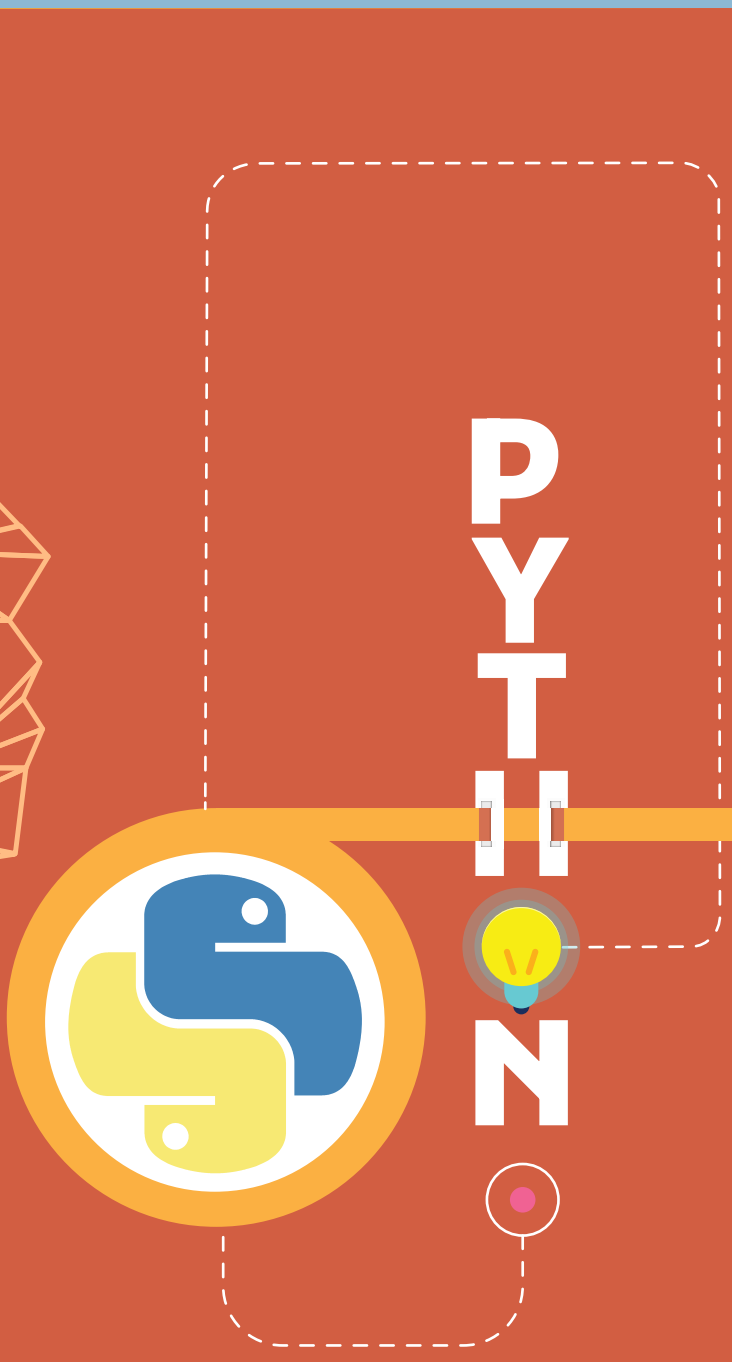
وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويـت

كويت جديدة
NEWKUWAIT

تقنية المعلومات

الصف العاشر - الجزء الثاني



الطبعة الأولى

وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويت

تقنية المعلومات

الصف العاشر - الجزء الثاني

إشراف

أ. منى سالم عوض سالم (رئيس اللجنة)

