف داخل النافذة.
- عبارة الوقت والتاريخ الفعلي:
  - عنصر يعرض الوقت الفعلي يتحدث بانتظام لكل من التاريخ والوقت الحاليين بصيغة مثل "16-9-2024 09:30:00.023".
- صورة الساعة:
  - عنصر صورة يمثل أيقونة الساعة، ويظهر على الواجهة يعبر عن موضوع التطبيق.

ملاحظة للمعلم

## تفاصيل تنظيم العناصر يجب على المعلم التأكيد عليها

1- المكتبة (ICT_KW) هي المصدر الأساسي الذي يتم من خلاله بناء العناصر.

2- النافذة الرئيسية تحتوي على كل العناصر الفرعية وهي النقطة المركزية يعرض فيها واجهة المستخدم.

3- العناصر الفرعية تتضمن النصوص والصورة موزعه على النافذة ظاهرة للمستخدم.

التأكد قبل العمل في أي مشروع إضافة المكتبة (ICT_KW)

1- انسخ مجلد المكتبة ict_Kw إلى نفس المجلد الخاص بمشروعك.

2- استورد المكتبة في الكود الخاص بك:

```
from ict_kw. components import Component Name
```

3- يمكن تحويلها إلى حزمة قابلة للتثبيت وارفاقه بمجلد المشروع في صورة ملف بايثون

# تابع - مكتبة ICT_KW

## ثالثا : كتابة التعليمات البرمجية

### أ-إنشاء مشروع جديد باسم FirstGUI

1- من القائمة الرئيسية، اختر **File --- New Project**

2- اختر مكان حفظ المشروع - اضغط **Create** لإنشاء المشروع.

### ب_ إنشاء ملف Python باسم time_date_GUI

1-من خلال نافذة المشروع، اضغط زر الفأرة الأيمن على المجلد **FirstGUI**

2-اختر من القائمة المختصرة **New** -- **Python File** اكتب اسم الملف **time_date_GUI**

### ج-تثبيت مكتبة ICT_KW

افتح أداة **Terminal** في **PyCharm** واكتب الأمر التالي:

```
Pip install ICT_KW
```

الأمر سيعمل على تنزيل وتثبيت المكتبة على بيئة **Python** الخاصة بالمشروع( يأخذ فترة من الوقت لاكتمال التثبيت)

```
Terminal: Local
8c3cab015ad7ce4b838f8a24f16b676b1ac7c/pillow-11.0.0-cp312-cp312-win_amd64.whl.metadata
  Downloading pillow-11.0.0-cp312-cp312-win_amd64.whl.metadata (9.3 kB)
Collecting numpy>=1.21.2 (from opencv-python->ICT_KW)
  Obtaining dependency information for numpy>=1.21.2 from https://files.pythonhosted.org/packages/a6/84/fa11dad3404b7634
aaab50733581ce11e5350383311ea7a7010f464c0170/numpy-2.1.3-cp312-cp312-win_amd64.whl.metadata
  Using cached numpy-2.1.3-cp312-cp312-win_amd64.whl.metadata (60 kB)
Downloading ICT_KW-0.0.2-py3-none-any.whl (4.8 kB)
Downloading opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl (38.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.8/38.8 MB 10.0 kB/s eta 0:55:11
```

### د- استدعاء مكتبةICT_KW

يتم استدعاء المكتبة في ملف time_date_GUI.py باستخدام الأمر التالي:

```
import ICT_KW
```

### هـ - تشغيل النافذة الرئيسية باستخدام الدالة التالية :

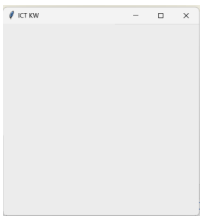
```
ICT_KW . run()
```

**تشغيل النافذة**

```
import ICT_KW

# تشغيل النافذة الرئيسية باستخدام دالة run (إذا كانت متوفرة في المكتبة)
ICT_KW.run()
```

# تابع - مكتبة ICT_KW

و- تشغيل البرنامج بالضغط على زر Run أو المفاتيح ( Shift + F10 ) ، لتظهر النافذة التالية:




استخدام الأمر **dir (ICT_KW)** للتأكد من وجود دالة **run()** داخل مكتبة **ICT_KW**. أو أي دالة تحتاجها يتم عرضها.

**Input**

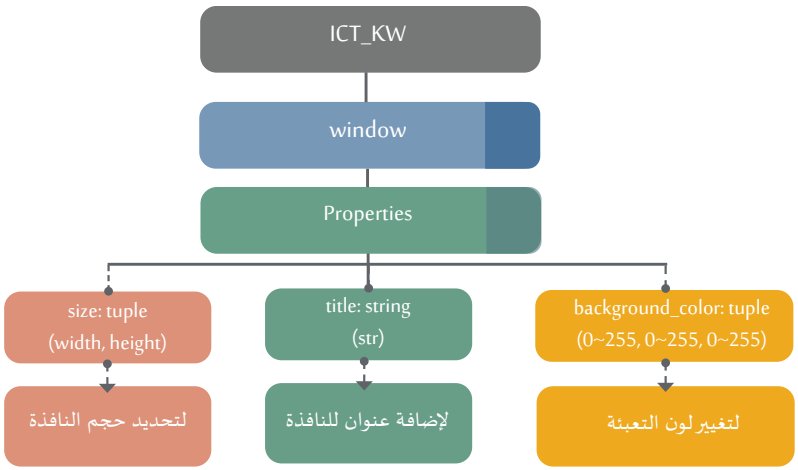
```
import ICT_KW

print(dir(ICT_KW))
```

**Output**

```
C:\Users\HP\Documents\firstGUI\.venv\Scripts\python.exe C:\Users\HP\Documents\firstGUI\time_date_GUI.py
['ICT_KW', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__', '__package__', '__path__',
 '__spec__', 'new_Button', 'new_Image', 'new_Label', 'new_ListBox', 'new_TextBox', 'new_Timer', 'run', 'show_messagebox',
 'window']

Process finished with exit code 0
```

## رابعا :التحكم في خصائص النافذة الرئيسية (Window)

النافذة الرئيسية هي الإطار الأساسي الذي يحتوي على جميع عناصر الواجهة الرسومية، مثل الأزرار، العناوين، والصور .



# تابع - مكتبة ICT_KW

المخطط يوضح كيفية استخدام مكتبة **CT_KW** الإنشاء نافذة رسومية والتحكم في خصائصها المختلفة باستخدام الدالة **window**، بالإضافة إلى الخصائص (**Properties**) التي يمكن تعديلها لتخصيص النافذة.

## المكتبة(ICT_KW)

- هي المكتبة المستخدمة لتصميم الواجهات الرسومية تحوي دالة رئيسية تُسمى **window** لإنشاء النافذة الرسومية.

## الدالة(window)

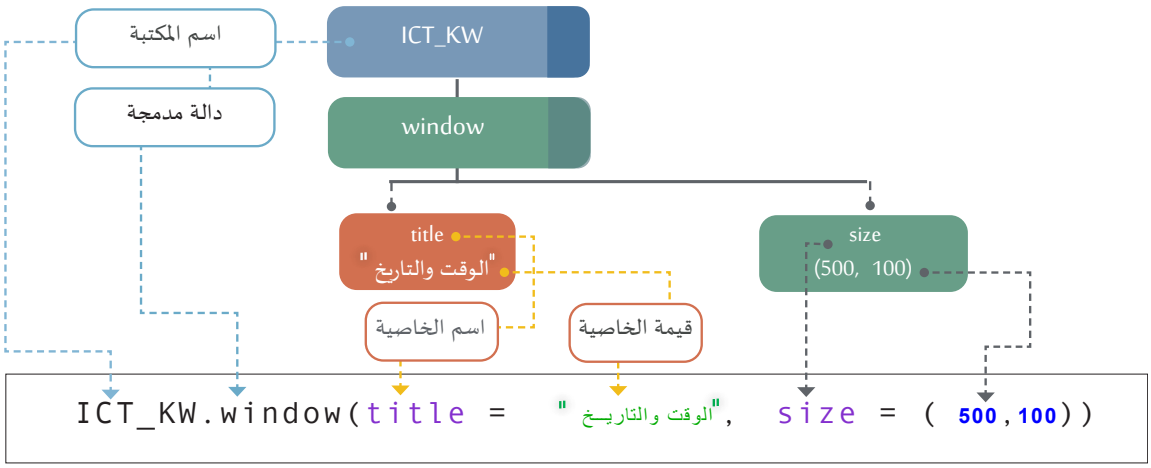
هي دالة رئيسية مدمجة داخل مكتبة ICT_KW، وتُستخدم لإنشاء نافذة يمكن التحكم بخصائصها

**الصيغةSynt:** ICT_KW. window(Properties)

## خصائص النافذة(Properties)

يوجد في المخطط ثلاث خصائص رئيسية يمكن استخدامها لتخصيص النافذة:

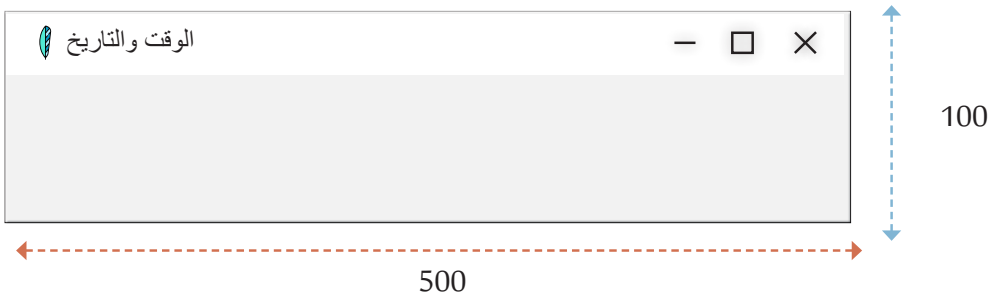
| | |
|---|---|
| تحديد حجم النافذة(**size**)<br>نوع الخاصية:(tuple لتحديد حجم النافذة بقيمتين ( العرض والارتفاع ) | |
| عنوان النافذة(title)<br>نوع الخاصية ( string) :نص لتعيين عنوان للنافذة يظهر في شريط العنوان. | |
| لون خلفية النافذة(background color)<br>نوع الخاصيةtuple :، تتكون من ثلاث قيم (أحمر، أخضر، أزرق) تتراوح بين ( 0 g255 ) لكل لون ،لتغيير لون خلفية النافذة باستخدام الألوان **RGB** . | |



# تابع - مكتبة ICT_KW

Input

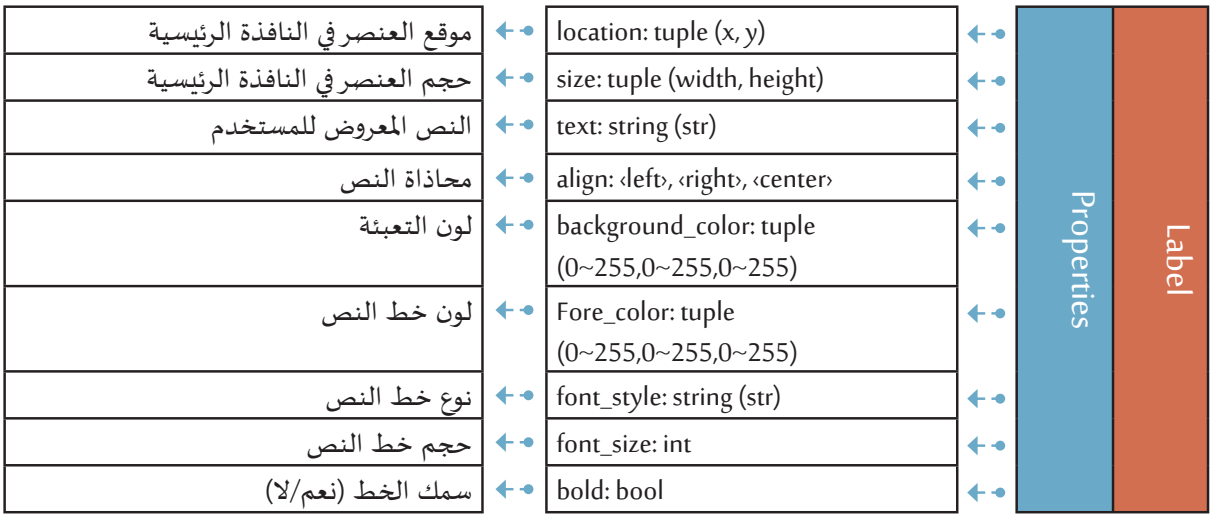
```
import ICT_KW

# إعداد النافذة بخصائص معينة مع تعيين الخلفية باللون الأبيض باستخدام RGB
ICT_KW.window(title="الوقت والتاريخ", size=(500, 100),
    background_color=(255, 255, 255) )

# تشغيل النافذة باستخدام ICT_KW.run()، إذا كانت الدالة موجودة في المكتبة
ICT_KW.run()
```




## اضافة عناصر واجهة المستخدم (Widgets) إلى النافذة الرئيسية :

باستخدام مكتبة ICT_KW، هناك طريقتان يمكنك من خلالهما إنشاء العناصر (مثل العناوين والأزرار) ووضعها في النافذة.

1. إنشاء عنصر بطريقة مباشرة لا يتم إسناده إلى متغير، وبالتالي لا يمكن تعديل خصائصه بعد إنشائه.
2. إنشاء عنصر بطريقة غير مباشرة وإسناده إلى متغير وبالتالي يمكن تعديل خصائصه بعد انشائه.



# تابع - مكتبة ICT_KW

## ١- إضافة العنوان(Label)

Label هو عنصر مرئي يعرض سلسلة

الصيغة العامة لإنشاء عنصر Label - باستخدام مكتبة CT_KW اهي:

Label_name (Optional) = ICT_KW.new_Label (Properties)

- Label_name: اسم اختياري لإسناد Label إلى متغير - امكانية تعديل خصائصه لاحقًا.
- ICT_KW.new_Label() الدالة المستخدمة لإنشاء Label في مكتبة ICT_KW.
- Properties الخصائص المختارة للعنصر - مثل النص، الخط، اللون، والموقع.

## شرح الخصائص - لعنصر Label

| | |
|---|---|
| الموقع:(location) : | يحدد موقع Labelداخل النافذة<br>location=(x, y) ( x و y تمثل الإحداثيات الأفقية والعمودية)<br>tuple (x, y)<br>location=(120, 15)<br>**ملاحظة:موقع العنصر Location داخل النافذة يحدد من بداية الزاوية العلوية اليسرى حسب الإحداثي السيني والصادي للنافذة.** |
| الحجم:(size) | يحدد أبعاد Label ( العرض والارتفاع)<br>size=(width, height)<br>tuple (width, height)<br>size=(250, 60) |
| النص:(text) | text="النص المعروض للمستخدم "<br>نوع الخاصية: string (str)<br>مثال: "الوقت والتاريخ الحالي" = text |
| محاذاة النص:(align) | يحدد محاذاة النص داخل Label<br>"left", "right", "center"<br>مثال:align="center" |
| لون الخلفية:(background_color) | لون خلفية Label باستخدام قيم RGB<br>background_color=(R, G, B)<br>tuple (R, G, B) 255و 0 تتراوح بين gR و G و B<br>مثال: للون الأبيض background_color=(255, 255, 255) |

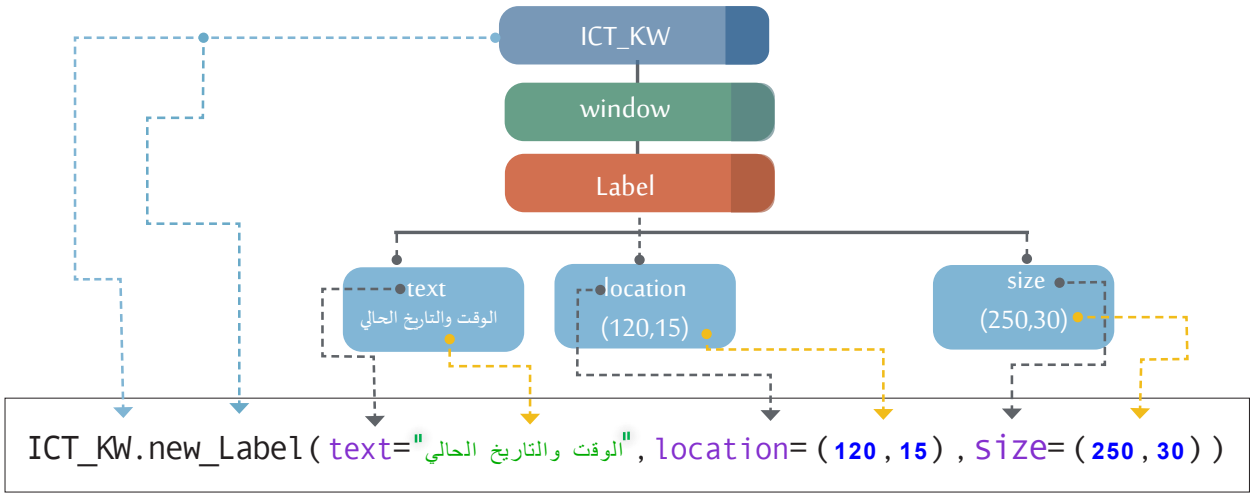
# تابع - مكتبة ICT_KW

## تابع شرح الخصائص - لعنصر Label

| | |
|---|---|
| لون النص:(Fore_color) | لون النص داخل Label باستخدام قيم RGB<br>Fore_color=(R, G, B)<br>tuple (R, G, B)<br>مثال للون الأسود: Fore_color=(0, 0, 0) |
| نوع الخط:(font_style) | نوع الخط المستخدم للنص<br>font_style="اسم الخط "<br>string (str)<br>مثال:"font_style="Arial |
| حجم الخط:(font_size) | حجم الخط للنص<br>رقم font_size: int<br>font_size=<br>مثال: font_size=14 |
| سماكة الخط:(bold) | يحدد سماكة النص بخط عريض(True)<br>أو غير عريض(False)<br>bool (True /False)<br>مثال:bold=True |

إنشاء عنوان Label1 باستخدام الطريقة المباشرة يتم ذلك بإضافة Labelمباشرة داخل النافذة الرئيسية دون إسناده إلى متغير وهنا لا يمكن التعديل على الخصائص بعد انشائه .

"الوقت والتاريخ الحالي" من خلال الهيكل الرسومي الهيكل الرسومي التالي:



# تابع - مكتبة ICT_KW

## تابع :إضافة العنوان(Label)

إنشاء عنوان Label 1 باستخدام الطريقة غير المباشرة وإسناده لمتغير، مثال:

```
label2=ICT_KW.new_Label(text=str(datetime.now()),location=(25,50),size=(450,30))
```

**استخدام الطريقة المباشرة**

```
import ICT_KW
from datetime import datetime
# إعداد النافذة بخصائص معينة
window = ICT_KW.window(
    title="الوقت والتاريخ ",
    size=(500, 100),
    background_color=(255, 255, 255)
)
# إنشاء عنوان "الوقت والتاريخ الحالي" بالطريقة المباشرة
ICT_KW.new_Label(
    text="الوقت والتاريخ الحالي",   # النص الذي يظهر كعنوان
    location=(120, 15),             # موقع العنوان داخل النافذة
    size=(250, 30),                 # حجم العنوان
   )
# إنشاء Label2 لعرض الوقت الحالي بالطريقة المباشرة
ICT_KW.new_Label(
    text=str(datetime.now()),       # النص الذي يعرض الوقت والتاريخ الحالي
    location=(120, 50),             # موقع النص داخل النافذة
    size=(250, 30),                 # حجم النص
   )
# تشغيل النافذة
ICT_KW.run()
```

**استخدام الطريقة غير المباشرة**

```
import ICT_KW
from datetime import datetime
# إعداد النافذة بخصائص معينة
window = ICT_KW.window(
    title="الوقت والتاريخ ",
    size=(500, 100),
    background_color=(255, 255, 255) )
# إنشاء عنوان "الوقت والتاريخ الحالي" بالطريقة غير المباشرة
label1 = ICT_KW.new_Label(
    text="الوقت والتاريخ الحالي",   # النص الذي يظهر كعنوان
    location=(120, 15),             # موقع العنوان داخل النافذة
    size=(250, 30),                 # حجم العنوان
   )
# إنشاء Label2 لعرض الوقت الحالي بالطريقة غير المباشرة
label2 = ICT_KW.new_Label(
    text=str(datetime.now()),       # النص الذي يعرض الوقت والتاريخ الحالي
    location=(120, 50),             # موقع النص داخل النافذة
    size=(250, 30),                 # حجم النص
)
# تشغيل النافذة
ICT_KW.run()
```

عند تشغيل البرنامج تظهر النافذة الرئيسية بالشكل التالي:


## 2-إضافة الصورة (Image)

عنصر مرئي يعرض صورة

| image | Properties | |
|---|---|---|
| | location: tuple (x, y) | موقع العنصر في النافذة الرئيسية |
| | size: tuple (width, height) | حجم العنصر في النافذة الرئيسية |
| | image_file_path: string (str) | اسم وامتداد الصورة المراد عرضها للمستخدم من نوع سلسلة نصية. |
| | background_color: tuple (0~255,0~255,0~255) | لون التعبئة |

# تابع - مكتبة ICT_KW

التأكيد للمعلم على كيفية شرح الخصائص **(Properties)** التي يمكن تخصيصها عند اضافة الصورة داخل النافذة الرسومية.

يتم استخدام الدالة **new_Image** لإضافة الصور، مع إمكانية تحديد موقعها، حجمها، لون الخلفية، ومسار ملف الصورة.