تأليف

أ. رأفـت صـابر عبداللاه أحمـــد د. أشـرف رضوان رضوان سليمان

د. يوسف منصور يوسـف الخليفي أ. منيرة خـالد محمد أبو شيبـــة

أ. إبراهيم عبدالله إبراهـيم المياس

تصميم

أ. إيمان عبدالعزيز أحمد الفارسي أ. سنية محـــمد على المـــــؤمن

الطبعة الأولى
١٤٤٦ هـ
٢٠٢٤/٢٠٢٥ م

الطبعة الأولى: 2024-2025م


الكتاب كاملاً


شاركنا بتقييم مناهجنا


القناة التربوية

المطبعة العصرية
AL ASSRIYA PRINTING PRESS
+965 22423543
+965 22420364
alassriya.com

أودع بمكتبة الوزارة تحت رقم (1) بتاريخ 2025/1/6م

حضرة صاحب السمو الشيخ مشعل الأحمد الجابر الصباح

أمير دولة الكويت

H.H. Sheikh Meshal AL-Ahmad Al-Jaber Al-Sabah
Amir Of The State Of Kuwait

سمو الشيخ صباح خالد الحمد الصباح

ولي عهد دولة الكويت

H. H. Sheikh Sabah Khaled Al-Hamad Al-Sabah
Crown Prince Of The State Of Kuwait

المحتوى

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

المقدمة

في ظل الثورة الرقمية التي أحدثت تغييرات جوهرية في مختلف المجالات، أصبح إتقان البرمجة ضرورة ملحة لتحقيق الطموحات، وقد برزت لغة Python بسبب سهولة تعلمها وشيوعها كأداة مثالية للدخول إلى عالم الذكاء الاصطناعي.

يهدف هذا الكتاب من تقنية المعلومات في لغة البرمجة Python إلى تزويدك بمعرفة شاملة حول توظيف Python في تطوير حلول فعّالة في مجال الذكاء الاصطناعي، بدءًا من المفاهيم الأساسية وصولًا إلى التطبيقات المتقدمة، وذلك بناءً على الأساسيات التي تم تناولها في الجزء الأول، نستكشف في هذا الكتاب الإمكانيات الواسعة للغة Python من خلال الوحدة الأولى التي تستعرض مهارات اكتشاف الأخطاء، والمكتبات، الواجهة الرسومية، الدوال، القواميس، وتوظيفها في الواجهة الرسومية.

تقدم الوحدة الثانية مقدمة شاملة في الذكاء الاصطناعي، بما في ذلك الذكاء الاصطناعي التوليدي، وأدواته المختلفة، وتطبيقاته العملية، إلى جانب استعراض المبادئ الأخلاقية التي تحكم استخدامه. كما يحتوي الكتاب على مشروعات عملية وأمثلة تطبيقية مصممة لتوضيح المفاهيم المعقدة بأسلوب بسيط ومنهجي، مما يتيح للطلاب بناء تطبيقات تعتمد على الذكاء الاصطناعي وتطوير برامج بواجهات رسومية (GUI) لتحقيق مشاريعهم الخاصة.

تتضمن الوحدة الثالثة مشروعات عملية تعتمد على استخدام تقنيات الذكاء الاصطناعي ومكتبات لغة Python لتطبيق ما تعلمه الطلاب في بيئة حقيقية. ومنها مشروع تطوير نظام ذكي مثل تطبيق التعرف على الوجوه، حيث يتم استخدام مكتبات متقدمة لتحليل الصور واستخراج البيانات منها. كما يمكن التوسع في المشروع لتضمين تطبيقات أخرى مثل تصنيف الصور، معالجة النصوص، أو بناء أنظمة توصية. يهدف هذا المشروع إلى تعزيز فهم المتعلمين لتوظيف أدوات الذكاء الاصطناعي في حل المشكلات الواقعية، مما يفتح أمامهم آفاقًا واسعة للإبداع والتعلم التطبيقي، ويعزز من قدرتهم على تطوير حلول مبتكرة باستخدام البرمجة والتقنيات الحديثة.

برمجة Python

الوحدة الأولى

تصحيح الأخطاء والاستثناءات
Debugging and Exceptions

1. مدخل إلى تصحيح الأخطاء والاستثناءات
2. تصحيح الأخطاء Debugging
3. الأخطاء اللغوية Syntax Errors
4. الأخطاء الدلالية Symanitc Errors
5. الأخطاء المنطقية Logic Errors
6. الاستثناءات Exceptions

تصحيح الأخطاء والاستثناءات
Debugging and Exceptions

نتاجـــات التعـلم

- التعرف على مفهوم الأخطاء (Errors) والاستثناءات (Exceptions)، والتمييز بين أنواع الأخطاء الشائعة مثل الأخطاء النحوية (اللغوية) (Syntax Errors) ، الأخطاء الدلالية Semantic Error، الأخطاء المنطقية Logic Errors.

- إدراك أهمية التعامل مع الاستثناءات لضمان استمرار عمل البرامج وتجنب توقفها المفاجئ عند مواجهة أخطاء غير متوقعة.

- استخدام التعليمة البرمجية Try لمحاولة تنفيذ الأوامر الأساسية، وتحديد السيناريوهات التي قد تؤدي إلى أخطاء.

- تطبيق التعليمة البرمجية Except لمعالجة الاستثناءات وتنفيذ تعليمات بديلة عند حدوث خطأ لضمان استمرار البرنامج بسلاسة.

- تحليل رسائل الخطأ واستنتاج أسبابها لتحسين مهارات تصحيح الأخطاء (Debugging).

- تطبيق آليات إدارة الأخطاء في مشروعات عملية لضمان استقرار البرامج وسهولة صيانتها.

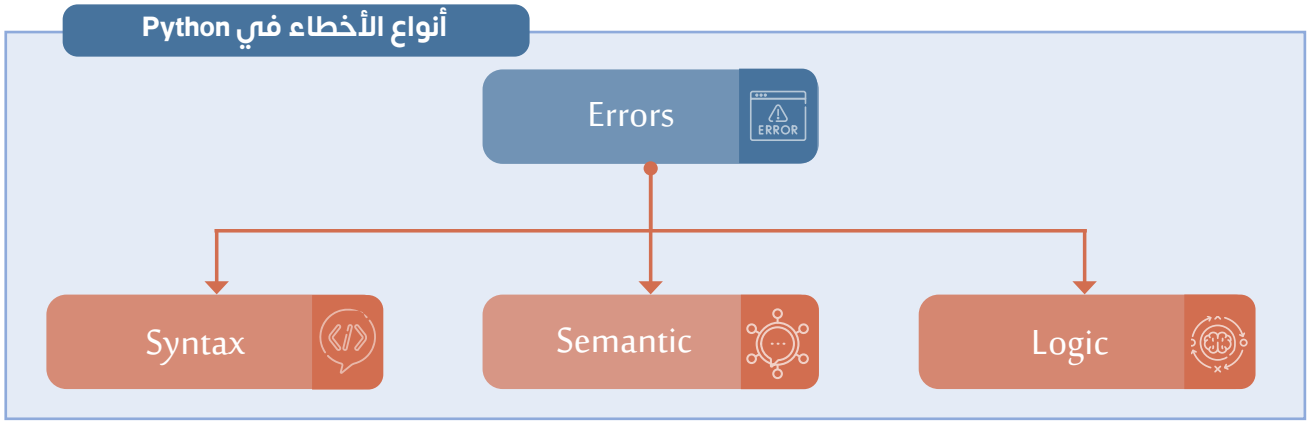
مدخل إلى تصحيح الأخطاء والاستثناءات

تعتبر الأخطاء والاستثناءات مفهوم أساسي في البرمجة، وتهدف إلى التعامل مع الأخطاء التي قد تحدث أثناء تشغيل البرنامج بطريقة منظمة تمنع توقفه، وتتيح آلية معالجتها للمبرمجين التحقق من الأخطاء والتعامل معها بمرونة. كما تساعد في تحسين استقرار البرنامج وتقديم تجربة مستخدم أفضل من خلال عرض رسائل تنبيهية أو اتخاذ إجراءات بديلة دون تعطيل البرنامج بالكامل.