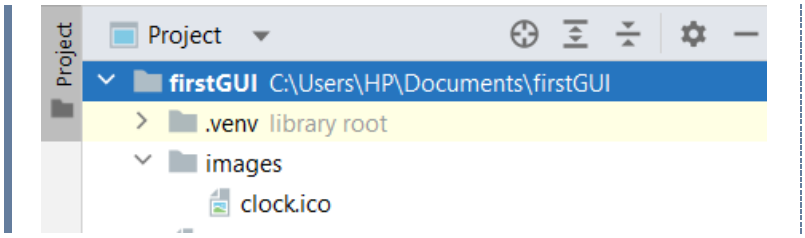
## شرح الخصائص(Properties)

| | |
|---|---|
| الموقع ( location ): | يحدد موقع الصورة داخل النافذة<br>tuple (x, y)<br>location= (x, y) ( X و Y تمثل الإحداثيات الأفقية والعمودية)<br>location= (40, 20) |
| الحجم:(size) | يحدد أبعاد Label ( العرض والارتفاع) — size=(width, height)<br>tuple (width, height) size=(60, 60)<br>تعني أن عرض الصورة 60 بكسل وارتفاعها 60 بكسل |
| مسار ملف الصورة<br>(image_file_path): | سلسلة نصية تحتوي على اسم وامتداد الصورة المراد عرضها، مع المسار لملف الصورة<br>string (str)<br>image_file_path ="images/clock.ico "<br>تعني أن ملف الصورة clock.ico الموجود في مجلد images داخل المشروع.<br>الامتدادات الشائعة المدعومة:( TIFF- BMP- GIF- JPG/JPEG- PNG - CO ) |

### إضافة ملف "صورة" إلى مجلد المشروع FirstGUI

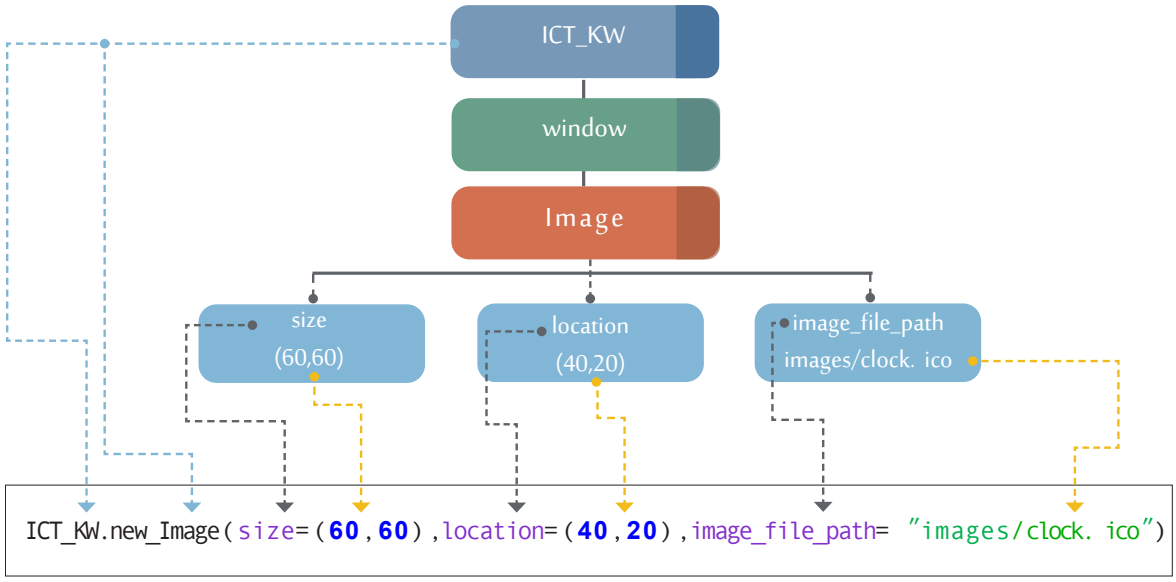
**استيراد ملف الصورة clock.ico من مجلد الصور إلى مجلد المشروع باتباع الخطوات التالية:**

1. فتح مستكشف الملفات وابحث عن المجلد الخاص بالمشروع وإنشاء مجلد باسم **images**.
2. نسخ ملف الصورة **clock.ico** من مجلد أوراق العمل ولصق في مجلد **images** داخل مجلد المشروع.
3. الانتقال إلى برنامج **PyCharm** وملاحظة ظهور ملف الصورة في مجلد المشروع.




```
image_file_path="images/clock.ico"
```

# تابع - مكتبة ICT_KW




| لون الخلفية:(background_color) | لون خلفية خلف الصورة باستخدام قيم RGB |
|---|---|
| | background_color=(R, G, B) |
| | tuple (R, G, B) |
| | تتراوح بين -0 و255 (R, G, B) |
| | خلفية بيضاء background_color=(255,255,255) |

**استخدام الطريقة المباشرة**

```
22  #   إضافة صورة بالطريقة المباشرة
23  ICT_KW.new_Image
24      location=(40
25      size=(60, 60),
26      image_file_path="images/clock.ico",   # مسار ملف الصورة
27  )
```

Typo: In word 'صورة'
Save 'صورة' to dictionary Alt+Shift+Enter  More actions... Alt+Enter

**استخدام الطريقة غير المباشرة**

```
20  #   إضافة صورة بالطريقة غير المباشرة
21  Image_name = ICT_KW.new_Image(
22      location=(40, 20),              # موقع الصورة
23      size=(60, 60),                  # حجم الصورة
24      image_file_path="images/clock.ico",   # مسار ملف الصورة
25  )
```

## الخطوة الثانية: تصميم الاستجابة للأحداث

لا يحتوي المشروع الحالي على عناصر تستجيب لمدخلات المستخدم

## الخطوة الثالثة: اختبار البرنامج

عند تنفيذ البرنامج تظهر النافذة التالية:

الانتقال إلى برنامج PyCharm وملاحظة ظهور ملف الصورة في مجلد المشروع.

# تابع - مكتبة ICT_KW


## الخطوة الرابعة: تصدير البرنامج إلى ملف تنفيذي(EXE)

تتيح لك عملية تحويل ملف **py** إلى ملف تنفيذي **exe** تشغيل البرنامج مباشرة دون الحاجة إلى التعليمات البرمجية في **PyCharm**

يمكن استخدام مكتبة **auto-py-to-exe** حيث توفر واجهة رسومية تتيح للمستخدم تحديد اختياراته

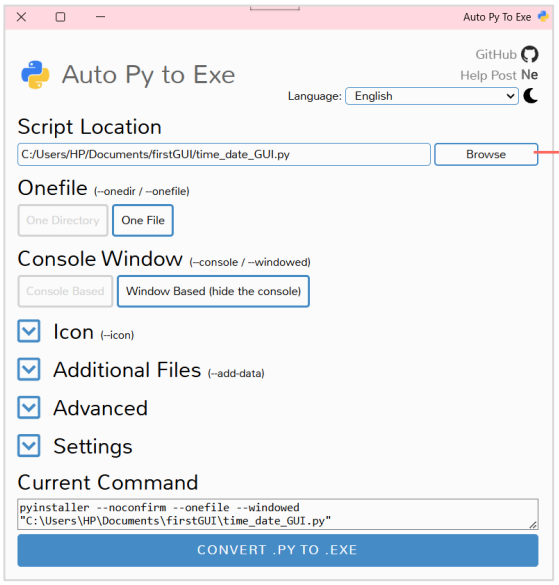
### خطوات تحويل البرنامج إلى ملفEXE

### المرحلة الأولى : تثبيت مكتبة auto-py-to-exe

Terminal

```
Downloading pefile-2023.2.7-py3-none-any.whl (71 kB)
Downloading pyinstaller_hooks_contrib-2024.10-py3-none-any.whl (338 kB)
Downloading pywin32_ctypes-0.2.3-py3-none-any.whl (30 kB)
Using cached setuptools-75.5.0-py3-none-any.whl (1.2 MB)
Downloading urllib3-2.2.3-py3-none-any.whl (126 kB)
Downloading altgraph-0.17.4-py2.py3-none-any.whl (21 kB)
Downloading bottle-0.13.2-py2.py3-none-any.whl (104 kB)
Downloading future-1.0.0-py3-none-any.whl (491 kB)
Downloading pyparsing-3.2.0-py3-none-any.whl (106 kB)
Downloading whichcraft-0.6.1-py2.py3-none-any.whl (5.2 kB)
Downloading gevent_websocket-0.10.1-py3-none-any.whl (22 kB)
Downloading gevent-24.11.1-cp312-cp312-win_amd64.whl (1.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 5.5 MB/s eta 0:00:00
Downloading cffi-1.17.1-cp312-cp312-win_amd64.whl (181 kB)
Downloading greenlet-3.1.1-cp312-cp312-win_amd64.whl (299 kB)
```

# تابع - مكتبة ICT_KW




- افتح Terminal من داخل PyCharm
- pip install **auto-py-to-exe**

اضغط مفتاح الإدخال Enter حتى يتم تثبيت المكتبة.

- تشغيل أداة **auto-py-to-exe** بعد التثبيت

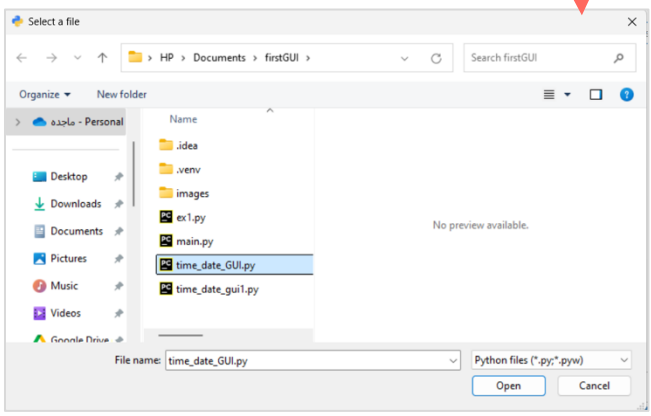
**auto-py-to-exe** ثم اضغط مفتاح الإدخال Enter

ستظهر واجهة رسومية تشبه صندوق حوار

**إعداد التحويل داخل واجهة auto-py-to-exe**

1. تحديد ملف البرنامج ملف
   - في الحقل الأول "**Script Location**"،

اضغط على زر Browse.




   - حدد ملف **Python** الذي تريد تحويله على

سبيل المثال. (**time_date_GUI.py**):

2. اختيار نوع الإخراج:
   - اختر أحد الخيارات:

- **One Directory** في حالة الإخراج مجلدًا يحتوي على الملف التنفيذي وملفات أخرى.
- **One File** في حالة الاخراج ملفًا تنفيذيًا واحدًا فقط .

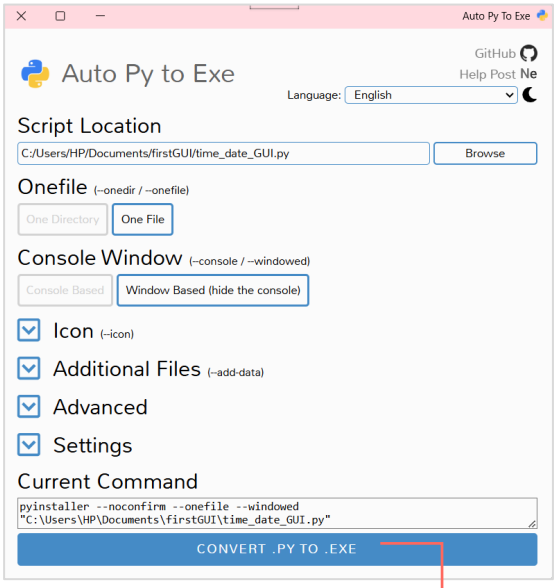
3. تحديد نوع الواجهة:

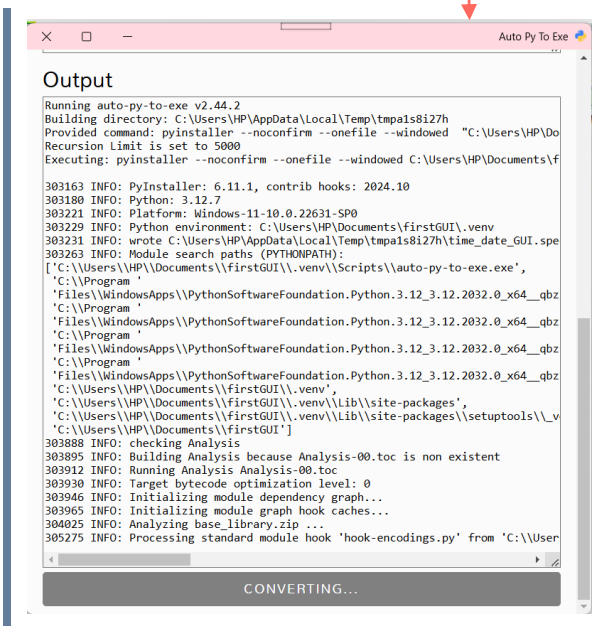
- **Console Based**: إذا كان البرنامج يعرض واجهة نصية.
- **Window Based**: إذا كان البرنامج يعرض واجهة رسومية

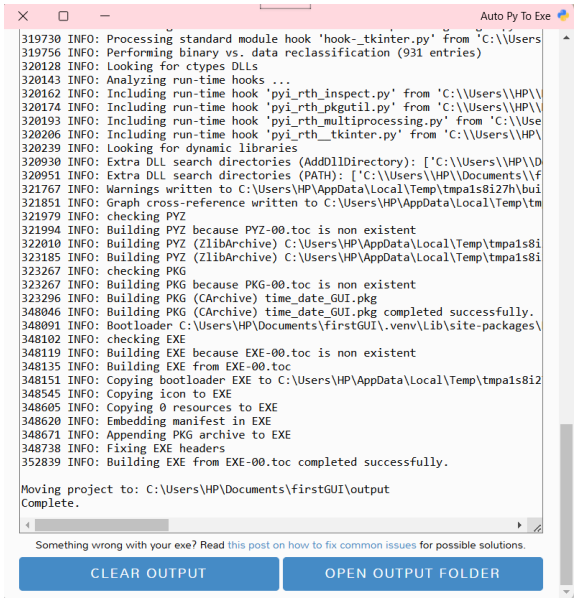
4. إضافة أيقونة للبرنامج (اختياري):
   - في حالة تخصيص أيقونة البرنامج، اختر "**Icon**" واضغط على **Browse** لتحديد ملف الأيقونة مثل **(clocK.ico)**.

5. بدء التحويل:

اضغط على زر Convert .py to .exe.

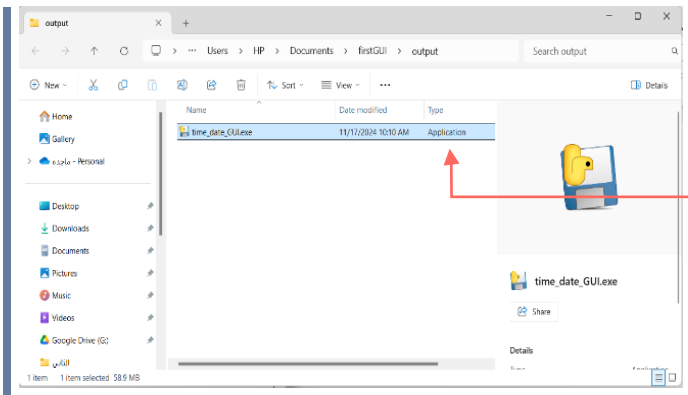



- Open Output Folder يفتح النافذه التي تحوي ملف exe

تشغيل البرنامج بالضغط على الملف Time_date_GUI.exe



# تابع - مكتبة ICT_KW

ورقة عمل (1)

اسم البرنامج: الوقت والتاريخ.

فكرة البرنامج: تصميم برنامج يعرض الوقت والتاريخ في واجهة مستخدم رسومية

شغل الملف التنفيذي Time&date.exe وعاين المنتج النهائي.

.شغل برنامج PyCharm.

.افتح المشروع First_GUI ، ثم أنشئ ملف Python جديد باسم Time&date.py.

.استدع مكتبة ICT_KW و datetime.

.صمم الواجهة الرسومية التالية بكتابة التعليمات البرمجية المناسبة مستعيناً بالمخطط في الأسفل.


الحـــل:

برنامج الوقت والتاريخ (التعليمات البرمجية)

```
import ICT_KW
from datetime import datetime
# إعداد النافذة بخصائص معينة
window = ICT_KW.window(
    title="الوقت والتاريخ ",
    size=(500, 100),
    background_color=(255, 255, 255) )
# إنشاء عنوان "الوقت والتاريخ الحالي" بالطريقة غير المباشرة
label1 = ICT_KW.new_Label(
    text="الوقت والتاريخ الحالي",    # النص الذي يظهر كعنوان
    location=(120, 15),              # موقع العنوان داخل النافذة
    size=(250, 30),                  # حجم العنوان
   )
# إنشاء Label2 لعرض الوقت الحالي بالطريقة غير المباشرة
label2 = ICT_KW.new_Label(
    text=str(datetime.now()),        # النص الذي يعرض الوقت والتاريخ الحالي
    location=(120, 50),              # موقع النص داخل النافذة
    size=(250, 30),                  # حجم النص
)
#   إضافة صورة بالطريقة غير المباشرة
Image_name = ICT_KW.new_Image(
    location=(40, 20),               # موقع الصورة
    size=(60, 60),                   # حجم الصورة
    image_file_path="images/clock.ico",  # مسار ملف الصورة
  )
# تشغيل النافذة
ICT_KW.run()
```

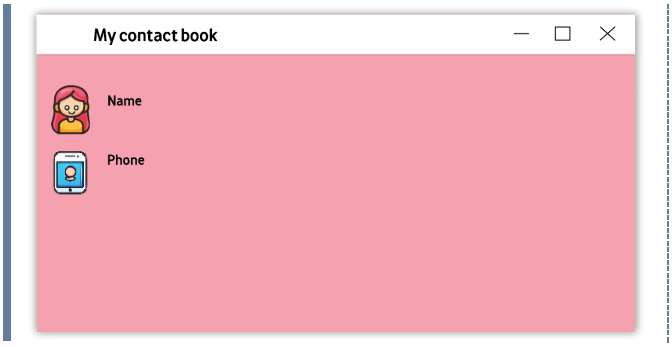
# تابع - مكتبة ICT_KW

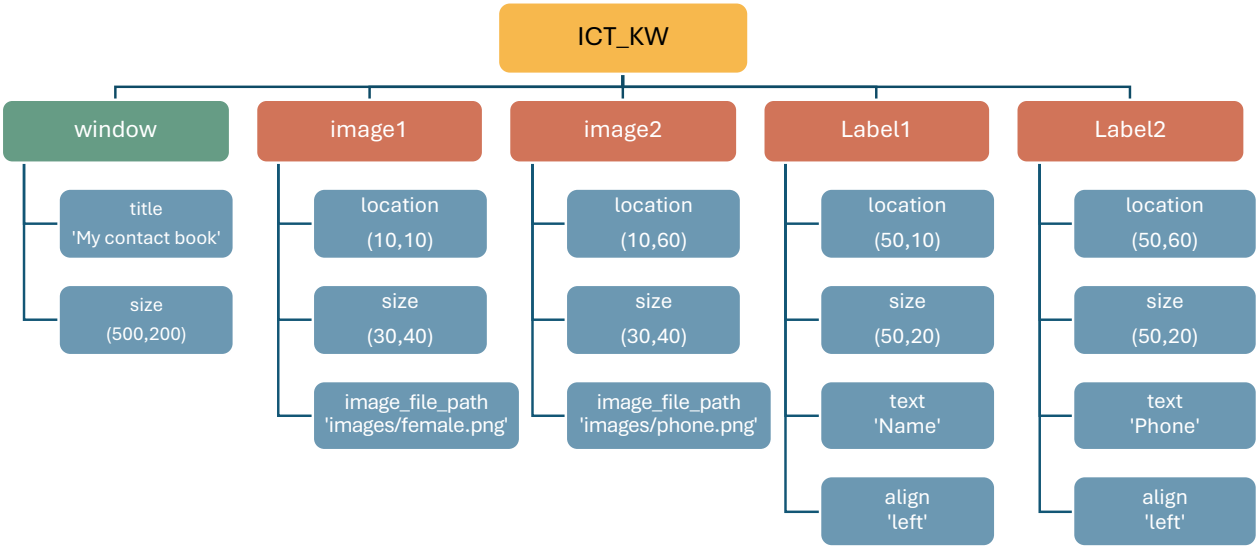
ورقة عمل ( 2 )

اسم البرنامج My Contact BooK :

فكرة البرنامج: تصميم برنامج جهة الاتصال يعرض اسم المتصل ورقم هاتفه في واجهة مستخدم رسومية، متبعًا الخطـوات التـالية:

- شغل الملف التنفيذي My Contact BooK.exe ، ثم عاين المنتج النهائي.لِبرنامج PyCharm.
- افتح المشروع First_GUI ، ثم أنشئ ملف Python جديد باسمMy Contact BooK.py.
- استدع مكتبة ICT_KW.
- صمم الواجهة الرسومية التالية بكتابة التعليمات البرمجية المناسبة مستعيناً بالمخطط في الأسفل.



# تابع - مكتبة ICT_KW

تابع حل ورقة عمل 2

برنامج My Contact Book ( التعليمات البرمجية)

```
import ICT_KW
# إنشاء النافذة الرئيسية
window = ICT_KW.window(
    title="My Contact Book",            # عنوان النافذة
    size=(500, 200),                    # حجم النافذة
    background_color=(253, 143, 170)    # لون الخلفية
)
# (female.png) إضافة الصورة الأولى
image1 = ICT_KW.new_Image(
    location=(10, 10),                  # موقع الصورة
    size=(30, 40),                      # حجم الصورة
    image_file_path="C:/Users/HP/Documents/firstGUI/images/female.png"  # مسار ملف الصورة
)
# (phone.png) إضافة الصورة الثانية
image2 = ICT_KW.new_Image(
    location=(10, 60),                  # موقع الصورة داخل النافذة
    size=(30, 40),                      # حجم الصورة
    image_file_path="C:/Users/HP/Documents/firstGUI/images/phone.png"  # مسار ملف الصورة
)

# (Label1: Name) إضافة النص الأول
Label1 = ICT_KW.new_Label(
    location=(50, 10),                  # موقع النص
    size=(50, 20),                      # حجم النص
    text="Name",                        # النص المعروض
    align="left",                       # محاذاة النص لليسار
)
# (Label2: Phone) إضافة النص الثاني
Label2 = ICT_KW.new_Label(
    location=(50, 60),                  # موقع النص
    size=(50, 20),                      # حجم النص
    text="Phone",                       # النص المعروض
    align="left",                       # محاذاة النص لليسار
)
# تشغيل النافذة
ICT_KW.run()
```

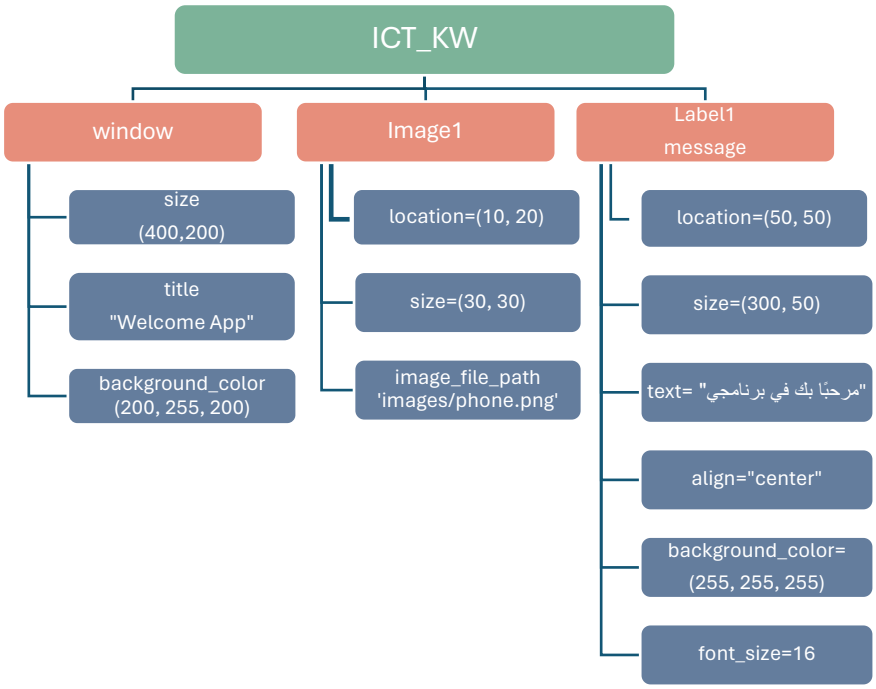
# تابع - مكتبة ICT_KW

ورقة عمل اثرائية ( 2 )

اسم البرنامج : رسالة ترحيبية

فكرة برنامج : تصميم برنامج بسيط يعرض رسالة ترحيبية في نافذة رسومية باستخدام مكتبة ICT_KW. متبعًا الخطـوات التـالية:

- أنشئ مشروعا جديدا باسم Welcome.
- أنشئ ملف Python جديد باسمwelcom_app.py.
- استدع مكتبة ICT_KW.
- صمم الواجهة الرسومية التالية بكتابة التعليمات البرمجية المناسبة مستعيناً بالمخطط في الأسفل.
- إنشاء نافذة بعنوان "Welcome App".
- عرض رسالة نصية في منتصف النافذة، مثل: "**مرحبًا بك في برنامجي**"!
- تغيير لون خلفية النافذة الى اللون الأخضر.
- أدرج صورة مناسبة.
- اختبر البرنامج.