تصحيح الأخطاء Debugging

قد يقع المبرمج في بعض الأخطاء عند تعامله مع النصوص البرمجية في لغة Python، ويمكن استخدام بعض التقنيات التي يتم تنفيذها لاكتشاف تلك الأخطاء وتصحيحها.

وهناك ثلاثة أنواع من الأخطاء وهي:

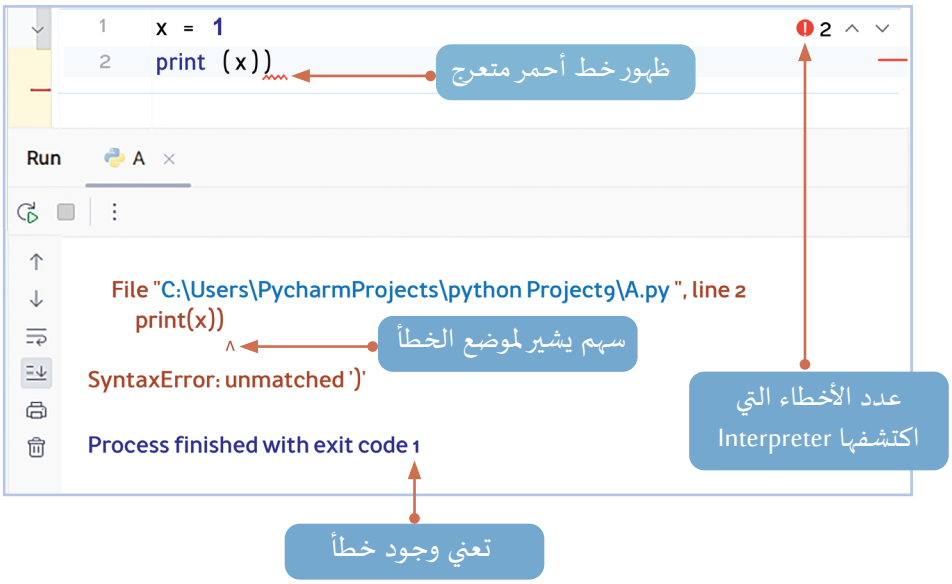


الأخطاء اللغوية Syntax Errors

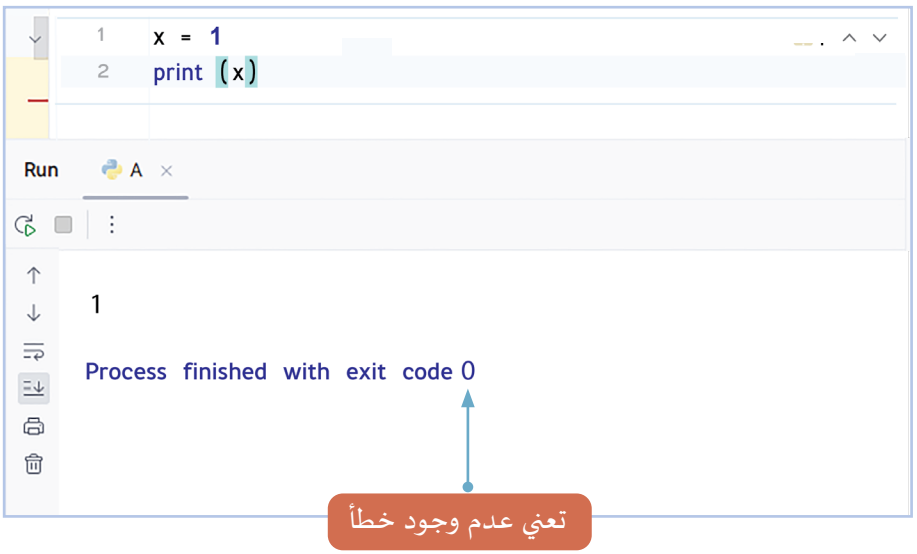
أخطاء تحدث عندما لا تتبع التعليمات البرمجية المكتوبة لقواعد اللغة البرمجية المستخدمة، وهذه الأخطاء تمنع البرنامج من العمل، ولا يستطيع المفسر Interpreter فهم تلك التعليمات بسبب وجود خطأ في تركيب الجملة البرمجية منها:

- استخدام مسافة إضافية أو ناقصة بين الكلمات أو الرموز.
- استخدام الأقواس أو علامات الاقتباس بشكل غير صحيح.

مثال 1: الأخطاء اللغوية Syntax Errors



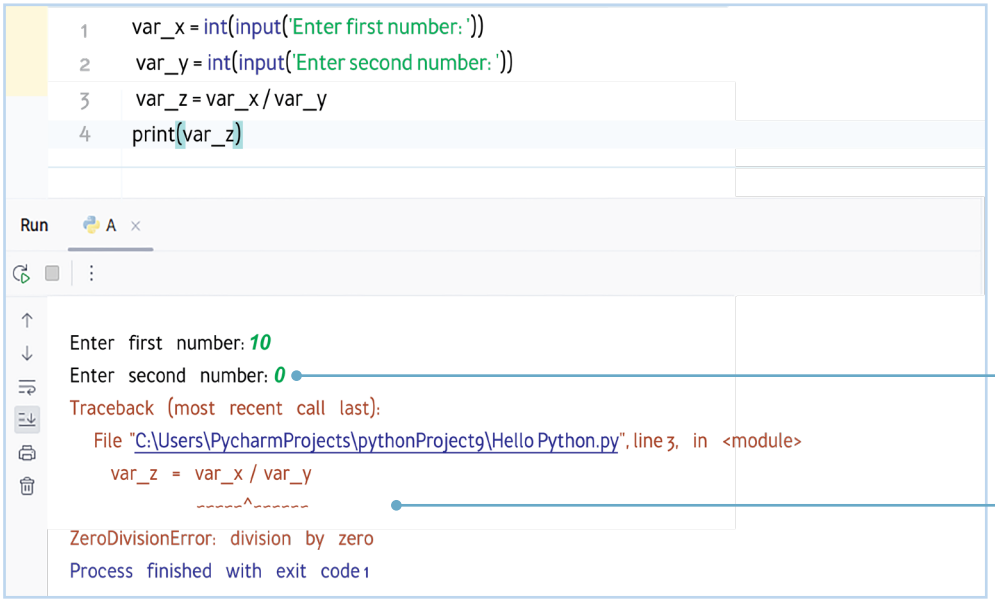
تصحيح الخطأ



الأخطاء الدلالية Semantic Errors

الأخطاء المكتشفة أثناء التنفيذ ولكن يجب أن تنشأ ظروف خاصة لتظهر تلك الأخطاء. على سبيل المثال:

- الوصول إلى عنصر غير موجود في قائمة محددة.
- الأخطاء الحسابية مثل تقسيم عدد صحيح على صفر.



هذا خطأ دلالي لأن المستخدم أدخل (صفر) كقيمة للمتغير الثاني، وحسابيًا لا يمكن القسمة على (صفر).

الأخطاء المنطقية Logic Errors

الأخطاء التي لا يتعرف عليها Python Interpreter، ولكنها قد تؤدي إلى نتائج غير متوقعة أو غير صحيحة (بمعنى أن النص البرمجي يعمل بدون مشكلات لكن نتيجة هذا النص البرمجي غير صحيحة). على سبيل المثال:

- استخدام شرط غير صحيح.
- استخدام العمليات الحسابية بشكل غير صحيح.

مثال 3

```
1  var_l = 5
2  var_w = 6
3  var_area = 2 *(var_l + var_w)
4  print ('Area = ' , var_area )
```

```
Run   A
Area = 22

Process finished with exit code 0
```

هذا خطأ منطقي لأن التعليمات البرمجية تحسب محيط المستطيل وليس المساحة.

تصحيح الأخطاء: عملية تحديد وإصلاح الأخطاء في التعليمات البرمجية.

مثال 4

المتغير نصي

logic error

```
1  x = input(' Enter first number: ')
2  if x == 0
3  print ( 'x is equal to 0. ')
4  else:
5     print('x is not equal to 0 .')
```

عدم وجود :

syntax error

عدم وجود مسافة بادئة

syntax error

لاحظ: في الجملة الشرطية يتم مقارنة قيمة x وهي قيمة نصية بالرقم 0 (مقارنة غير صحيحة)، وبالتالي يتم تنفيذ التعليمة البرمجية التابعة للأمر else.

الاستثناءات Exceptions

يمكن استخدام التعليمات البرمجية (try/except) لتحديد الجزء من التعليمات البرمجية المتوقع حدوث خطأ به، لاستمرار تنفيذ البرنامج دون توقف.

يوجد العديد من أنواع الاستثناءات ومنها (ValueError- NameError - IndexError - TypeError)، وسيتم التطرق إلى أحد هذه الأنواع:

ValueError

أن تطلب من المستخدم قيمة رقمية و بدلاً من ذلك يقوم المستخدم بإدخال قيمة نصية وهو ما ينتج عنه خطأ في البرنامج يسمى ValueError.

تنقسم التعليمات البرمجية try/except إلى:

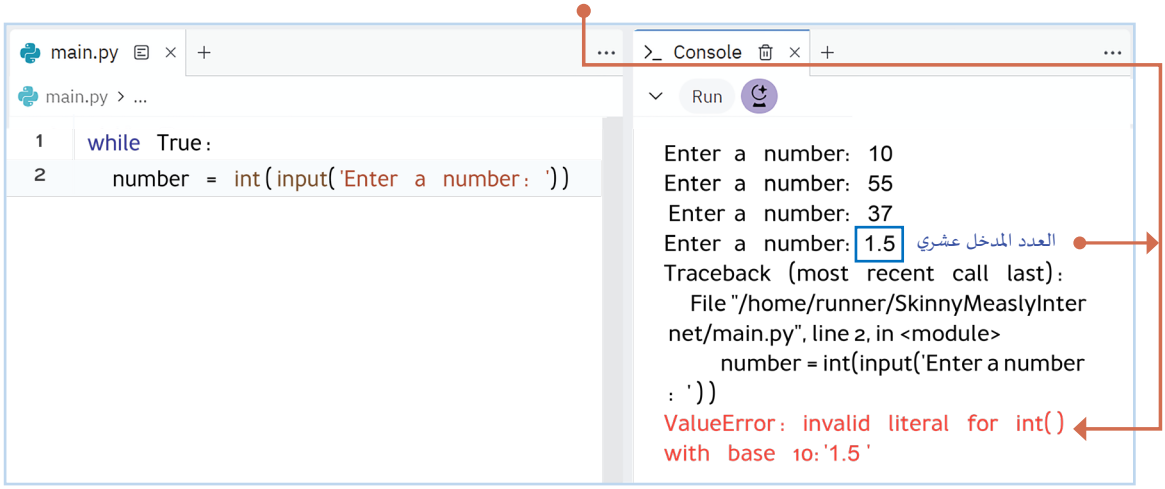
- التعليمة البرمجية try: تشمل التعليمات البرمجية التي قد تحدث فيها أخطاء أثناء التنفيذ.
- التعليمة البرمجية except: تشمل التعليمات البرمجية التي يتم تنفيذها إذا حدث خطأ في كتلة التعليمات البرمجية try.