# تابع - مكتبة ICT_KW

تابع :ورقة عمل اثرائية ( 2)

برنامج ( رسالة ترحيبية) التعليمات البرمجية

Reader Mode

```
import ICT_KW
# إنشاء النافذة الرئيسية
window = ICT_KW.window(
    title="Welcome App",                # عنوان النافذة
    size=(400, 200),                    # حجم النافذة
    background_color=(200, 255, 200) # لون الخلفية (أخضر فاتح)
)
# إضافة رسالة ترحيبية
message = ICT_KW.new_Label(
    location=(50, 50),                  # موقع الرسالة داخل النافذة
    size=(300, 50),                     # حجم الرسالة
    text="مرحبًا بك في برنامجي!",      # النص المعروض
    align="center",                     # محاذاة النص
    background_color=(255, 255, 255), # لون خلفية النص
    font_size = 16 # تنسيق الخط
)
# إضافة صورة
phone_image = ICT_KW.new_Image(
    location=(10, 20),
    size=(30, 30),
    image_file_path=r"C:\Users\HP\Documents\pythonProjectnew\images\phone.png"  # استخدام مسار الصورة
)
# تشغيل النافذة
ICT_KW.run()
```

# الوحدة الأولى :برمجة Python

## الدوال Functions

04

# الدوال Functions

## نتاجـــات التعـــلم

| | |
|---|---|
| 1 | التعرف على مفهوم الدوال وأهميتها:<br>• التعرف أن الدوال هي مجموعة تعليمات برمجية تنفذ مهام محددة.<br>• إدراك أهمية الدوال واستخدامها لتنظيم التعليمات البرمجية مما يسهل تطويرها وصيانتها. |
| 2 | إنشاء دوال بسيطة:<br>• كتابة دوال باستخدام الكلمة المفتاحية **def** وتنظيم التعليمات البرمجية داخلها. |
| 3 | استخدام المتغيرات داخل الدوال:<br>• التمييز بين المتغيرات المحلية التي يتم تعريفها واستخدامها داخل الدالة فقط، والعالمية التي يمكن الوصول إليها من أي جزء من البرنامج. |
| 4 | تمرير الوسائط إلى الدوال:<br>• تعلم كيفية إرسال البيانات إلى الدوال باستخدام الوسائط (**Parameters**) وكيفية استخدامها داخل الدالة. |
| 5 | إرجاع القيم من الدوال:<br>• التعرف على كيفية استخدام الكلمة المفتاحية **return** لإرجاع القيم من الدوال إلى البرنامج الرئيسي. |
| 6 | استكشاف الدوال المدمجة:<br>• التعرف على الدوال المدمجة في **Python** مثل **print()**، **input()**، **len()**، واستخدامها بشكل عملي. |

## المصطلحات : الدوال Functions

الدوال Functions

الدوال المدمجة Built-in functions

الدوال المعرَّفة من المستخدم User-defined functions

الدوال المجهولة Lambda functions

اسم الدالة: Function name

تعريف الدالة Define a function

# تابع المصطلحات : الدوال Functions

Function body

Function Head

صيغة الدالة Function Syntax

Function Parameter / معامات الدالة.

التعليمات Statements

الإرجاع Return Expression

Parameterالمعاملات

القيمالمُرسَلة Arguments

القيم الافتراضية لمعاملات الدوال defult parameters

مجهولة الاسم Anonymo

نطاق المتغيرات Scope

المتغير المحلي Local Variable

المتغير العام global variable

متوالية فيبوناشي Fibonacci Generator

النسبة الذهبية Golden Ratio

نظام الألوان Red, Green, Blu ( RGB )

# مدخل إلى الدوال Functions

**التعرف على مفهوم الدوال**

- **تعريف الدوال على أنها مجموعة من التعليمات البرمجية التي تؤدي مهام محددة.**

نشاط : سؤال للمتعلمين : ماهي الدوال؟

- تخيل أن لديك قائمة مهام يومية (الاستيقاظ -تنظيف الأسنان - تناول الفطور)
- كل مهمة منها هي "دالة "تقوم بتنفيذ شيء محدد.

أهمية الدوال:

1. تسهيل كتابة البرامج.
2. تحسين كفاءة المقطع البرمجي.
3. إعادة استخدام نفس التعليمات في أكثر من موضع.
4. تسهيل اكتشاف الأخطاء وإصلاحها.
5. الدوال تُساعد في تقسيم البرامج الكبيرة إلى أجزاء صغيرة (مهام فرعية) تسهل قراءتها، تعديلها، وتطويرها.
6. يتم استدعاء الدالة عند الحاجة إلى تنفيذها، مما يجعل الكود أكثر تنظيمًا وأقل تكرارًا.

شرح أهمية الدوال:

- التوضيح للمتعلمين أن الدوال تُستخدم لتقسيم البرامج الكبيرة إلى أجزاء صغيرة.
- ربط المفهوم بحياتهم اليومية: مثل تقسيم المهام اليومية (الغسيل، الطهي، الدراسة، التخطيط ليوم دراسي).

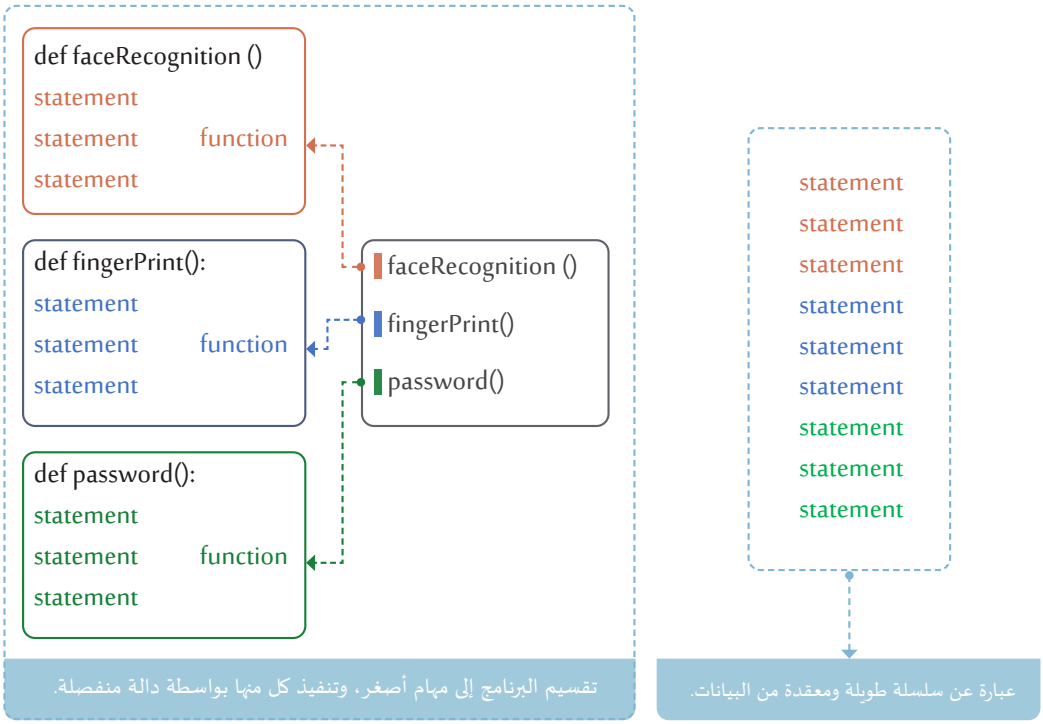
  مثال : التخطيط ليوم دراسي.

  مراجعة الدروس :قراءة الملاحظات ومراجعة المواضيع.

  أداء الواجبات :حل التمارين أو كتابة التقارير.

  أخذ استراحة :الحصول على وقت للاسترخاء – اللعب .
- تعريف الدوال على أنها مجموعة من التعليمات البرمجية التي تؤدي مهام محددة.

# مدخل إلى الدوال Functions




## شرح المخطط

### مفهوم الدالة:

- المخطط يوضح طريقة استخدام الدوال لتنظيم المقطع البرمجي:
  - يتم تقسيم البرنامج إلى عدة مهام (مثل: التعرف على الوجه، التعرف على البصمة، إدخال كلمة المرور).
  - كل مهمة يتم تنفيذها بواسطة دالة منفصلة.

### أجزاء الدالة:

- تعريف الدالة:
  - يتم استخدام الكلمة المفتاحية def لتعريف الدالة.
  - يتم إعطاء الدالة اسمًا يعبر عن وظيفتها.
- جسم الدالة:
  - يحتوي على التعليمات البرمجية التي تنفذ المهمة المطلوبة.

### تطوير المفهوم:.

- اجعل الطلاب يكتبون دوالًا متعددة تنفذ مهام مختلفة، مثل:
  - دالة للجمع بين رقمين.
  - دالة لطباعة تاريخ اليوم.

# مدخل إلى الدوال Functions

## الدوال المدمجة(Built-in Functions)

هي دوال جاهزة موجودة في Python. مثل الأدوات التي تأتي مع حقيبة الإسعافات الأولية، لا تحتاج إلى صنعها، بل فقط استخدامها.

- أمثلة:
  1. **print():** للطباعة على الشاشة.
  2. **input():** للإدخال من المستخدم.
  3. **len():** لمعرفة طول كلمة أو جملة.

## الدوال المعرفة من المستخدم(User-defined Functions)

- هي دوال يمكننا كتابتها لتنفيذ أمر معين

مثل: عندما نريد أن نقول **"مرحبًا"** لكل أصدقائنا، بدلاً من كتابة **"مرحبًا" 10 مرات**، نكتب دالة تقول ذلك مرة واحدة ونستخدمها عدة مرات.

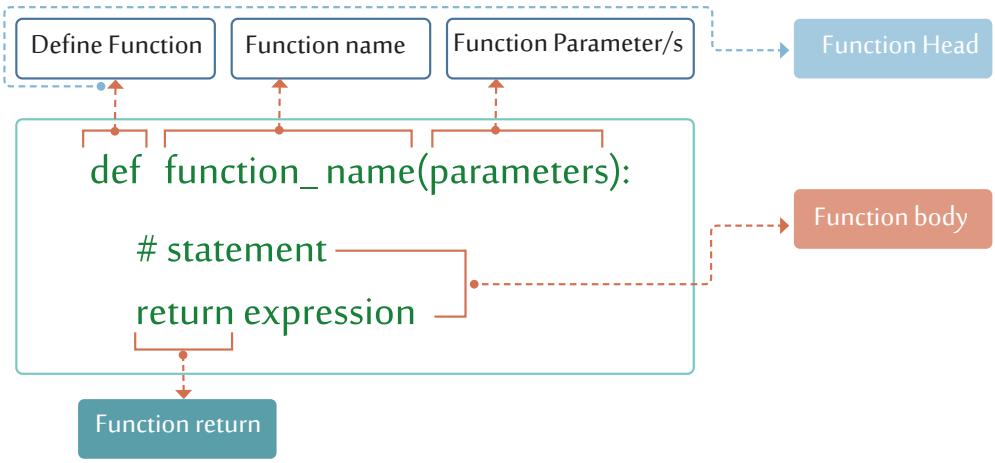
## الدوال المجهولةLambda function

- الدوال التي تُستخدم لتنفيذ وظائف بسيطة ومباشرة، وتُكتب في سطر واحد دون استخدام def للتعريف عنها.

## إنشاء الدوال Functions

### كيفية انشاء الدالة؟

بكتابة كلمة **def**، تسمية الدالة اسمًا، ويتم وضع التعليمات داخلها.



# مدخل إلى الدوال Functions

## Function Head

التأكيد على النقاط التالية :

- ما هي الدالة(Function) ؟
- رأس الدالة (Function Head)

اسم الدالة:(Function Name) مع التأكيد على قواعد تسمية المتغيرات عند تسمية الدوال في Python .

1. يجب أن يبدأ الاسم بحرف أو بشرطة سفلية(_).
2. لا يمكن أن يحتوي على مسافات (استخدم _ للفصل بين الكلمات).
3. لا يمكن أن يكون الاسم من الكلمات المحجوزة مثل def أو return.

- **معاملات الدالة:(Function Parameters)** المعامل **هو** جزء من تعريف الدالة : ويتم استخدامه لاستقبال القيم التي يتم تمريرها إلى الدالة عند استدعائها
  - معاملات :(parameters) .
  - قيم فعلية :(arguments) .

جسم الدالة(Function Body)

- يحتوي جسم الدالة على:
  - تعليمات (statements) لتنفيذ المهام.
  - تعليمة Return Expression لإرجاع قيمة عند الانتهاء.

## استدعاء دالة Function calling

- كتابة اسم الدالة متبوع بالأقواس الدائرية ( ) Parentheses.
- يمكن تمرير قيم (Arguments ) للدوال التي تتضمن معاملات ( Parameters).
- تنفيذ التعليمات البرمجية الموجودة في جسم الدالة.
- الانتقال مرة أخرى إلى جزء البرنامج الذي استدعى الدالة.
- استئناف البرنامج التنفيذ عند هذه النقطة.

## تابع:استدعاء دالة Function calling

مثـــال1: إنشاء دالة (بدون معاملات)

Input

```
def greet():   # تعريف الدالة
    print("Hello, Kuwait!")   # تعليمات داخل الدالة، تعرض رسالة ترحيبية عند استدعائها

greet()   # استدعاء الدالة لتنفيذ ما بداخلها

```

Output

```
Run:  ex3
C:\Users\HP\Documents\projectfunction\
Hello, Kuwait!

Process finished with exit code 0
```

شرح المثال للمتعلمين

1. def greet():
   - رأس الدالة Function Head
     - def: لتعريف الدالة - **greet** اسم الدالة.
     - () الأقواس الفارغة تشير إلى أن هذه الدالة لا تحتوي على أي معاملات.
     - النقطتان: تُستخدم لتحديد بداية جسم الدالة.
2. print("Hello, Kuwait!")
   - جسم الدالة Function Body :
     - تحتوي على التعليمات البرمجية التي يتم تنفيذها عند استدعاء الدالة.
     - **print()** دالة مدمجة في Python لطباعة النصوص.
     - **"Hello, Kuwait!"**: النص الذي سيتم طباعته.
3. greet()
   - استدعاء الدالة Function Calling
     - عند استدعاء الدالة greet، يتم تنفيذ التعليمات الموجودة في جسم الدالة (طباعة النص).
     - في حالة الاستدعاء الدالة greet لأكثر من مره

## استدعاء دالة أكثر من مرة

Input

```
def greet():  # تعريف الدالة
    print("Hello, Kuwait!")  # تعليمات داخل الدالة، تعرض رسالة ترحيبية عند استدعائها

greet()  # استدعاء الدالة لتنفيذ ما بداخلها


greet()
greet()
greet()
```

Output

```
C:\Users\HP\Documents\projectfunction
Hello, Kuwait!
Hello, Kuwait!
Hello, Kuwait!
Hello, Kuwait!

Process finished with exit code 0
```

مثـــال (2)

### إنشاء دالة (مع معاملات)

كتابة دالة باسم **add_numbers** تقوم بالآتي:

- تأخذ رقمين كمدخلات (معاملين) **(num1, num2)**
- تعيد ناتج جمع الرقمين.

يتم استدعاء الدالة باستخدام القيم 10 و 30

Input

```
def add_numbers(num1, num2):     #(num1, num2) حيث معاملين add_numbers تعريف الدالة

#  طباعة الناتج
    print( num1+num2)

# 30 و 10 استدعاء الدالة بقيم
add_numbers( num1: 10,  num2: 30)
```

شرح المثال للمتعلمين :

### تعريف الدالة:

- تعريف دالة باسم **add_numbers** باستخدام الكلمة المفتاحية def.
- الدالة تأخذ معاملين هما **num1 و num2**.
- يتم جمع القيمتين .

## تابع مثال ( 2): إنشاء دالة (مع معاملات)

### استدعاء الدالة:

- يتم استدعاء الدالة مع القيم 10 و30.
- القيمة 10 يتم تخزينها في المتغير num1، والقيمة 30 يتم تخزينها في المتغير num2 .

### طباعة الناتج:

- يتم استخدام print() لطباعة الناتج 40.

مثـــال ( 3) **استدعاء برنامج يحتوي على أكثر من دالة.**

**التفسير**: برنامج يمثل إضافة عنصر إلى عربة التسوق( shopping cart ) من نوع List ، وعرض محتوياتها.

Program

```
# تعريف دالة لإضافة عنصر إلى سلة التسوق
def add_item(shopping_cart, item):
    """
    تضيف عنصرًا جديدًا إلى سلة التسوق وتعرض رسالة تأكيد.#
    """
    shopping_cart.append(item)  # إضافة العنصر إلى القائمة
    print(f'This item: {item} added to shopping cart')  # طباعة رسالة تأكيد

# تعريف دالة لعرض محتويات سلة التسوق
def view_cart(shopping_cart):
    """
    تعرض جميع العناصر الموجودة في سلة التسوق.
    إذا كانت السلة فارغة، يتم عرض رسالة توضح ذلك.
    """
    if not shopping_cart:  # التحقق إذا كانت السلة فارغة
        print('Your shopping cart is empty.')  # رسالة عند عدم وجود عناصر
    else:
        print('Shopping cart contains:')  # رسالة توضح أن السلة تحتوي على عناصر
        for item in shopping_cart:  # عرض كل عنصر في السلة
            print(f'- {item}')

# إنشاء سلة تسوق فارغة
shopping_cart = []

# إضافة عناصر إلى سلة التسوق
add_item(shopping_cart, item: "Apple")  # إضافة تفاحة
add_item(shopping_cart, item: "Banana")  # إضافة موزة
add_item(shopping_cart, item: "Bread")  # إضافة خبز

# عرض محتويات سلة التسوق
view_cart(shopping_cart)
```

## تابع مثال 3 إنشاء دالة (مع معاملات)

### فكرة البرنامج:

البرنامج يمثل سلة تسوق إلكترونية توضيح أنه يمكن للمستخدم:

1. إضافة عناصر إلى السلة.
2. عرض العناصر الموجودة في السلة .إذا لم تكن هناك عناصر، سيعرض البرنامج رسالة تُوضح أن السلة فارغة.

Program

```
# تعريف دالة لإضافة عنصر إلى سلة التسوق
def add_item(shopping_cart, item):
    """
    #.تضيف عنصرًا جديدًا إلى سلة التسوق وتعرض رسالة تأكيد
    """
    shopping_cart.append(item)  # إضافة العنصر إلى القائمة
    print(f'This item: {item} added to shopping cart')  # طباعة رسالة تأكيد
```

### شرح المثال للمتعلمين :

- تعريف الدالة add_item .
  - **اضافة عنصرًا** جديدًا إلى سلة التسوق.
  - تُطبع رسالة تعلم المستخدم أن العنصر تمت إضافته.
- المعاملات:(Parameters)
  - shopping_cart: القائمة التي تمثل سلة التسوق .
  - item: العنصر الذي سيتم إضافته.
  - append() تستخدم لإضافة العنصر إلى القائمة ويتبعها رسالة تأكيد باستخدام print().
- تعريف الدالة الثانية view_cart.
  - عرض محتويات سلة التسوق.
- المعاملات:(Parameters)
  - shopping_cart قائمة تحتوي على العناصر المضافة.

## تابع مثال ( 3 )

if not shopping_cart. للتحقق إذا كانت سلة التسوق فارغة

- إذا كانت السلة فارغة، يتم طباعة رسالة توضح " "Your shopping cart is empty
- إذا كانت تحتوي على عناصر، يتم عرض قائمة بجميع العناصر.

### المقطع البرمجي

```
print('Shopping cart contains:')
for item in shopping_cart:
    print(f'- {item}')
```

---

- إنشاء سلة التسوق : shopping_cart = []
- من خلال تعريفه كمتغير فارغ باسم shopping_cart ليكون سلة تسوق لإضافة العناصر فيها.
- إضافة عناصر للسلة باستدعاء الدالة add_item ثلاث مرات لإضافة العناصر:
  - "Apple"
  - "Banana"
  - "Bread"

### المقطع البرمجي

```
add_item(shopping_cart, "Apple")
add_item(shopping_cart, "Banana")
add_item(shopping_cart, "Bread")
```

---

عرض محتويات سلة التسوق باستدعاء الدالة view_cart .

## مثــال ( 4 ) | دالة ترجع قيمة استكمالاً لمثال 2

**Input**

```
# تعريف دالة لجمع عددين
def add_numbers(a, b):

    return a + b  # ترجع ناتج جمع العددين

# 3 و 5 القيمتين add_numbers استدعاء الدالة
result = add_numbers( a: 5,  b: 3)  # result تخزين ناتج الجمع في المتغير

# print طباعة الناتج باستخدام دالة
print(result)  # 8 :عرض الناتج
```

**Output**

```
C:\Users\HP\Documents\projectfunction
8

Process finished with exit code 0
```

### فكرة البرنامج:

- البرنامج يقوم بتعريف دالة **add_numbers** لجمع عددين.
- يستدعي الدالة ويضيف القيم إليها ويتم تخزين الناتج في متغير ثم طباعته.

### شرح المثال للمتعلمين :

| | |
|---|---|
| تعريف الدالة: | **add_numbers** اسم الدالة<br>المعاملات (**Parameters**) التي تستقبل القيم عند استدعاء الدالة<br>**return a + b** تجمع القيمتين |
| استدعاء الدالة وتخزين الناتج: | **result = add_numbers(3 ,5)**<br>في المتغير result يتم تخزين الناتج الذي تُعيده الدالة add_numbers(5, 3)<br>القيمتين: a = 5 b = 3 |
| دالة print() لطباعة القيمة المخزنة في المتغير result | |

### التأكيد في هذا المثال على :

- **الدالة:(Function)** : تحتوي على مجموعة من التعليمات - وتعمل على إعادة استخدام التعليمة البرمجية .
- **المعاملات:(Parameters)** : متغيرات داخل الدالة تستقبل القيم عند الاستدعاء.
- **الإرجاع : (Return)** : تُعيد قيمة أو نتيجة معينة من الدالة إلى مكان استدعائها.

## تابع مثـــال ( 4 )

مثـــال إثرائـــي(5):

إنشاء دالة (بمعاملات غير محددة )، والإرجاع **return** بأكثر من متغير.