الصيغة العامة للاستثناءات try/except Syntax

```
try:
    # try block
    # Put here code to test
except:
    # except block
    # Will executed if existed error in try block
```

مثال 5

1. يستمر البرنامج في استقبال بيانات عددية صحيحة integer فقط، ويتوقف البرنامج عند استقبال غير ذلك من البيانات، ويظهر رسالة الخطأ التالية: ValueError: invalid literal for int( )

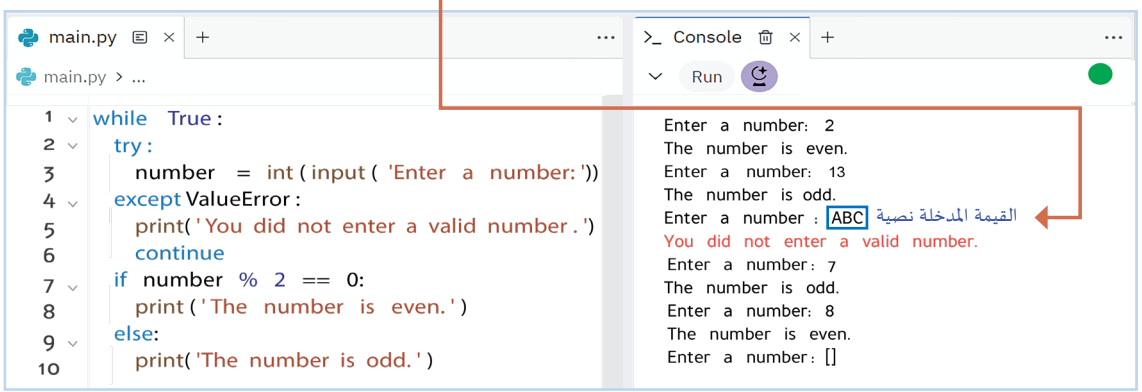


2. لاستمرار البرنامج دون توقف يتم استخدام try / except.

يستمر البرنامج في استقبال قيم متغيرة وعند إدخال بيانات غير عددية صحيحة، تُظهر رسالة للمستخدم You did not enter a valid number، ويستمر في استقبال البيانات، كما في المثال التالي:

مثال 6

برنامج يستقبل أعداد صحيحة، ويُظهر رسالة يحدد ما إذا كان العدد المدخل زوجيًا أم فرديًا، وعند إدخال بيانات من نوع آخر، يتم استثناء الخطأ وطباعة رسالة You did not enter a valid number، ويستمر البرنامج دون توقف.



ملاحظة: - تستخدم التعليمة continue لتجاوز بقية النص البرمجي في الحلقة الحالية والانتقال للتكرار التالي.

- تستخدم التعليمة break للخروج من الحلقة التكرارية فورًا.

مثال 7

البرنامج يهدف إلى حساب إجمالي الراتب للموظف بعد إضافة العلاوة.

إذا كانت المدخلات صحيحة، سيتم حساب إجمالي الراتب (الراتب الأساسي + العلاوة) وطباعته.

إذا كانت المدخلات غير صحيحة سيتم طباعة رسالة توضح نوع الخطأ.

- إدخال فهرس عنصر قائمة (اسم الموظف) غير صحيح تظهر رسالة خطأ من نوع IndexError.
  - إدخال فهرس عنصر قائمة = ٦ (مثلاً)
- إدخال قيمة العلاوة غير صحيحة (نصية) تظهر رسالة خطأ من نوع ValueError.
  - إدخال قيمة العلاوة = Ali (مثلاً)

```
salary = 50
names = ["Hoda", "Amal", "Sara", "Dalal", "Fatema"]
for name in names:

    try:
        # Get the index of the name from the user
        index = int(input("Enter the index of the name you want to access: "))
        # Try to access the name at the given index
        print(names[index])
        # Try to convert a string to an integer (which could lead to a TypeError)
        bouns = int(input("Enter a bouns: "))
        total_salary = salary + bouns
        print(f"{names[index]}, {total_salary} ")
    except IndexError:
        print("Error: Invalid index.")
    except ValueError:
        print("Error: Invalid value.")
```