**التفسير:** برنامج ينشيء دالة تحسب المجموع الكلي والمتوسط لدرجات الطلاب المدخلة كمعاملات.

**Input**

```
# student_result تعريف الدالة
def student_result(*degrees):

    total_sum = 0  # متغير لتخزين المجموع الكلي
    number_degrees = 0  # متغير لتتبع عدد الدرجات المدخلة

    # حلقة لحساب المجموع الكلي وعدد الدرجات
    for degree in degrees:
        number_degrees += 1  # 1 زيادة عدد الدرجات بمقدار
        total_sum += degree  # إضافة الدرجة إلى المجموع الكلي

    # إرجاع المتوسط والمجموع الكلي
    return (total_sum / number_degrees), total_sum

# student_result استدعاء الدالة مع تمرير درجات الطلاب
student_degrees, total_sum = student_result( *degrees: 80, 85, 99, 70, 88)
# طباعة المجموع الكلي للدرجات
print(f"Total student degree: {total_sum}")
# طباعة متوسط الدرجات
print(f"Average student degree: {student_degrees}")
```

**Output**

```
C:\Users\HP\Documents\projectfunction\
Total student degree: 422
Average student degree: 84.4

Process finished with exit code 0
```

### فكرة البرنامج:

- إنشاء دالة **student_result** تأخذ عددًا غير محدد من درجات الطلاب كمدخلات.
- تقوم الدالة بحساب:
  1. المجموع الكلي للدرجات.
  2. متوسط الدرجات.
- **Return** تستخدم لإرجاع كل من المتوسط والمجموع الكلي كقيمتين.

## شرح المثال (5)للمتعلمين :

### تعريف الدالة:

| | |
|---|---|
| تعريف الدالة student_result : | • *degrees<br>○ النجمة (*) تستخدم لجعل الدالة تستقبل عدد غير محدود من المعاملات.<br>○ يتم تخزين جميع الدرجات المدخلة في قائمة باسم **degrees**<br>• total_sum = 0<br>○ متغير لتخزين المجموع الكلي للدرجات.<br>• number_degrees = 0<br>○ متغير لتتبع عدد الدرجات المدخلة.<br>• الحلقة (for)<br>○ المرور على كل درجة داخل قائمة degrees<br>○ تزيد الدرجات بمقدار **1** في كل دورة باستخدام **number_degrees += 1**<br>○ تضيف كل درجة إلى المجموع باستخدام **total_sum += degree**<br>• return (total_sum / number_degrees), total_sum<br>○ تعيد الدالة قيمتين:<br>— متوسط الدرجات.<br>— المجموع الكلي للدرجات. |
| student_result لاستدعاء الدالة وتخزين النتائج | |
| طباعة النتائج: | • print(f"Total student degree: {total_sum}")<br>■ طباعة المجموع الكلي للدرجات.<br>• print(f"Average student degree: {student_degrees}")<br>■ طباعة متوسط الدرجات. |

# مدخل إلى الدوال Functions

إثرائي تابع مثال(5): تطوير البرنامج: إدخال معامل الصف.
ترتيب إدخال المعاملات.

**Input**

```python
# student_result تعريف الدالة
def student_result(grade, *degrees):

    total_degrees = sum(degrees)  # حساب مجموع الكلي الدرجات
    average_degrees = total_degrees / len(degrees)  # حساب متوسط الدرجات
    return average_degrees, total_degrees, grade  # إرجاع النتائج الثلاث

# student_result استدعاء الدالة وتمرير معامل الصف والدرجات
average_degrees, total_degrees, grade = student_result( grade: 12,  *degrees: 80, 85, 99, 70, 88)

# طباعة معامل الصف
# طباعة مجموع الدرجات
print(f"Student Grade: {grade}\n Total Student Degrees: {total_degrees}")

# طباعة متوسط الدرجات
print(f"Average Student Degrees: {average_degrees}")
```

**Output**

```
C:\Users\HP\Documents\projectfunction
Student Grade: 12
 Total Student Degrees: 422
Average Student Degrees: 84.4
```

استكمالا من المثـــال السابق:

| | |
|---|---|
| نجد أن المعاملات في هذا المثال : | • **grade** معامل الصف.<br>• ***degrees** قائمة غير محدودة من الدرجات. |
| وأن الدالة | • **total_degrees = sum(degrees)** تجمع الدرجات المدخلة باستخدام الدالة المدمجة **sum**<br>• **average_degrees = total_degrees / len(degrees)** إيجاد المتوسط بقسمة المجموع الناتج على عدد الدرجات باستخدام **len** من خلال حساب عدد عناصر القائمة (list) .<br>• **return average_degrees, total_degrees, grade** تعيد الدالة القيم الثلاث :متوسط الدرجات، المجموع الكلي، و معامل الصف.<br>• **return average_degrees, total_degrees, grade** تعيد الدالة القيم الثلاث :متوسط الدرجات، المجموع الكلي، و معامل الصف. |

# مدخل إلى الدوال Functions

نشاط حياتي : إذا أردنا عد الكرات في صندوق - إمكانية الاستعانة آلة عد إلكترونية أو عدها يدويًا .

**سؤال تفكير :أن لديك قائمة بدرجات الطلاب وتحتاج إلى حساب المجموع والمتوسط. ما هي الطريقة الأسرع والأكثر تنظيما؟**

والتأكيد على المتعلم أنه يجب تفسير المشكلة وتوضيح المعطيات و ما هو مطلوب

- المطلوب صف المتعلم (Grade) أكثر من مرحلة مثال (10-11-12) وقائمة درجات (**Degrees**).
- نستخدم أدوات جاهزة (مثل '**sum**' و'**len**') لحساب المجموع والمتوسط بسهولة.
- المطلوب النتائج الثلاثة: المجموع، المتوسط، والصف.

## ربط هذا النشاط بالمثالين:

- المثال الأول: يشبه العد اليدوي (استخدام حلقة وعداد يدوي).
- المثال الثاني :يشبه استخدام آلة العد دوال جاهزة مثل **sum** و **len()**.

## المقارنة بين المثالين (6,5):

| البند | حل المثال الأول | الثاني حل المثال |
|---|---|---|
| طريقة حساب المجموع | باستخدام حلقة for | باستخدام الدالة المدمجة sum |
| طريقة حساب العدد | باستخدام عداد يدوي number_degrees. | باستخدام الدالة المدمجة len |
| الإرجاع(Return) | يرجع قيمتين: متوسط الدرجات، المجموع الكلي فقط. | يرجع ثلاثة قيم: متوسط الدرجات، المجموع الكلي، معامل الصف. |
| الإضافات | لا يتضمن معامل الصف. | يتضمن معامل الصف كجزء من الإخراج. |
| ما يميز البرنامج | استخدام الحلقات والعداد | استخدام الدوال المدمجة. |

إثرائي تابع مثال(5)

تطوير البرنامج:

ادرس التعليمات البرمجية التالية، وحدد وظيفة الجزء المحدد منها:

**Input**

```
# student_result تعريف الدالة
def student_result(*degrees):
    # حساب مجموع الدرجات
    total_degrees = sum(degrees)
    # (100 × عدد المواد) حساب أعلى درجة ممكنة
    max_total = len(degrees) * 100
    # طباعة مجموع الدرجات من أصل الدرجة الكلية
    print(f"Sum of student's degrees: {total_degrees} from {max_total} degrees")
    # حساب نسبة الدرجات
    rate = (total_degrees / max_total) * 100
    # (منزلتين عشريتين) طباعة النسبة المئوية للدرجات
    print(f"Rate of student's degrees: {rate:.2f} %")
    # تحديد حالة النجاح أو الفشل بناءً على النسبة
    print("Congratulations, You succeeded") if rate ≥ 50 else print("Sorry! You failed")
# استدعاء الدالة مع درجات الطالب
student_result( *degrees: 100, 80, 90, 95, 50.5)
```

**Output**

```
C:\Users\HP\Documents\projectfunction\.venv\Scripts\
Sum of student's degrees: 415.5 from 500 degrees
Rate of student's degrees: 83.10 %
Congratulations, You succeeded

Process finished with exit code 0
```

## شرح المثال للمتعلمين :

- حساب المجموع : (total_degrees) يتم جمع الدرجات باستخدام دالة sum.
- حساب الحد الأقصى : (max_total) يتم حساب أعلى درجة بضرب عدد المواد في 100.
- طباعة المجموع : يتم عرض المجموع الكلي للدرجات مقارنة بالحد الأقصى.
- حساب النسبة المئوية :يتم تقسيم مجموع الدرجات على الحد الأقصى وضرب الناتج في 100 للحصول على النسبة.
- طباعة النسبة :يتم تنسيق النسبة بحيث تظهر مع منزلتين عشريتين.
  - {rate:.2f} المقصود f"{value:.nf}".
  - rate المتغير الذي يحتوي على النسبة المئوية للدرجات.
  - f التنسيق خاص بالأعداد العشرية (float) .
  - (:.2f) عرض القيمة العشرية مع منزلتين عشريتين الناتج 83.10 %.
  - (:.1f) عرض القيمة العشرية مع منزلة عشرية الناتج 83.1 %.

ملاحظة : (:.0f) لإزالة الكسور العشرية (عدد صحيح) الناتج 83 %.

- تحديد النجاح أو الفشل يتم التحقق من أن النسبة أكبر من أو تساوي 50 لتحديد حالة النجاح، وإظهار الرسالة .

إثرائي تابع مثال(5) تطوير البرنامج:

## الحل المطلوب في المثال:

- طباعة النسبة المئوية للدرجات(منزلتين عشريتين)

```
print(f"Rate of student's degrees: {rate:.2f} %")
```

- تحديد حالة النجاح أو الفشل بناءً على النسبة

```
print ("Congratulations, You succeeded") if rate >= 50 else print ("Sorry! You failed")
```

إثرائي(مثال 6):

القيم الافتراضية لمعاملات الدوال **default parameter** .

**التفسير:** برنامج ينشئ دالة تتكون من معاملين أحداهما افترا ضي )في حال عدم تمرير قيمة له؛ يتم استخدام القيمة الافتراضية للمعامل).

Input

```
# تعريف دالة لإظهار اسم الطالب ودرجته
# يتم استخدام قيمة افتراضية للدرجة إذا لم يتم تمريرها
def student_grade(name, grade="Excellent"):
    # طباعة اسم الطالب والدرجة
    print(f"Hi {name} your grade is {grade}")

# استدعاء الدالة مع تمرير اسم الطالب فقط
# هنا يتم استخدام القيمة الافتراضية للدرجة وهي "Excellent"
student_grade("Ali")
# الناتج: Hi Ali your grade is Excellent

# استدعاء الدالة مع تمرير اسم الطالب والدرجة
# هنا يتم استبدال القيمة الافتراضية للدرجة بالقيمة الممررة وهي "Good"
student_grade( name: "Ali",  grade: "Good")
# الناتج: Hi Ali your grade is Good
```

Output

```
C:\Users\HP\Documents\projectfunction\
Hi Ali your grade is Excellent
Hi Ali your grade is Good

Process finished with exit code 0
```

# مدخل إلى الدوال Functions

شرح مثال(6) للمتعلمين : def student_grade(name, grade="Excellent") :

## تعريف الدالة ( student_grad):

يجب كتابة ( المعاملات الإلزامية name- بدون قيم افتراضية) أولاً

ثم (كتابة المعاملات **grade** (ذات القيم الافتراضية Excellent )) بعدها.

## الشرح:

1. المعاملات الإلزامية تأتي أولاً:
   - المعامل **name** في هذا المثال هو **معامل إلزامي**، يعني أنه يجب تمرير قيمة له عند استدعاء الدالة لاحقا
   - إذا لم يتم تمرير قيمة لهذا المعامل، سيحدث خطأ.
2. **المعاملات الافتراضية** تأتي بعد الإلزامية:
   - المعامل grade هو معامل يحتوي على قيمة افتراضية وهي "Excellent" .
   - إذا لم يتم تمرير قيمة لهذا المعامل أثناء استدعاء الدالة، ستُستخدم القيمة الافتراضية.

## دوال:Lambda

- هي **دوال مجهولة** الاسم (Anonymous Functions)تُكتب باستخدام **الكلمة المفتاحية lambda**.

إثرائي (مثال 7): إنشاء دالة **lambda** لجمع عددين.

**Input**

```
# إنشاء دالة Lambda لجمع عددين
add = lambda x, y: x + y   # الدالة تأخذ معاملين (x, y) وتعيد ناتج جمعهما

# استدعاء الدالة المجهولة Lambda وتمرير العددين 5 و 10 وطباعة النتيجة #
print(add( x: 5,  y: 10))
```

**Output**

```
C:\Users\HP\Documents\projectfunction
15

Process finished with exit code 0
```

إثرائي (مثال 8): إنشاء دالة **Lambda** لحساب الأس التربيعي لعدد.

**Input**

```
# إنشاء دالة Lambda لحساب تربيع عدد
exponention = lambda x: x ** 2  # الدالة تأخذ معاملًا واحدًا x وتعيد x مرفوعًا للأس 2

# استدعاء الدالة المجهولة وطباعة الناتج
print(exponention(5))  # يتم تمرير العدد 5 إلى الدالة
```

**Output**

```
C:\Users\HP\Documents\projectfunction\
25

Process finished with exit code 0
```

إثرائي (مثال 9): إنشاء دالة( **Lambda** )لتحديد نوع العدد (فرديًا أم زوجيًا).

**Input**

```
# إنشاء دالة Lambda لتحديد إذا كان العدد زوجيًا أو فرديًا
even_odd = lambda x: print("even") if x % 2 == 0 else print("odd")

# استدعاء الدالة مع أخذ المدخل من المستخدم
even_odd(float(input("Enter a number: ")))
```

**Output**

```
C:\Users\HP\Documents\projectfunction\
Enter a number: 9
odd

Process finished with exit code 0
```

مثال (10):

مجال المتغير المحلي في الدالة:(Local Variable) .

**التفسير:** عدم إمكانية استخدام المتغير المحلي local_var خارج نطاق الدالة وظهور رسالة الخطأ (المتغير غير معرف)

**Input**

```
1  def my_function():
2      local_var = 10  # تعريف متغير محلي
3      print (local_var)
4
5  my_function()
6  print(local_var)  # خطأ: المتغير غير معرف خارج الدالة
```

**Output**

```
C:\Users\HP\Documents\projectfunction\.venv\Scripts\python.exe C:\Users\HP\Documents\projectfunction\
10
Traceback (most recent call last):
  File "C:\Users\HP\Documents\projectfunction\.venv\Scripts\ex10.py", line 6, in <module>
    print(local_var)  # خطأ: المتغير غير معرف خارج الدالة
          ^^^^^^^^^
NameError: name 'local_var' is not defined

Process finished with exit code 1
```

## شرح المثال للمتعلمين :

### داخل الدالة : my_function

- يتم تعريف المتغير local_var د اخل الدالة .my_function()
- المتغير هنا يُعتبر محليًا (Local Variable) حيث تعريفه داخل الدالة فقط.
- عند استدعاء my_function()، يتم تنفيذ التعليمة البرمجية داخل الدالة، وطباعة قيمة local_var التي هي 10

### خارج الدالة:

- عند محاولة طباعة local_var الموجودة في السطر (7) خارج الدالة .my_function
- يظهر خطأ لأن المتغير local_var لا يمكن الوصول إليه لأنه خارج الدالة.

مثال (11):

مجال المتغير العام في الدالة:(Global Variable) .

التفسير: يتم تعريف المتغير العام global_var حيث يمكن استخدامه خارج نطاق الدالة.

Input

```
def my_function():
    local_var = 10  # تعريف متغير محلي داخل الدالة
    print(local_var)  # طباعة قيمة المتغير المحلي

    global global_var  # الإشارة إلى أن المتغير global_var هو متغير عام
    global_var = 500  # تعديل أو إنشاء المتغير العام داخل الدالة


my_function()

# طباعة قيمة المتغير العام بعد تعديلها داخل الدالة
print(f"Global Variable is: {global_var}")
```

Output

```
C:\Users\HP\Documents\projectfunction\
10
Global Variable is: 500

Process finished with exit code 0
```

شرح المثال للمتعلمين :

- تعريف الدالة :my_function تضمنت تعليمات لتعريف متغير محلي وآخر عام.
- local_var هو متغير محلي تم تعريفه داخل الدالة يمكن الوصـول إلى local_var فقط داخل نطاق الدالة.
- :global_var تم إنشاؤه كمتغير عام باستخدام الكلمة المفتاحية global وتعيين قيمته 500.

مثـــال( 12):

### متتالية فيبوناتشي والنسبة الذهبية:

النسبة الذهبية، والمعروفة أيضًا باسم الرقم الذهبي أو النسبة الذهبية أو النسبة الإلهية، هي نسبة بين رقمين تســاوي تقريبًا 1.618. وعادةً ما تُكتب بالحرف اليوناني **Φ**، وهي مرتبطة ارتباطًا وثيقًا بمتتالية فيبوناتشــي، . وهي ســلســلة من الأرقام حيث يضــاف كل رقم إلى الأخير قبله .أرقام فيبوناتشــي هي 0 ، 1 ، 1 ، 2 ، 3 ، 5 ، 8 ، 13 ، 21 ، وهكذا مع نســبة كل رقم والرقم الســابق يقترب تدريجياً من 1.618، أو فاي.(**Φ**) .

البرنامج التالي يوضـح كيفية الحصـول على سـلسـلة فيبوناتشـي مع حسـاب النسـبة الذهبية، قدم حلول برمجية بديلة للحصول على سلسلة فيبوناتشي وحساب النسبة الذهبية.

**Input**

```python
def fibonacci(number_fibonacci):  # تعريف دالة لتوليد متتالية فيبوناتشي وحساب النسبة الذهبية
    number_1 = 0  # الرقم الأول في متتالية فيبوناتشي
    number_2 = 1  # الرقم الثاني في متتالية فيبوناتشي
    sum_number = 1  # الرقم الثالث في المتتالية (مجموع الرقمين الأولين)

    if number_fibonacci > 3:  # التحقق من أن عدد العناصر أكبر من 3
        print(number_1, number_2, sum_number, end=" ")  # طباعة الأرقام الثلاثة الأولى في المتتالية

        # for توليد الأرقام الباقية باستخدام حلقة
        for i in range(3, number_fibonacci):  # بدءًا من الرقم الرابع وحتى العدد المطلوب
            # تحديث قيم الأرقام بناءً على قواعد متتالية فيبوناتشي
            number_1, number_2 = number_2, sum_number  # تبديل القيم
            sum_number = number_1 + number_2  # حساب الرقم التالي
            print(sum_number, end=" ")  # طباعة الرقم الجديد

        # حساب النسبة الذهبية
        golden_ratio = sum_number / number_2  # قسمة الرقم الأكبر على الرقم الذي يسبقه
        print(f"\nThe Golden Ratio is: {golden_ratio:.2f}")  # طباعة النسبة الذهبية
    else:
        print("Enter number more than 3")  # طباعة رسالة إذا كان العدد المدخل أقل من 3

```

**Output**

```
C:\Users\HP\Documents\projectfunction\.venv\Scripts\python.exe
Enter number of Fibonacci Sequence: 6
0 1 1 2 3 5
The Golden Ratio is: 1.67

Process finished with exit code 0
```

## شرح المثال للمتعلمين :

- كيفية توليد متتالية فيبوناتشي:
- إذا أردنا توليد أول 6 أرقام من المتتالية: 0 ، 1 ، 1 ، 2 ، 3 ، 5
- الرقم الأول 0
- الرقم الثاني 1
- الرقم الثالث 0 + 1 = 1
- الرقم الرابع 1 + 1 = 2
- الرقم الخامس 1 + 2 = 3
- الرقم السادس 2 + 3 = 5
- الرقم السابع 3 + 5 = 8
- تخصيص القيمة 0 للمتغير n
- التأكيد متتالية فيبوناتشي تبدأ دائمًا بـ 0 و 1
- وضع شرط للمستخدم ان يجب عدد العناصر المطلوبة في السلسلة أكبر من 3
- إذا كان العدد أقل من أو يساوي 3، يتم طباعة رسالة للمستخدم

```
print("Enter number more than 3")
```

النسبة الذهبية تُحسب باستخدام آخر رقمي وهي تقريبًا تساوي :

$$1{,}67 \approx \frac{5}{3} = \frac{\text{الأخير الرقم}}{\text{يسبقه الذي الرقم}} = \text{Golden Ratio}$$

```
golden_ratio = sum_number / number_2

print(f"\nThe Golden Ratio is: {golden_ratio:.2f}")
```

- print(number_1, number_2, sum_number, end="  ")
- طباعة أول ثلاثة أرقام مباشرة 0 ، 1 ،1
- استخدام end=" " لطباعة الأرقام في نفس السطر مع مسافات بينها.

# مدخل إلى الدوال Functions

حل ورقة عمل 1 ص 81 .

اسم البرنامج: آلة حاسبة بسيطة باستخدام الدوال.

مشكلة البرنامج: توظيف الدوال - لإجراء العمليات الحسابية بين عددين مدخلين من قبل المستخدم.

المطلوب: استدعاء الملف Simple_Calc.py من مجلد أوراق العمل.

**Input**

```python
# تعريف دالة الجمع
def addition(a, b):
    return a + b  # تُرجع ناتج جمع العددين
# تعريف دالة الطرح
def subtraction(a, b):
    return a - b  # تُرجع ناتج طرح العدد الثاني من الأول
# تعريف دالة الضرب
def multiplication(a, b):
    return a * b  # تُرجع ناتج ضرب العددين
# تعريف دالة القسمة
def division(a, b):
    if b == 0:  # التحقق من أن القاسم ليس صفرًا
        return "Sorry, Cannot divide by zero!!"  # رسالة خطأ إذا كان القاسم صفرًا
    return a / b  # تُرجع ناتج القسمة
# تعريف دالة الحاسبة الرئيسية
def simple_calculator():
    # طلب المدخلات من المستخدم
    number1 = float(input("Enter first number: "))  # إدخال العدد الأول
    number2 = float(input("Enter second number: "))  # إدخال العدد الثاني
    # طلب العملية الحسابية المطلوبة
    user_input = input("Enter your choice (+, -, *, /): ")  # إدخال نوع العملية
    # تنفيذ العملية بناءً على اختيار المستخدم
    if user_input == "+":
        result = addition(number1, number2)  # استدعاء دالة الجمع
    elif user_input == "-":
        result = subtraction(number1, number2)  # استدعاء دالة الطرح
    elif user_input == "*":
        result = multiplication(number1, number2)  # استدعاء دالة الضرب
    elif user_input == "/":
        result = division(number1, number2)  # استدعاء دالة القسمة
    else:
        # رسالة إذا كانت العملية المدخلة غير صحيحة
        print("Invalid input.")
        exit()  # إنهاء البرنامج
    # طباعة النتيجة النهائية
    print(result)
# استدعاء دالة الحاسبة الرئيسية لتشغيل البرنامج
simple_calculator()
```

**Output**

```
C:\Users\HP\Documents\projectfunction\
Enter first number: 43
Enter second number: 5
Enter your choice :(+, -, *, /): /
8.6

Process finished with exit code 0
```

# مدخل إلى الدوال Functions

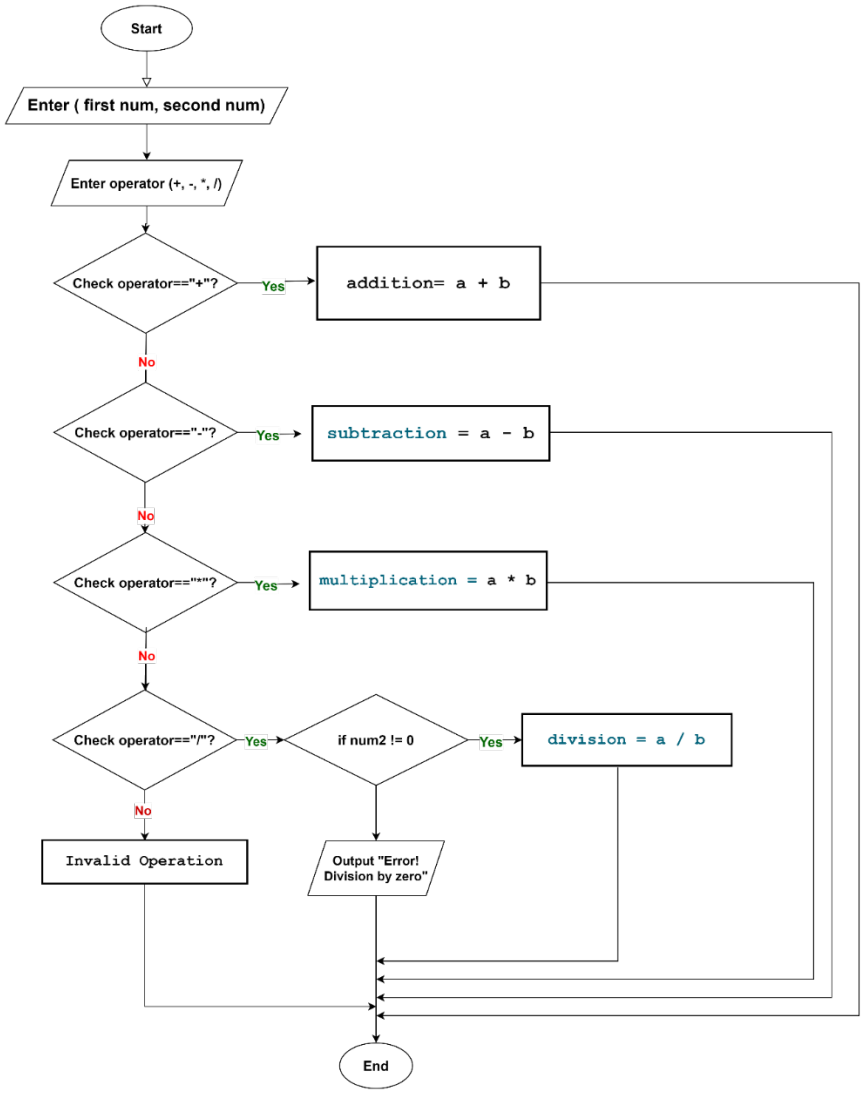
## شرح ورقة العمل للمتعلمين :

- التأكيد على تعريف الدوال الحسابية  تعريف أربع دوال، كل دالة مسؤولة عن عملية حسابية واحدة.
- دالة الحاسبة الرئيسية:(Simple Calculator)

- def simple_calculator:()

- جمع المدخلات من المستخدم، تحديد العملية المطلوبة، وتنفيذ العملية باستخدام الدوال المعرفة مسبقًا.
- باستخدام تعليمات الشرط :if-elif-else
  - التحقق من رمز العملية المدخل.
  - استدعاء الدالة المناسبة بناءً على العملية.
  - إذا كان الإدخال غير صحيح، يتم طباعة رسالة خطأ والخروج من البرنامج.
- التأكد من ظهور رسالة تنبيه في حال القسمة على صفر.

## خريطة التدفق



إثرائي: تطوير البرنامج:

- تطوير البرنامج بحيث يعمل بشكل مستمر.
- إيقاف تشغيل البرنامج بطريقة صحيحة (الخروج من التكرار).
- التأكد من إدخال المستخدم لقيم عددية فقط.

Input

Reader Mode

```
# تعريف دالة الجمع
def addition(a, b):
    return a + b
# تعريف دالة الطرح
def subtraction(a, b):
    return a - b
# تعريف دالة الضرب
def multiplication(a, b):
    return a * b
# تعريف دالة القسمة
def division(a, b):
    if b == 0:  # التحقق من أن القاسم ليس صفرًا
        return "Sorry, Cannot divide by zero!!"
    return a / b
# تعريف دالة الحاسبة الرئيسية
def simple_calculator():
    while True:  # حلقة تعمل بشكل مستمر
        try:
            # إدخال العدد الأول
            user_input1 = input("Enter first number (or type 'exit' to quit): ")
            if user_input1.lower() == "exit":  # التحقق من الخروج
                print("Exiting the calculator. Goodbye!")
                break
            number1 = float(user_input1)  # تحويل الإدخال إلى عدد
            # إدخال العدد الثاني
            user_input2 = input("Enter second number (or type 'exit' to quit): ")
            if user_input2.lower() == "exit":  # التحقق من الخروج
                print("Exiting the calculator. Goodbye!")
                break
            number2 = float(user_input2)  # تحويل الإدخال إلى عدد
            # إدخال العملية الحسابية
            operation = input("Enter an operation (+, -, *, /) or type 'exit' to quit: ")
            if operation.lower() == "exit":  # التحقق من الخروج
                print("Exiting the calculator. Goodbye!")
                break
            # تنفيذ العملية بناءً على اختيار المستخدم
            if operation == "+":
                result = addition(number1, number2)
            elif operation == "-":
                result = subtraction(number1, number2)
            elif operation == "*":
                result = multiplication(number1, number2)
            elif operation == "/":
                result = division(number1, number2)
            else:
                print("Invalid operation. Please try again.")
                continue
            # طباعة النتيجة النهائية
            print(f"Result: {result}")
        except ValueError:
            print("Invalid input! Please enter numeric values only.")
# استدعاء دالة الحاسبة الرئيسية لتشغيل البرنامج
simple_calculator()
```

# مدخل إلى الدوال Functions

| البرنامج الأول | البرنامج الثاني |
|---|---|
| 1. أدخال قيمة العدد مباشرة<br>number1 = float(input("Enter first number: "))<br>number2 = float(input("Enter second number: ")) | 1. إضافة خيار الخروج أثناء إدخال القيمة العدد وذلك بإضافة شرط<br>user_input1 = input("Enter first number (or type 'exit' to quit): ")<br>if user_input1.lower() == "exit:"<br>print("Exiting the calculator. Goodbye!")<br>break<br>number1 = float(user_input1)<br>==========<br>user_input2 = input("Enter second number (or type 'exit' to quit): ")<br>if user_input2.lower() == "exit":<br>print("Exiting the calculator. Goodbye!")<br>break<br>number2 = float(user_input2 |
| 2. لا يعمل في حلقة، يتم إنهاء التنفيذ مباشرة بعد إتمام عملية واحدة أو إدخال خاطئ، يحتاج المستخدم لإعادة تشغيل البرنامج يدويًا. | 2. حلقة مستمرة باستخدام while True، الهدف إجراء عمليات عدة دون إعادة تشغيل البرنامج. |
| 3. لا يوجد معالجة للأخطاء في حالة الخطأ بإدخال قيمة غير عددية، سيؤدي إلى تعطل البرنامج . | 3. معالجة للأخطاء باستخدام try/except:<br>Try:<br>..... مقاطع برمجية<br>except ValueError:<br>print("Invalid input! Please enter numeric values only.") |
| **مزايا البرنامج :** بسيط والسريع – غير مرن – لا يعالج الأخطاء. | **مزايا البرنامج :** التنظيم – مرن - معالجة الأخطاء. |

إثرائي: ورقة عمل 2 | اسم البرنامج: اكتشاف المواقع غير الموثوقة

Input

```
# قائمة المواقع المشبوهة
suspicious_domains = ["hacker.com", "fake.com", "phishingsite.com"]
# دالة لعرض قائمة المواقع المشبوهة
def view_suspicious(suspicious_domains):
    print(suspicious_domains)  # طباعة النطاقات المشبوهة
# دالة لإضافة نطاق جديد إلى قائمة المواقع المشبوهة
def add_suspicious(suspicious_domains, email):
    domain = email.split('@')[-1]  # استخراج النطاق (domain) من البريد الإلكتروني
    if domain not in suspicious_domains:  # التحقق إذا كان النطاق غير موجود في القائمة
        suspicious_domains.append(domain)  # إضافة النطاق إلى القائمة
        print(f"{domain} added")  # إعلام المستخدم بأنه تمت الإضافة
    else:
        print(f"{domain} already found")  # إعلام المستخدم بأن النطاق موجود مسبقًا
# الحلقة الرئيسية (Main Loop)
while True:
    # عرض الخيارات المتاحة للمستخدم
    print("\n[1] Detect & Add Email\n[2] Print Domain List\n[3] Exit")
    choice = input("Enter Your Choice: ")  # طلب إدخال الخيار من المستخدم
    # إذا اختار المستخدم الخيار الأول
    if choice == "1":
        email = input("Enter Email: ")  # طلب إدخال بريد إلكتروني
        if "@" in email:  # التحقق من صحة البريد الإلكتروني بوجود علامة @
            add_suspicious(suspicious_domains, email)  # إضافة النطاق إلى القائمة إذا كان البريد صحيحًا
        else:
            print("Invalid email. Please include '@'.")  # رسالة خطأ إذا كان البريد غير صالح
    # إذا اختار المستخدم الخيار الثاني
    elif choice == "2":
        view_suspicious(suspicious_domains)  # عرض قائمة النطاقات المشبوهة
    # إذا اختار المستخدم الخروج
    elif choice == "3":
        break  # إنهاء البرنامج

    # إذا أدخل المستخدم خيارًا غير صحيح
    else:
        print("Wrong choice.")  # رسالة خطأ إذا كان الخيار غير صحيح
```

## أجب عن الأسئلة التالية:

**ما الغرض من البرنامج السابق؟**

الغرض من البرنامج هو:

a. إدارة قائمة تحتوي على النطاقات المشبوهة (Suspicious Domains).

السماح للمستخدم بالقيام بما يلي:

1. التحقق من نطاق البريد الإلكتروني المدخل لمعرفة إذا كان مشبوهًا.
2. إضافة النطاقات الجديدة المستخرجة من عناوين البريد الإلكتروني إلى القائمة إذا لم تكن موجودة مسبقًا.
3. عرض قائمة النطاقات المشبوهة.
4. الخروج من البرنامج بشكل سهل.

## تابع - إثرائي: ورقة عمل 2

b. سجل التعليمات البرمجية المناسبة لتنفيذ المهام التالية:لتعليمات البرمجية المناسبة لتنفيذ المهام:

إذا تم إدخال بريد إلكتروني غير صحيح.

```
email = input("Enter Email: ")
    if "@" in email:
        add_suspicious(suspicious_domains, email)
    else:
print("Invalid email. Please include '@'.")
```

c. إضافة بريد إلكتروني مدخل من المستخدم إلى قائمة البريد الإلكتروني

```
def add_suspicious(suspicious_domains, email):
    domain = email.split
            if domain not in suspicious_domains :
                suspicious_domains.append(domain)
                print(f"{domain} added")
            else:
print(f"{domain} already found")
```

d. ما ناتج تنفيذ التعليمات البرمجية التالية:

```
add_suspicious(suspicious_domains, email):
```

يتم التحقق من النطاق المستخرج من البريد الإلكتروني

```
domain = email.split(@)[-1]
```

استخراج النطاق (domain) من البريد الإلكتروني

```
if domain not in suspicious_domains:
```

التحقق ا ذا كان النطاق غير موجود في قائمة

```
suspicious_domains.append(domain)
```

يتم إضافته إلى القائمة

```
print(f"{domain} added")
```

يتم طباعة رسالة."{domain} added" :

```
else:
  print(f"{domain} already found")
```

إعلام المستخدم بأن النطاق موجود مسبقًا

```
def view_suspicious(suspicious_domains)
    print(suspicious_domains)
```

وظيفة الدالة طباعة جميع النطاقات الموجودة حاليًا في قائمة suspicious_domains

# توظيف الدوال في الواجهة الرسومية:

## التأكيد أثناء شرح كيفية توظيف الدوال في الواجهة الرسومية .

- الدوال هي مجموعة تعليمات برمجية تنفذ مهام محددة.
- في الواجهات الرسومية(GUI) ، تُستخدم الدوال لتصميم الواجهة والتفاعل مع المستخدم.

تتضمن هذه الوظائف:

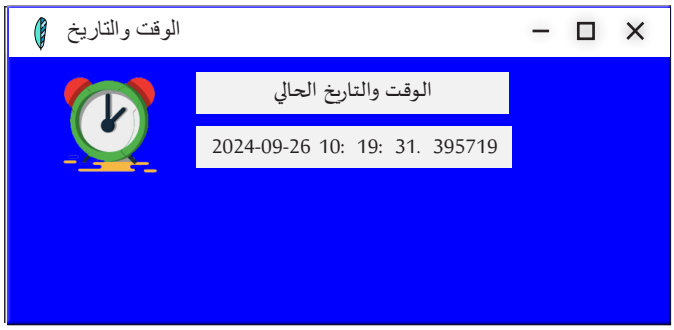
- إنشاء النافذة الرئيسية.
- إضافة عناصر رسومية.(Widgets)
- تعديل خصائص النافذة.

أولًا: تعديل خصائص النافذة الرئيسية

النافذة الرئيسية هي أول عنصر يظهر للمستخدم عند تشغيل تطبيق رسومي(GUI)

- تعديل خصائص النافذة مثل :
  - الحجم :(Size) عرض وارتفاع النافذة بالبكسل.
  - لون الخلفية :(Background Color) اللون الذي يظهر خلف العناصر داخل النافذة.

ICT_KW.window(size=(500, 150), background_color=(0, 0, 255))




| | |
|---|---|
| 1 | size=(500, 150) حجم النافذة:<br>○ العرض 500 (Width) بكسل.<br>○ الارتفاع 150 (Height) بكسل. |
| 2 | لون خلفية النافذة background_color =(255 ,0 ,0)<br>يحدد لون خلفية النافذة باستخدام نظام الألوان: RGB |
| 3 | الناتج: لون خلفية النافذة يصبح أزرق بالكامل. |
| 4 | بعد التعرف على جدول الألوان ص 87 . |

## نشـــاط:

ما الذي يحدث إذا قمنا بتغيير القيم في background_color إلى:

- (255, 0, 0)؟
- (0, 255, 0)؟
- (255, 255, 0)؟

## نشاط عملي :

ماذا يحدث ؟

```
ICT_KW.window(size=(600, 300), background_color=(255, 255, 0),
title="Experiment with Colors")
```

# توظيف الدوال في الواجهة الرسومية:

إثرائي: ورقة عمل 2 | اسم البرنامج: اكتشاف المواقع غير الموثوقة

ما الغرض من البرنامج السابق؟

الغرض إدارة قائمة من النطاقات المشبوهة. - السماح للمستخدم بإضافة نطاقات جديدة ، مستخرجة من عناوين البريد الإلكتروني - عرض قائمة النطاقات الحالية - وفير طريقة تفاعلية للمستخدم للتحكم في القائمة.

سجل التعليمات البرمجية المناسبة لتنفيذ المهام التالية:

إذا تم إدخال بريد إلكتروني غير صحيح:

عرض رسالة للمستخدم البريد الإلكتروني غير صالح.

```
:if "@" not in email
  print("Invalid email. Please include '@'.")
```

إضافة بريد إلكتروني مدخل من المستخدم إلى قائمة البريد الإلكتروني:

```
email = input("Enter Email: ")
if "@" in email:
  domain = email.split[1-]("@")
  if domain not in suspicious_domains:
    suspicious_domains.append(domain)
    print(f"{domain} added")
  else:
    print(f"{domain} already found")
else:
  print("Invalid email. Please include '@'.")
```

ما ناتج تنفيذ التعليمات البرمجية التالية:

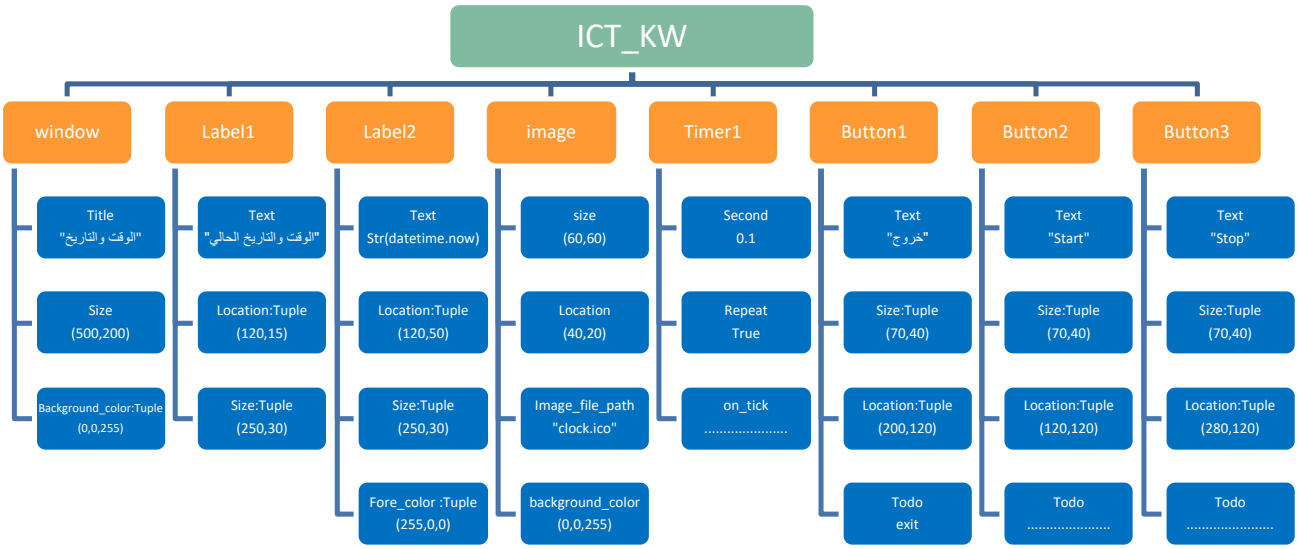
```
    def add_suspicious(suspicious_domains, email):
      domain = email.split #استخراج النطاق من البريد الإلكتروني  [1-]('@')
      if domain not in suspicious_domains:
        suspicious_domains.append(domain)l # إضافة النطاق إذا لم يكن موجودً
        print(f"{domain} added") #طباعة رسالة تفيد بإضافة النطاق
      else:
        print(f"{domain} already found")  # طباعة رسالة إذا كان النطاق موجودًا مسبقًا
    def view_suspicious(suspicious_domains):
      print(suspicious_domains)  #  عرض قائمة النطاقات المشبوهة
```

# توظيف الدوال في الواجهة الرسومية:

ورقة عمل 1:

صمم واجهة المستخدم الرسومية السابقة بكتابة التعليمات البرمجية الخاصة بإنشاء عناصرها مستعيناً

بالمخطط التالي:



## تابع ورقة عمل 1:

### Input

```python
from ICT_KW import window, new_Label, new_Button, new_Image, new_Timer
from datetime import datetime

# إنشاء النافذة الرئيسية
main_window = window(
    title="الوقت والتاريخ",  # عنوان النافذة
    size=(500, 200),         # حجم النافذة
    background_color=(0, 0, 255)  # اللون الخلفي (أزرق)
)
# إضافة النصوص
Label1 = new_Label(
    text="الوقت والتاريخ الحالي",  # النص الأول
    location=(120, 15),            # الموقع
    size=(250, 30)                 # الحجم
)
Label2 = new_Label(
    text=str(datetime.now()),      # النص الثاني: الوقت والتاريخ الحالي
    location=(120, 50),            # الموقع
    size=(250, 30),                # الحجم
    fore_color=(255, 0, 0)         # اللون الأمامي للنص (أحمر)
)
# إضافة صورة (المنبه)
clock_image = new_Image(
    image_file_path="clock.ico",   # ملف الصورة (تأكد من وجوده في نفس المجلد)
    size=(60, 60),                 # حجم الصورة
    location=(40, 20),             # موقع الصورة
    background_color=(0, 0, 255)   # اللون الخلفي للصورة (أزرق)
)
# دالة لتحديث الوقت والتاريخ
def update_time():
    """
    تقوم هذه الدالة بتحديث النص في Label2 إلى الوقت والتاريخ الحالي.
    """
    Label2.edit(text=str(datetime.now()))
# إعداد Timer لتحديث الوقت كل 0.1 ثانية
timer = new_Timer(
    second=0.1,      # التحديث كل 0.1 ثانية
    repeat=True,     # التكرار مستمر
    on_tick=update_time  # الدالة التي يتم تنفيذها في كل تحديث
)
# دالة لبدء تشغيل المؤقت
def start_timer():
    """
    تقوم هذه الدالة ببدء تشغيل المؤقت.
    """
    timer.start()
# دالة لإيقاف تشغيل المؤقت
def stop_timer():
    """
    تقوم هذه الدالة بإيقاف تشغيل المؤقت.
    """
    timer.stop()
# دالة لإنهاء البرنامج
def exit_program():
    """
    تقوم هذه الدالة بإغلاق النافذة الرئيسية.
    """
    main_window.close()
# إضافة الأزرار
Button1 = new_Button(
    text="خروج",          # نص الزر
    size=(70, 40),        # حجم الزر
    location=(200, 120),  # موقع الزر
    todo=exit_program     # الإجراء عند الضغط على الزر
)
Button2 = new_Button(
    text="Start",         # نص الزر
    size=(70, 40),        # حجم الزر
    location=(120, 120),  # موقع الزر
    todo=start_timer      # الإجراء عند الضغط على الزر
)

Button3 = new_Button(
    text="Stop",          # نص الزر
    size=(70, 40),        # حجم الزر
    location=(280, 120),  # موقع الزر
    todo=stop_timer       # الإجراء عند الضغط على الزر
)

# تشغيل النافذة
main_window.show()
```

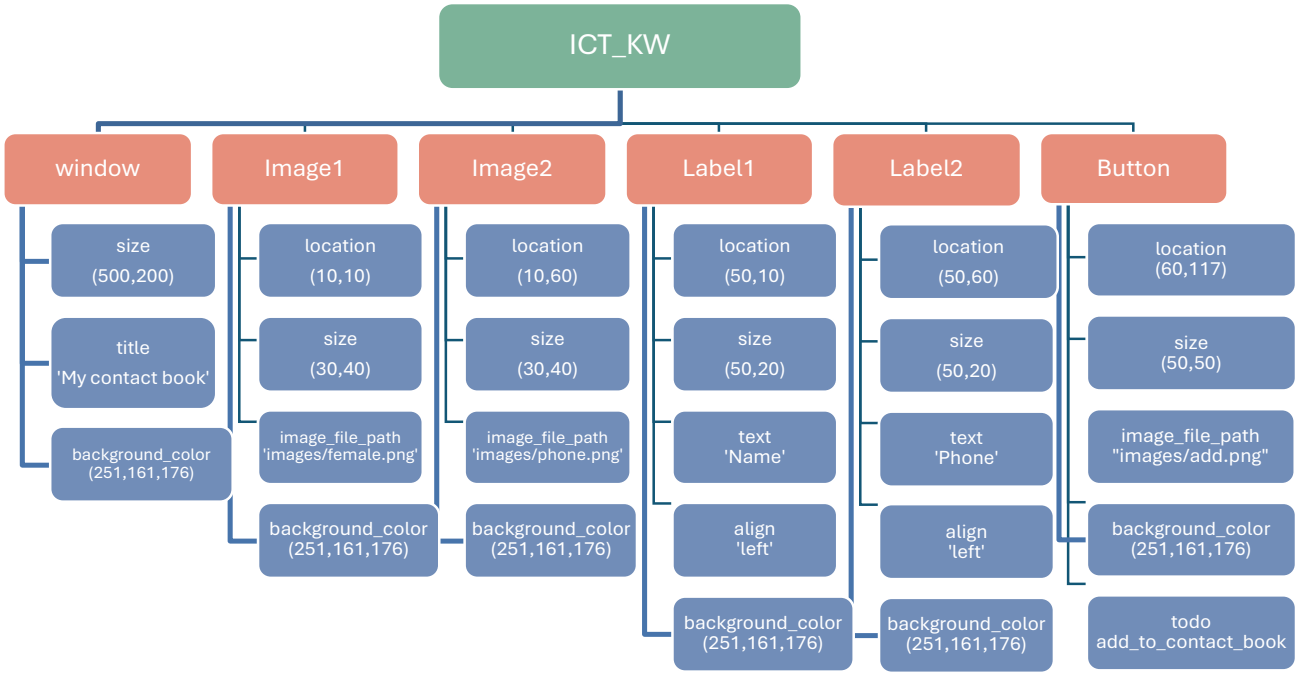
ورقة عمل 2:

اسم البرنامج: برنامج«My Contact Book»

افتح برنامج «PyCharm» ثم افتح المشروع «My Contact Book 2» ثم نفذ ما يلي:

- استدع مكتبة ICT_KW
- أعلن عن دالة باسم «add_to_contact_book» خالية من المعاملات، لطباعة رسالة«button»
- clicked«عند تفعيل الدالة.
- تصميم الواجهة الرسومية:

اكتب التعليمات المناسبة لإنشاء الواجهة الرسومية مستعيناً بالمخطط التالي:



## تابع ورقة عمل 2:

Input

```
from ICT_KW import window, new_Label, new_Button, new_Image

# "Add Contact" تعريف الدالة الخاصة بزر
def add_to_contact_book():
    """
    تُظهر رسالة عند الضغط على الزر.
    """
    print("Button clicked")
# إنشاء النافذة الرئيسية
main_window = window(
    title="My contact book",          # عنوان النافذة
    size=(500, 200),                  # حجم النافذة
    background_color=(251, 161, 176) # اللون الخلفي للنافذة
)
# (Image1) إضافة الصورة الأولى
Image1 = new_Image(
    image_file_path="images/female.png",  # مسار الصورة الأولى
    size=(30, 40),                        # حجم الصورة
    location=(10, 10),                    # موقع الصورة
    background_color=(251, 161, 176)      # اللون الخلفي للصورة
)
# (Image2) إضافة الصورة الثانية
Image2 = new_Image(
    image_file_path="images/phone.png",   # مسار الصورة الثانية
    size=(30, 40),                        # حجم الصورة
    location=(10, 60),                    # موقع الصورة
    background_color=(251, 161, 176)      # اللون الخلفي للصورة
)
# (Label1) إضافة النص الأول
Label1 = new_Label(
    text="Name",                          # النص الأول
    location=(50, 10),                    # موقع النص
    size=(50, 20),                        # حجم النص
    align="left",                         # محاذاة النص إلى اليسار
    background_color=(251, 161, 176)      # اللون الخلفي للنص
)
# (Label2) إضافة النص الثاني
Label2 = new_Label(
    text="Phone",                         # النص الثاني
    location=(50, 60),                    # موقع النص
    size=(50, 20),                        # حجم النص
    align="left",                         # محاذاة النص إلى اليسار
    background_color=(251, 161, 176)      # اللون الخلفي للنص
)
# (Button) إضافة زر
Button = new_Button(
    image_file_path="images/add.png",     # مسار صورة الزر
    size=(50, 50),                        # حجم الزر
    location=(60, 117),                   # موقع الزر
    background_color=(251, 161, 176),     # اللون الخلفي للزر
    todo=add_to_contact_book              # الإجراء عند الضغط على الزر
)
# تشغيل النافذة
main_window.show()
```

05

# الوحدة الأولى:برمجة Python

## القواميس Dictionaries

# القواميس Dictionaries

القاموس هو نوع من البيانات في بايثون يسمح لك بتخزين البيانات في أزواج من المفاتيح والقيم. المفاتيح هي عادةً سلاسل أو أعداد صحيحة، بينما يمكن أن تكون القيم أي نوع من البيانات.

يستخدم القاموس في مجموعة متنوعة من التطبيقات، بما في ذلك:

- تخزين البيانات المرتبطة، مثل قائمة بأسماء الطلاب ودرجاتهم.
- إنشاء فهارس البيانات، مثل قائمة بأسماء الكتب ومؤلفيها.
- إنشاء نماذج البيانات، مثل خريطة لمدينة مع أسماء الشوارع والأرقام.
- إنشاء قواعد البيانات الصغيرة، مثل قائمة بالعملاء وحساباتهم المصرفية.

يمكن الوصول إلى عناصر القاموس باستخدام مفتاحها.

## نتاجات التعلم :

| | |
|---|---|
| 1 | التعرف على مفهوم القواميس. |
| 2 | التعرف على خصائص القواميس. |
| 3 | إنشاء القاموس **Dictionary** . |
| 4 | التعرف على طرق انشاء القاموس. |
| 5 | التعرف على كيفية الوصول الى عنصر بالقاموس عن طريق المفتاح الخاص به. |
| 6 | التعامل مع عناصر القاموس **( الحذف، الإضافة، والتعديل)**. |
| 7 | التعرف على حذف القاموس بالكامل. |
| 8 | التعرف على طرق حذف عنصر في القاموس. |
| 9 | التعرف على تكرار عبر عناصر القاموس. |
| 10 | طباعة المفاتيح والقيم للقاموس. |

## المصطلحات : القواميس Dictionaries

القواميس Dictionaries

الأقواس المعقوفة Curly Brac {} .

البيانات الغير الأولية Non-Primiti

المفاتيح Key

القيم Value

القوائم List

الصفوف Tup

المجموعات Sets

عناصر غير مرتبة Not Ordered

فريدة Unique

غير قابلة للتعديل Immutab

صيغة القاموس Dictionary Syntax

الدالة dic

# مدخل الى القواميس Dictionaries

القاموس هو نوع من البيانات في بايثون يسمح بتخزين البيانات في أزواج من المفاتيح والقيم. المفاتيح هي عادةً سلاسل أو أعداد صحيحة، بينما يمكن أن تكون القيم أي نوع من البيانات.

يستخدم القاموس في مجموعة متنوعة من التطبيقات، بما في ذلك:

- تخزين البيانات المرتبطة، مثل قائمة بأسماء الطلاب ودرجاتهم.
- إنشاء فهارس البيانات، مثل قائمة بأسماء الكتب ومؤلفيها.
- إنشاء نماذج البيانات، مثل خريطة لمدينة مع أسماء الشوارع والأرقام.
- إنشاء قواعد البيانات الصغيرة، مثل قائمة بالعملاء وحساباتهم المصرفية.

يمكن الوصول إلى عناصر القاموس باستخدام مفتاحها.

- تعتبر القواميس **Dictionaries** في بايثون مرنة جدًا، حيث يمكن استخدامها لحفظ مختلف أنواع البيانات، مثل النصوص والأرقام والقوائم وحتى الأشياء المعقدة مثل الدوال والكائنات **Objects** .

مثـــال:

```
Capitals={' France':'Paris' ,'Germany':'Berlin','Spain': 'Madred' }
```

## خصائص القواميس

1- في القاموس **Dictionary** المفاتيح **Keys** فريدة لايمكن أن تكون متكررة. وغير قابلة للتغير لهذا يمكن استخدام أنواع بيانات غير القابلة للتغيير مثل **float** و **frozenset** و **int** و **str** و **tuple** كمفاتيح في الـ **Dictionary**، لكن الأنواع القابلة للتغيير مثل **dir** و **list** و **set** لا يمكن استخدامها.

2- في القاموس **Dictionary** القيم **Values** يمكن أن تكون مكررة ومن أي نوع من أنواع البيانات مثلاً **Numbers, Strings, Lists, Sets, Dictionaries, Functions** وما إلى ذلك.

3- في Python يمكن مقارنة أنواع القواميس Dictionaries باستخدام عوامل المقارنة المعيارية للمساواة **( == و !=)** و تطبيق المقارنات للعناصر عنصرًا تلو الآخر (وبشكل متكرر للعناصر المتداخلة مثل المجموعات أو القواميس داخل القواميس). اما عناصر المقارنة التكافئية **(<، = <، =>)** غير مدعومة لأنها لا معنى لها للمجموعات غير المرتبة مثل القواميس **Dictionaries**.

# التعامل مع القواميس Dictionaries

## 1-إنشاء القواميس Dictionaries في Python

في **Python** يمكننا إنشاء قاموس **Dictionary** من خلال تضمين قائمة مفصولة بفواصل " , " لأزواج من القيمة **Value** ومفتاح **Key** بأقواس معقوفة .{...} **Curly braces** وتفصل النقطتان " : " **Colon** كل مفتاح عن القيمة المرتبطة به.

مثـــال(1) :

```
dir0 = {id: 1, 'name':'Nora', 'size': 37}
```

يمكن ايضاً استخدام الدالة **dict ()** لإنشاء **Dictionary** من خلال مجموعة متغيرات أو **List** أو **Tuple**

مثـــال(2):

**Input**

```
d1 = dict(id=1, name="Noor", size=37)
d2 = dict({"id": 1, "name": "Noor", "size": 37})
print(d1)
print(d2)
```

**Output**

```
C:\Users\HP\AppData\Local\Microsoft\WindowsApps\python3.12.exe D:\dictionaries\n.py
{'id': 1, 'name': 'Noor', 'size': 37}
{'id': 1, 'name': 'Noor', 'size': 37}

Process finished with exit code 0
```

ملاحظة :

ويعبر عن القاموس **Dictionary** الفارغ بدون أي عناصر مكتوب قوسين فقط ، على النحو التالي: .{..}

# التعامل مع القواميس Dictionaries

## 2. الوصول إلى العناصر ( قيمة العنصر)

توفر بايثون عدد من الدوال للوصل الى عناصر القواميس **Dictionaries** حيث يتم الوصول إلى جميع القيم الموجودة في القاموس **Dictionary** باستخدام الدالة **values()** بينما يتم الوصول الى المفاتيح بواسطة الدالة **Keys()**

مثـال(1):

**Input**

```
Capitals={"France ":"Paris","Germany":"Berlin","Spain": "Madrid"}
print(Capitals["Germany"])
```

**Output**

```
C:\Users\HP\AppData\Local\Microsoft\WindowsApps\python3.12.exe D:\dictionaries\n.py
Berlin

Process finished with exit code 0
```

مثـال(2):

**Input**

```
data = {
    1: 'Admin',
    2: 'Editor',
    3: 'Reader'
}
# هنا قمنا بطباعة لعنصر الذي يمكاك الرقم 1
print(data[1])
```

**Output**

```
C:\Users\HP\AppData\Local\Microsoft\WindowsApps\python3.12.exe D:\dictionaries\n.py
Admin

Process finished with exit code 0
```

ملاحظة :

إذا لم يتم العثور على المفتاح في القاموس يرجع القيمة **None** من **KeyError**

مثـــال(3):

Input

```
data = {
    1: 'Admin',
    2: 'Editor',
    3: 'Reader'
}
# هنا قمنا بطباعة لعنصر الذي يحمل الرقم 4
print(data[4])
```

Output

```
C:\Users\HP\AppData\Local\Microsoft\WindowsApps\python3.12.exe D:\dictionaries\n.py
Traceback (most recent call last):
  File "D:\dictionaries\n.py", line 8, in <module>
    print(data[4])
          ~~~~^^^
KeyError: 4

Process finished with exit code 1
```

## 3.إضافة وتعديل العناصر في القاموس

قواميس بايثون قابلة للتغيير وديناميكية. وبالتالي، يمكن تحديث عناصر القاموس، أو إضافة عناصر جديدة، أو حذف العناصر الموجودة.

**لتحديث عنصر قاموس، يتم تعيين قيمة جديدة لهذا العنصر من خلال الإشارة إليه بالمفتاح المرتبط.**

### أ-إضافة عنصر جديد

يمكن إضافة أزواج قيمة-مفتاح إلى قاموس دون استخدام توابع أو دوال باستخدام الصيغة التالية:

dict[Key] = value

مثـــال(4):

Input

```
Capitals={"France":"Paris","Germany":"Berlin","Spain":"Madred"}
print (Capitals)
# اضافة عنصر جديد الى القاموس
Capitals["Italy"] = "Roma"
print (Capitals)
```

Output

```
{'France': 'Paris', 'Germany': 'Berlin', 'Spain': 'Madred'}
{'France': 'Paris', 'Germany': 'Berlin', 'Spain': 'Madred', 'Italy': 'Roma'}

Process finished with exit code 0
```

### ب- تعديل قيمة موجودة

مثـال(5):

Input

```
cars = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(cars)
cars["brand"]="Gmc"
print(cars)
```

Output

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
{'brand': 'Gmc', 'model': 'Mustang', 'year': 1964}
```

### 4.حذف العناصر من القاموس

يتيح برنامج Python حرية حذف أي عنصر من القاموس بحسب الحاجة

أ- حذف عنصر باستخدام الدالة del.

مثـال(6):

Input

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print(Dict)
# حذف العنصر الثاني في القاموس
del Dict['Charlie']
print(Dict)
```

Output

```
{'Tim': 18, 'Charlie': 12, 'Tiffany': 22, 'Robert': 25}
{'Tim': 18, 'Tiffany': 22, 'Robert': 25}

Process finished with exit code 0
```

ب- حذف عنصر باستخدام الدالة Pop

مثـال(7):

Input

```
cars = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(cars)
cars.pop("model")
print(cars)
```

Output

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
{'brand': 'Ford', 'year': 1964}
```

ج- حذف العنصر الأخير بالقاموس باستخدام الدالة( )popitem.

مثـال(8):

Input

```
cars = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(cars)
cars. popitem()
print(cars)
```

Output

```
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
{'brand': 'Ford', 'model': 'Mustang'}
```

## التعامل مع القواميس Dictionaries


تتيح الدالة **del** حذف الـ القاموس بأكمله أو حذف عناصر محددة منه.

مثـال(9):

**Input**

```
cars = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
del cars
print(cars)
```

**Output**

```
NameError: name 'cars' is not defined. Did you mean: 'vars'?

Process finished with exit code 1
```

- ظهر خطأ دلالي لأنه طلب طباعة القاموس والقاموس تم حذفه بالكامل

د - مسح جميع عناصر القاموس باستخدام دالة clear().

مثـال(10):

**Input**

```
cars = {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
cars.clear()
print(cars)
```

**Output**

```
{}
```


عند استخدام الأمر **clear** يتم حذف عناصر القاموس لذلك يظهر ناتج الطباعة فارغ ولكن عند استخدام الامر **del** فيتم حذف القاموس بأكمله وعند محاولة طباعته سينتج رسالة خطأ .

## التكرار عبر عناصر القاموس باستخدام دالة item( )

ترجع قائمة تحتوي على المفتاح والقيمة للعناصر الموجودة في القاموس

مثـــال :

يكنك استخدام حلقة for للتكرار على المفاتيح والقيم في القاموس.

التكرار على المفاتيح والقيم

```
for Key, value in my_dict.items():
  print(f"Key: {Key}, Value: {value}")
```

Input

```
d1 = {'id': 1, 'name': 'Noor', 'size': 37, 'age':26}
for x ,y in d1.items():
  print(x , y)
```

output

```
name Noor
size 37
age 26
```

### إنشاء قاموس متعدد الأبعاد Nested Dictionary

القاموس المتداخل **Nested Dictionary** في بايثون هو في الأساس قاموس داخل قاموس. وهي طريقة فعّالة لتخزين وتنظيم هياكل البيانات المعقدة في بايثون .

مثـال:

Input

```
#Build the nested dictionaries

customers  = {'customer1': {'id': 1, 'name': 'Noor', 'size': 37},
              'customer2': {'id': 2, 'name': 'Mona', 'size': 36},
              'customer3': {'id': 3, 'name': 'Sara', 'size': 38},
            }
#OR

customer1 = {'id': 1, 'name': 'Noor', 'size': 37}
customer2 = {'id': 2, 'name': 'Mona', 'size': 36}
customer3 = {'id': 3, 'name': 'Sara', 'size': 38}

customers=  {'customer1': customer1,
             'customer2': customer2,
             'customer3':customer3
             }
#To access items in nested dictionaries
print(customers['customer1']['name'])
```

## مثال إثرائي 13 : إنشاء قاموس والوصول لقيمة العنصر بمعلومية المفتاح ص 110:

Input

```
1  dic_student={"name" : "Ahmed Salem" ,  "arabic" :90 , "english" :85 , "science" :95, "computer" :98}
2  total_degrees=0
3  for key in dic_student:
4      # Print the element key in uppercase (upper() function)
5      print (key.upper() , ":", dic_student [key] )
6      # To get the sum of numeric values only type ( ) == int
7      if type (dic_student [key]) == int :
8          total_degrees  +=  dic_student [ key ]
9  print (f " Total  degrees  =  {total_degrees} " )
```

انشاء قاموس يحتوي على اسم الطالب ودرجات بعض المواد

طباعة القاموس كالتالي : المفتاح بأحرف كبيرة ثم" : " ثم القيمة المرتبطة بالمفتاح

قارن نوع القيمة في حال كانت القيمة عددية (درجة) يتم اضافتها الى المتغير total_degree

طباعة قيمة المتغيرمجموع الدرجات total_degree

## مثال إثرائي 14 : إنشاء قاموس متعدد الأبعاد Nested Dictionary ص110:

شرح المثال (14) اثرائي ص 111:

في البداية تم انشاء 3 قواميس كل قاموس يخزن بيانات ودرجات طالب واحد dict_student1 , dict_student2 , dict_student3 ثم تم انشاء قاموس متداخل باسم All_students وتجميع القواميس فيه كقيمة والمفتاح لها ( Key ) هو one,two,three

**Input**

```
dict_student1 = {"name" : "Ahmed" , "Arabic" :90, "English" :50, "Computer" :98}
dict_student2 = {"name" : "Ali" , "Arabic" :70, "English" :95 ,"Computer" :90}
dict_student3 = {"name" : "Fahed" , "Arabic" :40, "English" :92, "Computer" :98}

all_students = {
    "one" : dict_student1,
    "two" : dict_student2,
    "three" : dict_student3
}
# Printing students who failed in Arabic
count_arabic = 0
print ("Students who failed in Arabic")
print ("=====================================")
for x in all_students.values():
    if x["Arabic"] < 50:
        count_arabic += 1
        print ("name :", x["name"])
print ("Number of students failing in Arabic = ", count_arabic,"\n")

# Students with an excellent grade in English
print ("Students who get excellent in English")
print ("=====================================")
count_english = 0
for x in all_students.values():
    if x["English"] >= 90:
        count_english += 1
        print ("name :", x["name"])

print("Number of students who received an excellent grade in English =",count_english)
```

انشاء قاموس متداخل باسم all_students

انشاء متغير لحساب عدد الطلبة الراسبين في العربي وتخصيص القيمة له صفر

تكرار مقارنة درجة العربي لجميع الطلبة في القاموس عن طريق المفتاح
اذاكانت اقل من 50 , اضف واحد للمتغير واطبع اسم الطالب الراسب

اطبع عدد الطلبة الراسبين في العربي (قيمة المتغير)

انشاء متغير لحساب عدد الطلبة الممتازين في الانجليزي وتخصيص القيمة له صفر

تكرار مقارنة درجة الانجليزي لجميع الطلبة في القاموس عن طريق المفتاح
اذاكانت أكبر من اوتساوي 90 , اضف واحد للمتغير واطبع اسم الطالب المتميز

اطبع عدد الطلبة المتميزين في الانجليزي (قيمة المتغير)

حل ورقة عمل 1
ص112

اسم البرنامج: دولة الكويت

مشكلة البرنامج: التعامل مع عناصر القاموس (الحذف، الإضافة، والتعديل.

المطلوب : دراسة التعليمات البرمجية، ثم تنفيذ التالي:

1-على القاموس: **my_country()**

- تعديل قيمة العنصر **«population»** لتصبح .**«5000000»**
- إضافة مفتاح جديد **«Currency»** ، وإدخال قيمته عن طريق المستخدم.

2-على القاموس : **gcc_countries()**

- عدل ما يلزم ليتضمن القاموس دول مجلس التعاون الخليجي فقط.
- طباعة القاموس.

## ملاحظات للمعلم التأكيد على :

| | |
|---|---|
| إضافة عناصر جديدة إلى القاموس. | dictionary[Key] = value |
| تعديل القيم الموجودة في القاموس. | **التعديل** :يتم بنفس طريقة الإضافة، حيث يتم استبدال القيمة القديمة بالجديدة. |
| حذف عناصر من القاموس. | del dictionary[Key] |
| **إضافة عنصر جديد :** | |
| إضافة عملة الكويت إلى القاموس .my_country<br>المفتاح"currency"<br>القيمة"Kuwaiti Dinar" | # إضافة العملة<br>my_country["currency"] = "Kuwaiti Dinar" |
| **تعديل قيمة موجودة :** | |
| تحديث عدد السكان في القاموس my_country إلى قيمة جديدة<br>المفتاح "population" .<br>القيمة الجديد5000000 : | # تعديل عدد السكان<br>my_country["population"] = 5000000 |
| حذف دولة معينة من القاموس دول مجلس التعاون الخليجي gcc_countries .<br>المفتاح الذي سيتم حذفه"Yemen" | # حذف دولة من القاموس<br>del gcc_countries["Yemen"] |
| طباعة جميع العناصر في القاموس my_country. | # طباعة عناصر القاموس<br>print("\nInfo About Kuwait\n" + "-" * 40)<br>for Key, value in my_country.items:()<br>print(Key, ":", value) |

## حل ورقة عمل 1 ص112

### تحتاج تبديل Input

```python
# قاموس دول مجلس التعاون الخليجي
gcc_countries = {
    "Saudi Arabia": "Riyadh",
    "Kuwait": "Kuwait City",
    "United Arab Emirates": "Abu Dhabi",
    "Qatar": "Doha",
    "Oman": "Muscat",
    "Bahrain": "Manama",
    "Yemen": "Sana'a"
}
# قاموس معلومات عن الكويت
my_country = {
    "Country": "Kuwait",
    "Capital": "Kuwait",
    "population": 5000000
}

# إضافة عملة إلى القاموس
my_country["currency"] = "Kuwaiti Dinar"
# تعديل عدد السكان
my_country["population"] = 4500000
# حذف عنصر من قاموس دول مجلس التعاون الخليجي
del gcc_countries["Yemen"]
# طباعة معلومات عن الكويت
print("\nInfo About Kuwait\n" + "-" * 40)
for key, value in my_country.items():
    print(key, ":", value)
```

### Input

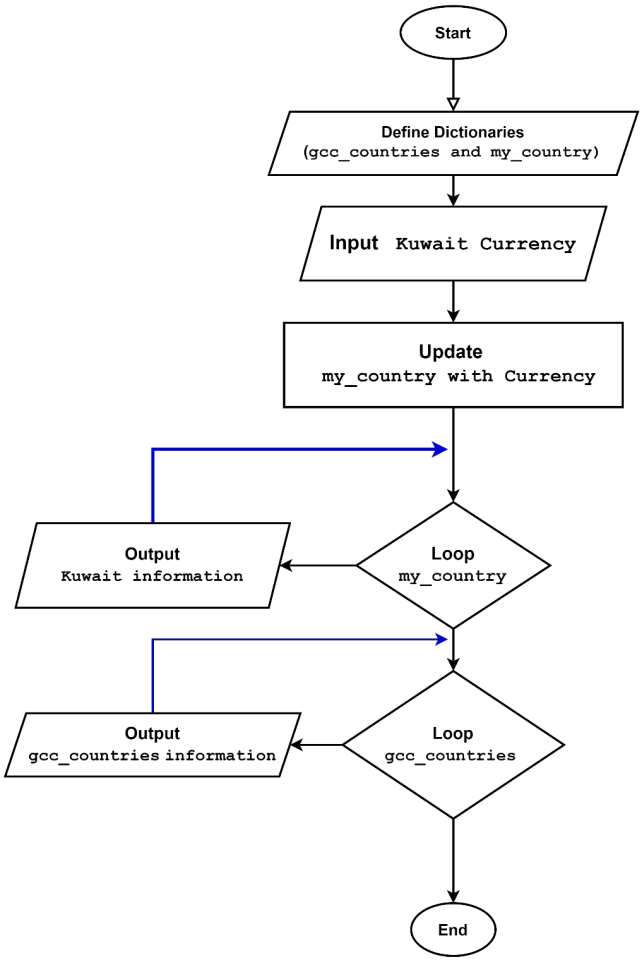
```python
# قاموس دول مجلس التعاون الخليجي
gcc_countries = {
    "Saudi Arabia": "Riyadh",
    "Kuwait": "Kuwait City",
    "United Arab Emirates": "Abu Dhabi",
    "Qatar": "Doha",
    "Oman": "Muscat",
    "Bahrain": "Manama"
}
# قاموس معلومات الكويت
my_country = {
    "Country": "Kuwait",
    "Capital": "Kuwait",
    "population": 5000000
}
# طلب إدخال العملة من المستخدم
currency = input("Enter Kuwait Currency: ")
my_country["currency"] = currency
# طباعة معلومات عن الكويت
print("\nInfo About Kuwait")
print("*" * 40)
for key, value in my_country.items():
    print(f"{key:<12} {value}")
# طباعة دول مجلس التعاون الخليجي
print("\nThe Gulf Cooperation Council Countries")
print("*" * 40)
for country, capital in gcc_countries.items():
    print(f"{country:<25} {capital}")
```

## حل ورقة عمل 1 ص112

### Output

```
C:\Users\HP\PycharmProjects\learv\.venv\Scripts\
 GUI\Time&date\n11.py"
Enter Kuwait Currency: Kuwait Dinar

Info About Kuwait
****************************************
Country       Kuwait
Capital       Kuwait
population    5000000
currency      Kuwait Dinar

The Gulf Cooperation Council Countries
****************************************
Saudi Arabia              Riyadh
Kuwait                    Kuwait City
United Arab Emirates      Abu Dhabi
Qatar                     Doha
Oman                      Muscat
Bahrain                   Manama

Process finished with exit code 0
```

### خريطة التدفق



إثرائي (2) ص 115

اسم البرنامج: درجات الطلاب

مشكلة البرنامج: التعامل مع عناصر القاموس (**Dictionaries**) ، والقوائم(**Lists**)

المطلوب: ادرس التعليمات البرمجية التالية، وناقش مع معلمك.

**.تطوير البرنامج بتنفيذ التالي :**

.إنشاء قاموس باسم( **student4** يحتوي بيانات متعلم)

.إضافة القاموس إلى القائمة **all_students** باستخدام الدالة المناسبة

## شرح ورقة العمل :

- تم في البداية انشاء قاموس خاص بكل طالب يحتوي على بياناته ( الاسم والنتيجة والدرجة الكلية )
- بعد ذلك تم انشاء قائمة باسم **all_students** تتكون من القواميس الخاصة بالطلبة

**( student1 ,student2,student3)**

- باستخدام حلقة for يتم طباعة مفاتيح القاموس**(name, result, total_degrees)** وقيمتها .لكل طالب
- تم استخدام تنسيق نصوص لجعل الطباعة مرتبة كالتالي :

| | | | |
|---|---|---|---|
| ({x:<20} ) | يعني أن المفتاح يأخذ عرض 20 خان | <: | يحدد اتجاه.محاذاة النص إلى اليسار. |
| x: | تمثل القيمة التي سيتم طباعتها (المفتاح في القاموس. | 20 | العرض الإجمالي للحقل (20 مسافة). |
| ( : ) | الفاصل بين اسم المتغير (x) وخيارات التنسيق. | | |

- بعد طباعة بيانات كل طالب، يتم طباعة خط فاصل **(40 * =)** للفصل بين طالب وآخر.

**Input**

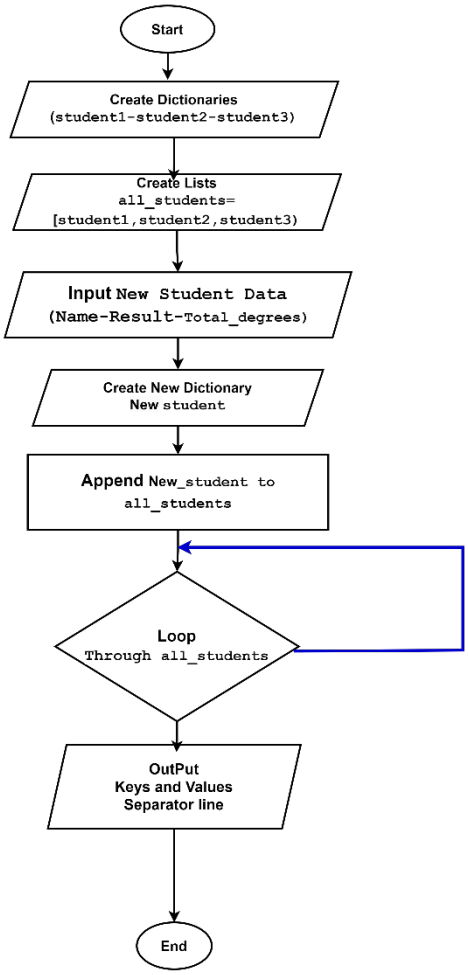
```
# إنشاء القواميس: لبيانات الطلاب
student1 = {"name": "ali", "result": "success", "total_degrees": 650}  # بيانات الطالب الأول
student2 = {"name": "khalid", "result": "success", "total_degrees": 700}  # بيانات الطالب الثاني
student3 = {"name": "ahmed", "result": "failed", "total_degrees": 250}  # بيانات الطالب الثالث
# إنشاء قائمة تحتوي على القواميس
all_students = [student1, student2, student3]  # قائمة تجمع بيانات الطلاب
# إضافة بيانات طالب جديد
# طلب إدخال البيانات من المستخدم
name = input("Enter new name: ")  # إدخال اسم الطالب الجديد
# إدخال نتيجة الطالب (نجاح أو رسوب)
result = input("Enter student result(success/failed): ")
# إدخال مجموع الدرجات (يجب أن يكون رقماً)
total_degrees = float(input("Enter total degrees: "))
# إنشاء قاموس جديد للطالب وإضافته إلى القائمة
new_student = {"name": name, "result": result, "total_degrees": total_degrees}
# إضافة قاموس الطالب الجديد إلى القائمة
all_students.append(new_student)
# طباعة بيانات جميع الطلاب
for i in all_students:  # المرور عبر كل طالب (قاموس) في القائمة
    # المرور عبر مفاتيح وقيم القاموس
    for x, y in i.items():
        # طباعة المفاتيح والقيم بتنسيق عرض 20 خانة
        print(f"{x:<20}{y}")
    # طباعة خط فاصل بين بيانات كل طالب
    print("=" * 40)
```

تابع : إثرائى (2) ص 115

Output

```
C:\Users\HP\PycharmProjects\learv\.venv\Scripts\
 GUI\Time&date\ex1.py"
Enter new name: Fahad
Enter student result(success/failed): success
Enter total degrees: 500
name                ali
result              success
total_degrees       650
========================================
name                khalid
result              success
total_degrees       700
========================================
name                ahmed
result              failed
total_degrees       250
========================================
name                Fahad
result              success
total_degrees       500.0
========================================
```

خريطة التدفق



# توظيف القواميس في واجهة المستخدم الرسومية

تم التطرق سابقا الى بعض عناصر واجهة التطبيق مثل أداة العنوان Label وأداة الصورة Image وأداة الزر Button , وسيتم التعرف على أداتين جديدتين في واجهة التطبيق :

| | |
|---|---|
| صندوق النصوص TextBox | وهي أداة تستخدم في حال الرغبة في ادخال بيانات من المستخدم مشابهة لأمر input ويتم استدعاؤها بهذه الصيغة :<br>TextBox_name(Optional) = ICT_KW.new_TextBox(Properies) |
| أداة صندوق القوائم ListBox | يمكن أن list أداة رسومية مرئية تفاعلية تعرض مجموعة من البيانات من نوع قائمة تنفذ أوامر عند الضغط على أحد عناصرها ويتم استدعاؤها بهذه الصيغة :<br>ListBox_name(Optional) = ICT_KW.new_ListBox(Properies) |

## ملاحظة حول كيفية تشغيل البرنامج وأهمية توفر ملف مكتبة :ICT_KW

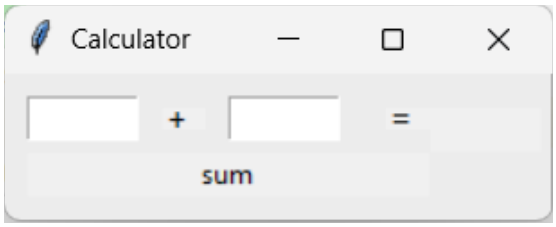
### تشغيل البرنامج:

1. تأكد من وجود مكتبة :ICT_KW
   - مكتبة ICT_KW هي المسؤولة عن توفير الأدوات الرسومية مثل (List Box, Button, Textbox)وتنفيذ واجهات المستخدم.
   - يجب أن تكون المكتبة مثبتة أو مرفقة مع البرنامج لكي يعمل.
2. التحقق من ملف المكتبة:
   - تأكد من وجود ملف المكتبة ICT_KW.py في نفس المجلد البرنامج ضروري لتشغيل الأدوات الرسومية.
   - إذا كانت المكتبة غير متوفرة، يمكن استبدالها بمكتبات أخرى مثل ..TKinter

شرح مثال (1)

## تطوير برنامج آلة حاسبة بسيطة لجمع عددين

**هدف البرنامج** :انشاء برنامج آلة حاسبة بواجهة رسومية تمكن المستخدم من إدخال عددين في مربع نص (Textbox) ، ثم عرض نتيجة الجمع عند الضغط على زر .




الخطوة الأولى: تصميم الواجهة الرسومية:

تتضمن:

- مربعي نص **(TextBox)** لإدخال الأرقام.
- رموز **(+)** و **(=** )عملية الجمع.
- زر **(Button)** لتنفيذ عملية الجمع.
- مربع نص **(Label)** لإظهار نتيجة الجمع.

كما تم تدريسه آلية التصميم في الموضوع السابق (تخطيط واجهة الرسومية - رسم مخططاً تخطيطياً للواجهة بناءً على المتطلبات)

الخطوة الثانية: تصميم الاستجابة للأحداث:

- الحدث :ضغط المستخدم على زر."**sum**"
- الاستجابة :استرجاع الأرقام من مربعي النصوص، جمعها، ثم عرض النتيجة في المربع الخاص بالنتيجة.

تعريف الدالة :**sum_numbers**

- تستدعي العناصر الثلاثة:**first_textbox - second_textbox- result_label** باستخدام الكلمة المفتاحية **global**.
- يتم استرجاع القيم النصية المدخلة باستخدام **get_text()** من مربع النصوص.
- تحول النصوص إلى أرقام باستخدام **float()**.
- تجمع الرقمين وتخزن الناتج في المتغير **sum_result**تعرض الناتج باستخدام الدالة **edit()** لتحديث النص في **result_label**.

## تابع -شرح مثال (1)

Input

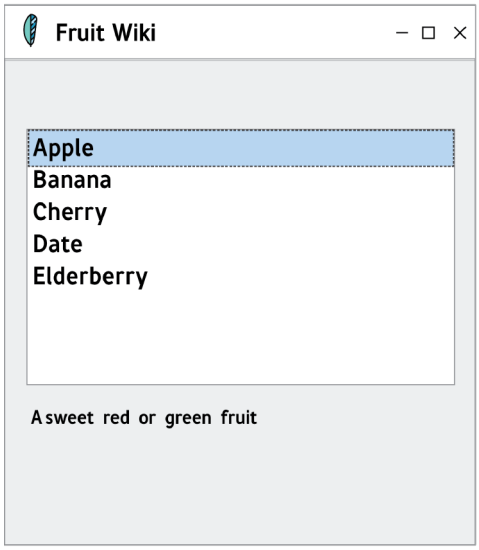
```
import ICT_KW
# تعريف الدالة لمعالجة الحدث
def sum_numbers():
    # استدعاء مربعات النص والعناصر النصية متغير معرف خارج دالة(متغير عام)
    global first_textbox, second_textbox, result_label

    # استرجاع النصوص المدخلة من مربعي النص
    a = float(first_textbox.get_text())  # استرجاع الرقم الأول وتحويله إلى رقم عشري
    b = float(second_textbox.get_text())  # استرجاع الرقم الثاني وتحويله إلى رقم عشري
    # جمع الرقمين
    sum_result = a + b
    # تحديث النص في result_label لإظهار النتيجة
    result_label.edit(text=str(sum_result))
# إنشاء نافذة واجهة المستخدم
ICT_KW.window(size=(250, 65), title='Calculator')
# إنشاء مربعات النص
first_textbox = ICT_KW.new_TextBox(location=(10, 10), size=(50, 20))
second_textbox = ICT_KW.new_TextBox(location=(100, 10), size=(50, 20))

# إنشاء العناصر النصية
ICT_KW.new_Label(location=(70, 15), size=(20, 10), text='+', align='left')
ICT_KW.new_Label(location=(170, 15), size=(20, 10), text='=', align='left')
result_label = ICT_KW.new_Label(location=(190, 15), size=(50, 20), text='', align='left')

# إنشاء زر الجمع وربط الدالة
sum_button = ICT_KW.new_Button(location=(10, 35), size=(180, 20), text='sum', todo=sum_numbers)
# تشغيل التطبيق
ICT_KW.run()
```

## شرح مثال (2)

### تطوير برنامج موسوعة الفواكه

**هدف البرنامج**: إنشاء برنامج يعرض معلومات عن الفواكه عند اختيار اسم فاكهة من القائمة (ListBox).

- يتم عرض الوصف التفصيلي للفاكهة المختارة في Label أسفل القائمة.



## تابع -شرح مثال (2)

### الخطوة الأولى: التخطيط للبرنامج

- إنشاء القاموس: **fruits_data**
  - المفتاح: (**Key**) اسم الفاكهة مثل (**Apple, Banana**)
  - القيمة: (**Value**) الوصف المرتبط بهذه الفاكهة.

### الخطوة الثانية: كتابة البرنامج

ربط الأحداث بالعناصر الرسومية : كيفية استخدام **on_select** لربط الأحداث بالدوال.

1. تعريف الدالة التي ستُنفذ عند الحدث حيث تسترجع العنصر المختار باستخدام **get_selected_item()**
2. تمرير الدالة إلى خاصية **on_select**

- إنشاء **ListBox**
- تمرير اسم الدالة إلى خاصية **on_select** عندما يختار المستخدم عنصرًا، يتم استدعاء الدالة تلقائيًا.

**Input**

```python
import ICT_KW
# تعريف بيانات الفواكه في قاموس
fruits_data = {
    'Apple': "A sweet red or green fruit.",
    'Banana': "A long yellow fruit that's soft inside.",
    'Cherry': "A small, round, red fruit with a pit.",
    'Date': "A sweet brown fruit from the date palm tree.",
    'Elderberry': "A small, dark purple fruit often used in syrups."
}
# تعريف الدالة لمعالجة الحدث
def fruits_information():
    global fruits_listbox, fruits_data, display_information_label
    selected_fruit = fruits_listbox.get_selected_item()
    if selected_fruit:
        display_information_label.edit(text=fruits_data[selected_fruit])
    else:
        display_information_label.edit(text="No fruit selected.")
# إنشاء نافذة البرنامج
ICT_KW.window(size=(300, 300), title='Fruit Wiki')

# إنشاء قائمة الفواكه
fruits_listbox = ICT_KW.new_ListBox(
    location=(10, 35),  # الموقع
    size=(230, 150),    # الحجم
    items=list(fruits_data.keys()),  # العناصر: أسماء الفواكه
    on_select=fruits_information      # استدعاء الدالة عند الاختيار
)
# إنشاء مربع النص لعرض الوصف
display_information_label = ICT_KW.new_Label(
    location=(10, 170),  # الموقع
    size=(250, 30),      # الحجم
    text="Click me for information!",  # النص الافتراضي
    align="left"         # المحاذاة
)
# تشغيل البرنامج
ICT_KW.run()
```

ورقة عمل (1) :

## تصميم برنامج«My Contact Book»

الهدف من البرنامج:

تصميم واجهة رسومية لإدارة جهات الاتصال باستخدام العناصر الرسومية التالية:

- مربعات نصية لإدخال اسم الشخص ورقم الهاتف.
- زر لإضافة جهة الاتصال إلى القائمة.
- قائمة (ListBox) لعرض جهات الاتصال المضافة.
- مربع نصي (Label) لتوضيح وظيفة البرنامج.

تعريف القاموس: contact_book لتخزين جهات الاتصال .

- المفتاح (Key): الاسم
- القيمة (Value): رقم الهاتف

حل ورقة العمل :

Input

```python
import ICT_KW

# إعداد البيانات لتخزين جهات الاتصال
contact_book = {}
# تعريف الدالة لإضافة الاسم ورقم الهاتف إلى القاموس
def add_to_contact_book():
    global name_textbox, phone_textbox, contact_book_listbox, contact_book
    # استرجاع النصوص المدخلة
    name = name_textbox.get_text()  # استرجاع الاسم المدخل
    phone = phone_textbox.get_text()  # استرجاع رقم الهاتف المدخل
    # التحقق من أن الحقول ليست فارغة
    if name and phone:
        # إضافة الاسم ورقم الهاتف إلى القاموس
        contact_book[name] = phone
        # تحديث قائمة العرض (ListBox) بالأسماء
        contact_book_listbox.edit(items=list(contact_book.keys()))
        # مسح مربعات النصوص بعد الإضافة
        name_textbox.edit(text="")
        phone_textbox.edit(text="")
    else:
        print("Both fields are required!")  # رسالة في حال كانت الحقول فارغة
# إعداد نافذة البرنامج
ICT_KW.window(size=(550, 300), title='My Contact Book', background_color=(251, 161, 175))

# إضافة صورة تعبيرية للاسم
ICT_KW.new_Image(location=(20, 30), size=(20, 40), image_file_path="images/female.png",
                 background_color=(251, 161, 175))
# إنشاء مربع النص لإدخال الاسم
name_label = ICT_KW.new_Label(location=(50, 30), size=(50, 20), text="Name:")
name_textbox = ICT_KW.new_TextBox(location=(100, 30), size=(150, 20))
# إضافة صورة تعبيرية للهاتف
ICT_KW.new_Image(location=(20, 80), size=(20, 40), image_file_path="images/phone.png",
                 background_color=(251, 161, 175))

# إنشاء مربع النص لإدخال رقم الهاتف
phone_label = ICT_KW.new_Label(location=(50, 80), size=(50, 20), text="Phone:")
phone_textbox = ICT_KW.new_TextBox(location=(100, 80), size=(150, 20))

# إنشاء الزر لإضافة جهة الاتصال
add_button = ICT_KW.new_Button(location=(100, 130), size=(80, 30), text="Add",
                               todo=add_to_contact_book)

# إنشاء القائمة لعرض جهات الاتصال
contact_book_listbox = ICT_KW.new_ListBox(location=(300, 50), size=(200, 130), items=[])

# تشغيل البرنامج
ICT_KW.run()
```

ورقة عمل (2) : **تطبيق تصميم معرض للصور. ص 128**

الهدف من البرنامج:

- استخدام المكتبة الرسوميةICT_KW.
- عرض وتحديث الوقت والتاريخ.
- عرض الصورة بعد تحديد مسارها.
- عرض قائمة تضم أسماء الصور التي تم عرضها ويتم معاينة الصورة من خلال الضغط على اسمها.

حل ورقة العمل:

## الخطوة الأولى: تصميم الواجهة الرسومية

1 اكتب التعليمة البرمجية لإنشاء النافذة الرئيسية:

```
import ICT_KW
ICT_KW.window(size=(1000, 850), title="Image Viewer",background_color=(255, 255, 255))
```

2 اكتب التعليمة البرمجية لإنشاء عنوانtimer_label:

```
timer_label = ICT_KW.new_Label( location=(0, 0), size=(1000, 25), text="Time & Date:", background_color=(240, 240, 240), fore_color=(0, 0, 0), align="center")
```

3 اكتب التعليمة البرمجية لإنشاء مؤقتTimer:

```
timer_label = ICT_KW.new_Label( location=(0, 0), size=(1000, 25), text="Time & Date:", background_color=(240, 240, 240), fore_color=(0, 0, 0), align="center")
```

4 اكتب التعليمة البرمجية لإنشاء صندوق الصورpicture_box:

```
picture_box = ICT_KW.new_Image(location=(0, 25), size=(800, 800), image_file_path="", background_color=(240, 240, 240))
```

5 اكتب التعليمة البرمجية لإنشاء صندوق النصوصimage_path_textbox:

```
image_path_textbox = ICT_KW.new_TextBox( location=(0, 810), size=(690, 25), text="")
```

6 اكتب التعليمة البرمجية لإنشاء زرview_button:

```
view_button = ICT_KW.new_Button( location=(700, 810), size=(95, 25), text="View", todo=view_picture)
```

7 اكتب التعليمة البرمجية لإنشاء صندوق القوائمimages_listbox:

```
images_listbox = ICT_KW.new_ListBox( location=(800, 25), size=(200, 800), items=[], on_select=load_image_from_list)
```

## تابع ورقة عمل (2)

### الخطوة الثانية: تصميم الاستجابة للأحداث

1 أعلن عن الدالة **update_date_and_time** ثم اكتب التعليمات البرمجية المناسبة لتغيير نص العنصر **timer_label** لعرض الوقت والتاريخ:

```
from datetime import datetime
def update_date_and_time():
current_time_date = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
timer_label.edit(text=f"Time & Date: {current_time_date}")
```

2 أعلن عن قاموس فارغ **image_dict** لتخزين اسم ومسار الصور:

```
image_dict = {}
```

3 أعلن عن الدالة **view_picture** واكتب التعليمات البرمجية اللازمة:

- عرض صورة في صندوق الصور **picture_box** مستعيناً بالمسار المدخل في صندوق النصوص **image_path_textbox**
- إضافة اسم ومسار الصورة للقاموس **image_dict** و تحديث العناصر في صندوق القوائم **image_listbox**:

```
def view_picture( )
```

4 أعلن عن الدالة **load_image_from_list** لعرض صورة في صندوق الصور **picture_box** عند الضغط على أحد عناصرها:

```
def load_image_from_list:()
  selected_image_name = images_listbox.get_selected_item()
  if selected_image_name:
    selected_image_path = image_dict.get(selected_image_name)
    if selected_image_path:
      picture_box.edit(image_file_path=selected_image_path)
    else:
      print("Image path not found in the dictionary!")
  else:
    print("No image selected!")
```

5 لتشغيل التطبيق، اكتب التعليمة البرمجية التالية في نهاية الملف:

```
ICT_KW.run()
```

### الخطوة الثالثة: اختبار البرنامج.

اختبار البرنامج والتأكد من خلوه من الأخطاء.

### الخطوة الرابعة: تصدير البرنامج.

06

# الوحدة الثانية: الذكاء الاصطناعي

# Artificial Intelligence

# الوحدة الثانية: الذكاء الاصطناعي Artificial Intelligence

## نتاجات التعلم :

| | |
|---|---|
| 1 | تعريف الذكاء الاصطناعي وفروعه المختلفة |
| 2 | التعرف على أحد الشخصيات الكويتية الفذة والتعرف على دورها ومساهمتها في مجال الذكاء الاصناعي , د. عبدالله المطوع |
| 3 | التعرف على مفهوم الذكاء الاصطناعي التوليدي وطريقة عمله |
| 4 | ادراك أهم التحديات المتعلقة بالذكاء الاصطناعي التوليدي |
| 5 | التعرف على أنواع أدوات الذكاء الاصطناعي |
| 6 | فهم طرق التعامل مع أدوات الذكاء الاصطناعي |
| 7 | التعرف على فوائد استخدام أدوات الذكاء الاصطناعي التوليدي |
| 8 | التعرف على أهم مجالات استخدامات للذكاء الاصطناعي التوليدي |
| 9 | التعرف على المبادئ الأخلاقية في التعامل مع أنظمة الذكاء الاصطناعي التوليدي |
| 10 | تمييز بعض التحديات التي تواجه استخدام أدوات الذكاء الاصطناعي |

## المصطلحات : الذكاء الاصطناعي Artificial Intelligence

الذكاء الاصطناعي **Artificial Intelligence**

تعلم الآلة **Machine Learning**

التعلم العميق **Deep Learning**

الذكاء الاصطناعي التوليدي

**Generative Artificial Intelligence**

## المصطلحات : الذكاء الاصطناعي Artificial Intelligence

الشبكات العصبية الاصطناعية

Artificial Neural NetworKs

نموذج اللغة الكبير

Large Language Model ( LLM)

المحول التوليدي سابق التدريب

pretrained Transform A Generative

فجوة القوة الحوسبية

Computing Power Gap

التطور السريع Rapid Development

انتهاك حقوق الملكية

Violation Intellectual Property

الحد من تنوع الآراء

Limiting Diversity of Opinions

التزييف العميق Deep faKe

توليد نص Text Generation

توليد الصور Image Generation

نماذج الانتشار Diffusion Mode

شبكات تنافسية توليدية

Generative Adversarial NetworKs( GANs )

توليد الفيديو Video Generate

تقنية الانتشار العكسي Inverse Diffusio

توليد الصوت Voice Generation

أدوات معالجة اللغة الطبيعية:( NLP)

أداة انشاء الصور الفنية Midjourney

أداة لإنشاء الصور الواقعية والفنية. DALL-E 2

أدوات الذكاء الاصطناعي الخاصة بجوجل

Google Cloud AI

منصة بناء وتدريب ونشر نماذج التعلم الآلي

Amazon Sage MaKer .

Microsoft Copilot

Google Gemini

أحد نماذج الذكاء الاصطناعي تقدما من جوجل.

## تابع-المصطلحات : الذكاء الاصطناعيArtificial Intelligence

الذكاء الاصطناعي من أجل الأرض

Artificial Intelligence for Earth

Internet of Things IoT

انترنت الأشياء

الذكاء الاصطناعي لمساعدة المكفوفين

Seeing Artificial Intelligence AI

06

# الوحدة الثالثة: المنتجات الرقمية

# المنتجات الرقمية

## نتاجات التعلم : المنتجات الرقمية

1. الإلمام بمفهوم التعرف على الوجوه:

- التعرف على الوجوه كأحد تطبيقات الذكاء الاصطناعي المستخدمة لتحليل الصور واكتشاف الوجوه فيها.
- التمييز بين تقنيات الكشف التقليدية مثل (Haar Cascade) والتقنيات الحديثة مثل Deep Neural NetworKs .
- إدراك أهمية هذه التقنية في التطبيقات العملية مثل الأمن والمراقبة والهواتف الذكية.

2. تثبيت واستخدام مكتبة:OpenCV :

- تثبيت مكتبة OpenCV والبدء باستخدامها لمعالجة الصور.
- التعرف على الدوال الأساسية في المكتبة، مثل قراءة الصور وعرضها.
- تطبيق تقنيات الكشف عن الملامح باستخدام نماذج مدعومة داخل المكتبة مثل Haar Cascade .

3. معالجة الصور:

- قراءة الصور من ملفات محددة وتحميلها في البرنامج.
- تحويل الصور إلى صيغة التدرج الرمادي لتقليل حجم البيانات وزيادة كفاءة المعالجة.
- تحليل تأثير تحويل الصور على سرعة ودقة الخوارزميات المستخدمة.

4. استخدام نموذج Haar Cascade:

- تعلم كيفية تحميل النموذج المدعوم من OpenCV .
- استخدام دالة detect Multiscale لاكتشاف مناطق الوجوه في الصورة.
- رسم مستطيلات حول الوجوه المكتشفة لعرض النتائج بشكل مرئي.

5. إعداد البيئة البرمجية:

- تثبيت الأدوات البرمجية الأساسية مثل OpenCVg PyCharm .
- تنظيم المشروع بهيكلية واضحة تشمل ملفات الكود، الصور، والنماذج.
- تصميم واجهة رسومية تسهل على المستخدم التفاعل مع المشروع.

## تابع -نتاجات التعلم : المنتجات الرقمية

6. تنمية المهارات:

- تطوير القدرة على التعلم الذاتي من خلال البحث عن حلول للمشكلات البرمجية.
- تطوير البرمجيات مع التعاون ضمن فريق.
  - تنمية مهارات العمل الجماعي عبر توزيع المهام على أعضاء الفريق.
- اكتساب مهارات توثيق المشاريع وعرض النتائج .
- تنمية الإبداع والابتكار للمتعلمين.
- اكتساب مهارات الالقاء وتعزيز الثقة في النفس.

## المصطلحات : المنتجات الرقمية

خوارزمية الرؤية الحاسوبية Haar Cascade

الشبكات العصبية العميقة Deep Neural NetworKs

تقنيات التعلم العميق Customize Libraries

تعلم الآلة Machine Learning

الشبكات التلافيفية

Convolutional Neural NetworKs

مكتبة مفتوحة المصدر لمعالجة الصور والرؤية الحاسوبية

Open-Source Computer Vision Library

صورة بتدرج الرمادي Grayscale Image

تعد لغة البرمجة بايثون أحد أكثر الأدوات شيوعًا ويتم استخدام بايثون في الذكاء الاصطناعي، حيث تتيح للمطورين بناء تطبيقات ذكاء اصطناعي متقدمة بسهولة وفاعلية، وهذه هي أشهر وأهم الأمثلة العملية والمشاريع الحقيقية التي تستخدم بايثون في مجال الذكاء الاصطناعي:

1. تطبيقات التعلم الآلي

- **تصنيف الصور**: يتم استخدام بايثون في الذكاء الاصطناعي بايثون مع مكتبات مثل TensorFlow و Keras لتطوير نماذج تعلم آلي لتصنيف الصور، مثل تحديد أنواع الحيوانات في الصور.
- **التعرف على الكلام**: يُستخدم بايثون لتطوير نماذج تعلم آلي للتعرف على الكلام وتحويله إلى نصوص مكتوبة.
- **التعرف على النصوص**: يُستخدم بايثون مع مكتبات مثل NLTK و SpaCy لتطوير نماذج للتعرف على النصوص وفهم معانيها.

2. تحليل البيانات والتنبؤات

- **تحليل البيانات المالية**: يتم استخدام بايثون في الذكاء الاصطناعي في تحليل البيانات المالية للتنبؤ باتجاهات الأسواق المالية واتخاذ القرارات الاستثمارية الذكية.
- **التنبؤ بالطقس**: يُستخدم بايثون في تطوير نماذج للتنبؤ بالظروف الجوية بناءً على البيانات الجوية التاريخية.

3. التصنيف والتحليل النصي

- **التصنيف الآلي للمقالات**: يتم استخدام بايثون في الذكاء الاصطناعي في تطوير نماذج تصنيف آلي لتصنيف المقالات والمدونات بناءً على مضمونها.
- **التحليل النصي لوسائل التواصل الاجتماعي**: يُستخدم بايثون لتحليل التغريدات والمنشورات على وسائل التواصل الاجتماعي لفهم مشاعر المستخدمين والاتجاهات العامة.

4. توليد النصوص الآلي

- **توليد التعليقات والمقالات**: يُستخدم بايثون مع تقنيات التعلم الآلي لتوليد تعليقات ومقالات آليًا بناءً على نماذج تعلم اللغة الطبيعية.

## ما هي مكتبة OpenCV؟

OpenCV (Open Source Computer Vision Library) هي مكتبة مفتوحة المصدر تحتوي على مئات الخوارزميات والأدوات لمعالجة الصور والرؤية الحاسوبية

استخدامات OpenCV

- **معالجة الصور**: تشمل عمليات التحسين، التصفية، التحويلات الهندسية، واستخراج الميزات.
- **الرؤية الحاسوبية**: تتضمن مهام مثل التعرف على الوجوه، تتبع الأجسام، والتعرف على الأنماط.

- **التعلم الآلي**: تستخدم في بناء وتصميم نماذج تعلم الآلة لتحليل الصور والفيديوهات.
- **التطبيقات الروبوتية**: تساعد في برمجة الروبوتات للتفاعل مع البيئة المحيطة من خلال الرؤية الحاسوبية.

تم التطرق بكتاب الطالب الى مشروع التعرف على الوجوه كتطبيق لمكتبة open CVبشكل تفصيلي , وسنقدم مقترحات لبعض البرامج التي تستخدم هذه المكتبة المهمة لاعطاء المعلم خيارات وأفكار إضافية في حال الرغبة بالاستزادة .

# مشروع : تطبيق فلتر المباشر

## فكرة المشروع:

إنشاء برنامج تفاعلي يستخدم مكتبة **OpenCV** لمعالجة الصور أو الفيديوهات وإضافة تأثيرات لتحويلها إلى مشاهد تشبه الرسوم المتحركة.

## خطوات المشروع:

1. اختيار المصدر صورة ثابتة أو مقطع فيديو.
2. تطبيق تأثير الكرتون:
   - تحويل المشاهد إلى رسومات كرتونية باستخدام تقنيات **OpenCV** مثل:
     - تحويل الصورة إلى تدرج رمادي.
     - تطبيق التنعيم (**Blur**) والتباين.(**Edge Detection**)
3. عرض النتيجة:
   - إذا كان المصدر فيديو، يتم العرض في الوقت الحقيقي.
   - إذا كان المصدر صورة، يتم عرض الصورة النهائية مع التأثير.

### التعليمات البرمجية

```python
import cv2

# تحميل الصورة
image_path = input("أدخل مسار الصورة: ")
image = cv2.imread(image_path)

# تحويل الصورة إلى تدرج الرمادي
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# تطبيق التنعيم
blurred = cv2.medianBlur(gray, ksize: 5)

# استخراج الحواف
edges = cv2.adaptiveThreshold(blurred, maxValue: 255, cv2.ADAPTIVE_THRESH_MEAN_C,
                              cv2.THRESH_BINARY, blockSize: 9, C: 9)

# تنعيم الألوان
color = cv2.bilateralFilter(image, d: 9, sigmaColor: 250, sigmaSpace: 250)

# دمج الحواف مع الألوان
cartoon = cv2.bitwise_and(color, color, mask=edges)

# عرض الصورة
cv2.imshow( winname: "Original Image", image)
cv2.imshow( winname: "Cartoon Image", cartoon)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

# تابع -مشروع : تطبيق فلتر المباشر

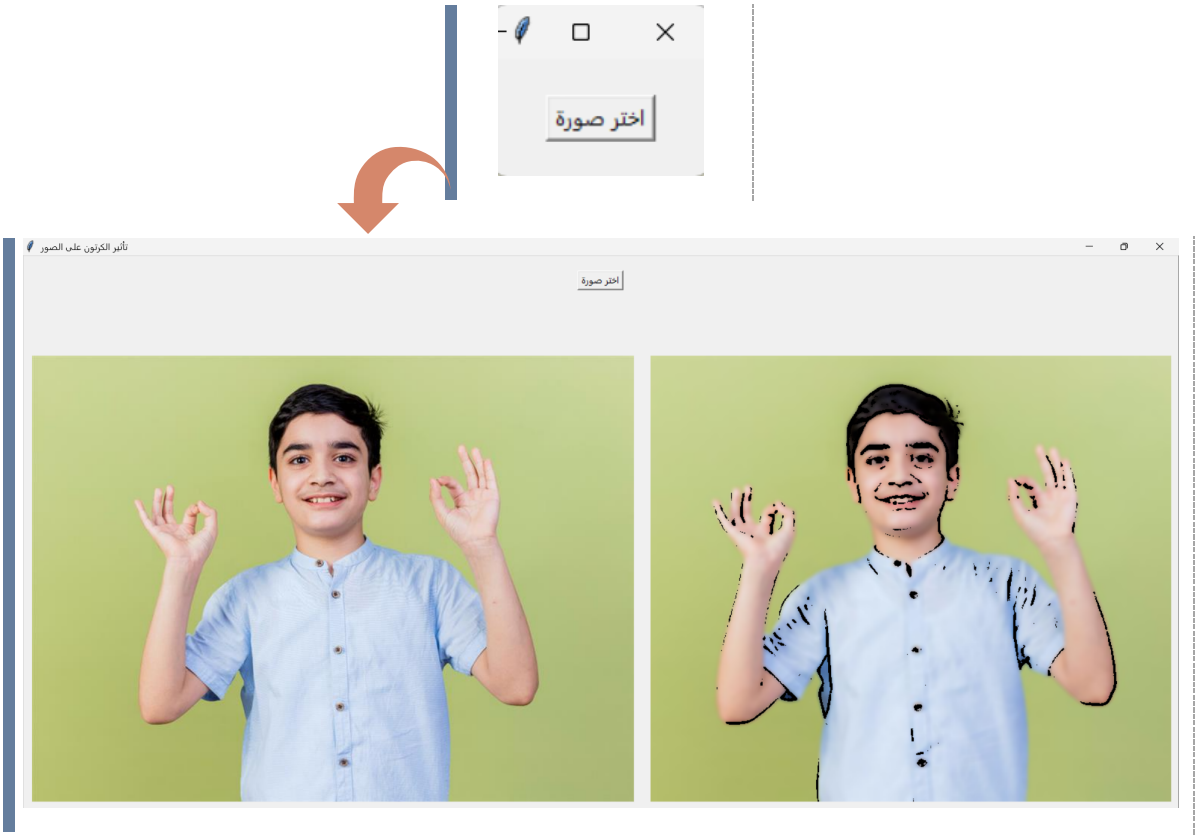
4. تطبيق المشروع مع إضافة واجهة تفاعلية:(GUI)

- استخدم مكتبة مثل tKinter لتبسيط اختيار الصورة أو الفيديو.
- واجهة لتحميل الصورة.
- زر لتطبيق التأثير وعرض النتيجة.

## توظيف الواجهة الرسومية مع مشروع تطبيق فلتر

```
import cv2
from tkinter import Tk, Button, Label, filedialog
from PIL import Image, ImageTk
# تطبيق تأثير الكرتون على الصور
def cartoonize_image(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    blurred = cv2.medianBlur(gray, ksize: 5)
    edges = cv2.adaptiveThreshold(blurred, maxValue: 255, cv2.ADAPTIVE_THRESH_MEAN_C,
                                  cv2.THRESH_BINARY, blockSize: 9, C: 9)
    color = cv2.bilateralFilter(image, d: 9, sigmaColor: 250, sigmaSpace: 250)
    cartoon = cv2.bitwise_and(color, color, mask=edges)
    return cartoon
# وظيفة اختيار الصوره
def select_image():
    global panel_original, panel_cartoon
    # فتح نافذه اختيار الصوره
    path = filedialog.askopenfilename(filetypes=[("Image files", "*.jpg *.png *.jpeg")])
    if len(path) > 0:
        # تحميل الصوره
        image = cv2.imread(path)
        if image is None:
            print("خطأ: تعذر تحميل الصورة. تحقق من المسار.")
            return
        # تطبيق تأثير الكرتون
        cartoon = cartoonize_image(image)
        # تحويل الصور لتنسيق PIL
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        cartoon = cv2.cvtColor(cartoon, cv2.COLOR_BGR2RGB)
        image_pil = Image.fromarray(image)
        cartoon_pil = Image.fromarray(cartoon)
        # تحويل الصور لتنسيق يمكن عرضه في tkinter
        image_tk = ImageTk.PhotoImage(image_pil)
        cartoon_tk = ImageTk.PhotoImage(cartoon_pil)
        # عرض الصور الأصلية والمعالجة
        if panel_original is None or panel_cartoon is None:
            panel_original = Label(image=image_tk)
            panel_original.image = image_tk
            panel_original.pack(side="left", padx=10, pady=10)
            panel_cartoon = Label(image=cartoon_tk)
            panel_cartoon.image = cartoon_tk
            panel_cartoon.pack(side="right", padx=10, pady=10)
        else:
            panel_original.configure(image=image_tk)
            panel_original.image = image_tk
            panel_cartoon.configure(image=cartoon_tk)
            panel_cartoon.image = cartoon_tk
# البرنامج الرئيسي
if __name__ == "__main__":
    # إنشاء نافذة التطبيق
    root = Tk()
    root.title("تأثير الكرتون على الصور")
    # إضافة زر لاختيار الصورة
    btn_select = Button(root, text="اختر صورة", command=select_image)
    btn_select.pack(side="top", pady=20)
    # متغيرات لعناصر واجهة العرض
    panel_original = None
    panel_cartoon = None
    # تشغيل التطبيق
    root.mainloop()
```

# تابع مشروع : تطبيق فلتر المباشر




ملاحظة للمعلم :

**تثبيت المكتبات التالية :**

1. تثبيت المكتبات باستخدام **pip**

**(python -m pip install --upgrade pip)**

2. تثبيت المكتبة **(OpenCV)**

**pip install opencv-python cv2**

3. تثبيت المكتبة (Pillow)

**pip install pillow**

# أفكار لمشاريع باستخدام مكتبة OpenCV

## معالجة الصور

- تطبيق تأثيرات كرتونية
- تحويل الصور إلى رسومات بالقلم الرصاص.
- تعديل الإضاءة والألوان

## التعرف على الكائنات

- اكتشاف الوجوه باستخدام خوارزميات مثل Haar Cascades.
- التعرف على السيارات أو الأشخاص في الصور والفيديوهات.
- بناء تطبيق يتعرف على الأشياء في الوقت الفعلي.

## تحليل الفيديو

- تتبع الأجسام المتحركة(Object TracKing) .
- تحليل حركات اللاعبين في مباراة رياضية.
- استخراج النصوص من الفيديو.

## مشاريع أخرى

- التعرف على الإيماءات (مثل اليد أو الأصابع).
- تطبيق فلاتر مثل تلك الموجودة في تطبيقات Snapchat.
- التعرف على النصوص في الصور باستخدام OCR

# مصادر تعلم :

| | | |
|---|---|---|
| 1 | الوحدة الأولى :<br>برمجة بايثونPython | الوحدة ١ - بايثون |
| | | ١ - أوراق العمل |
| 2 | الوحدة الثانية:<br>الذكاء الاصطناعيArtificial Intelligence | الوحدة ٢- بايثون |
| | | ٢- أوراق العمل |
| 3 | الوحدة الثالثة :<br>المنتجات الرقمية | الوحدة ٣ - بايثون |
| | | ٣- أوراق العمل |
| 4 | قناة التوجيه العام للحاسوب<br>مشغل فيديوهات الصف العاشر الفصل الدراسي الثاني | 10 - part 2 playlist |

امســـح الرمـــز

# الخاتمة

نختتم هذا الدليل بتفاؤل وامل في ان يكون له الأثر الإيجابي على رحلتكم التعليمية في تدريس منهج الصف العاشر وأن يكون معينا للمعلم أثناء تقديم المحتوى العلمي , فقد تم تصميم هذا الدليل لمساعدة المعلم في تقديم المادة العليمية من خلال فيديوات معدة وجاهزة لاستخدام المعلم يمكن أن يستعين بها لتقديم المحتوى العلمي وعرضه للمتعلمين كوسيلة مساعدة أثناء الشرح , وهي غير ملزمة له ,ويترك للمعلم حرية الاختيار في اعداد وتصميم ما يناسبه من فيديوات .

وندعو جميع المعلمين للاستفادة القصوى من الموارد والمعلومات المقدمة في هذا الدليل بهدف اثراء مهاراتهم وتعزيز معلوماتهم وخبراتهم , ونتطلع بشغف الى رؤية تفاعلاتهم ومساهماتهم في هذا المجال .

نحن ملتزمون بدعمكم في رحلتكم التعليمية التي ستستمر بفضل جهودكم المخلصة في تقديم الأفضل لمتعلميكم وحرصكم على استكشاف وتطوير مهاراتكم والاطلاع على كل ما هو جديد في مجال التكنلوجيا والبرمجة , حيث تحتم علينا ظروف العالم المتسارعة والتطور السريع في مجال مادة تقنية المعلومات الى ضرورة مواكبة كل ما هو جديد على الساحة .

نتطلع الى رؤية نتائجكم المذهلة في مجال البرمجة متمثلة في نتاجات المتعلمين من مشاريع جيدة تعكس مدى استفادة المتعلمين من معلومات أثناء الفصل الدراسي , وندعوكم الى المشاركة النشطة في المسابقات والتحديات التقنية لتطوير المهارات البرمجية لكل من المعلم والمتعلم .

نتمنى لكم كل التوفيق في رحلتكم التعليمية والمهنية ونأمل أن يكون هذا الدليل بمثابة مورد ومرجع ملهم لكم عند الحاجة.

10