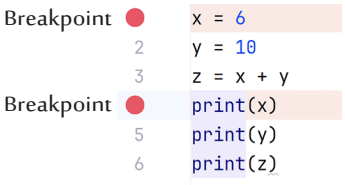
لمعلوماتك (تصحيح الأخطاء Debugging)

عملية تتبع سير التعليمات البرمجية خطوة بخطوة لتحديد الأخطاء البرمجية وإصلاحها باستخدام:

- أداة Debug من Main Menu .

- أدوات Debugging من Debug Menu.

- إضافة نقاط التوقف Breakpoints في أسطر التعليمات البرمجية.



أوراق العمل

ورقة عمل 1:

اسم البرنامج: برنامج إدخال الأوزان النسبية.

مشكلة البرنامج: يصحح الأخطاء المختلفة في التعليمات البرمجية.

- استدع الملف player_weight.py
- ادرس التعليمات البرمجية الخاصة بإدخال وزن اللاعب التالية:

- وزن الريشة (أقل من 63 كيلو جرام). - وزن المتوسط(63-90 كيلو جرام). - وزن ثقيل (يزيد عن 90 كيلو جرام)

المطلوب:

1. استكشف الأخطاء الموجودة وصححها ثم استكمل الجدول التالي:

رقم السطر	الخطأ	نوع الخطأ	التصحيح	ملاحظات
2				
2				
8				
10				
13				
21				

2. استخدم الاستثناءات Exceptions (try/except) لتطوير البرنامج.

```
# Create a list of player names
players = ["Ahmad", "Khaled", "Ali",  Salem , "Majed", "Fahed"
# Create an empty list
weights = []
# Player weights input by user
for player in players:
    weight = float(input(f"Enter player weight {player}: "))
    if weight = 63:
        print(f"Player weight {player}: {weight} Featherweight")
    elif 63 <= weight <  90:
        print(f"Player weight {player}: {weight} Middleweight")
    else:
    print(f"Player weight {player}: {weight} Heavyweight")
# Add player weights to the list
    weights.append(weight)
# Print players' names and weights
print(f"All player names: {players}\nAll player weights: {weights} ")
# Print average player weights - Total list elements / Length of list elements
print(f"Average weight: {sum(weights)/ len(weights)}")
# Print player name (Fahed) and his weight based on index
print(f"The winning player is: {players[10]},{weights[10]}")
```

ورقة عمل 2:

اسم البرنامج: حساب المتوسط الحسابي

مشكلة البرنامج: اكتشاف الأخطاء وتجاوزها (استكمال البرنامج العمل بصورة صحيحة).

المطلوب:

- افتح الملف average_Exceptions.py.
- عدل ما يلزم لتصحيح الأخطاء المحتمل استقبالها من المستخدم مستخدمًا الاستثناءات try/excep

```
# Create an empty list
numbers_list = []
sum = 0
# Infinite loop receiving data from the user
while True:

    input_number = input("Please enter a number ('Done' to exit): ")

    if input_number ='Done':
    break
    else:
        input_number = float(input_number)
        # Add number to list
        numbers_list.append(input_number)
    sum=sum+input_number


average = sum / len(numbers_list)

print("The numbers you entered:")
print(numbers_list)

print("Average:", average)

```

ورقة عمل 3:

اسم البرنامج: مؤشـر كتلة الجسم BMI (Body mass index).
مشكلة البرنامج: اكتشاف الأخطاء وتجاوزها (استكمال البرنامج العمل بصورة صحيحة).

المطلوب:

1. افتح الملف BMI.py، الذي يحسب مؤشـر كتلة الجسم BMI، بنـاءً على طـول الشخص (بالأمتار) ووزنه (بالكيلوجرامات)
2. تحقق من صحة الإدخال، مستخدمًا التعليمة البرمجية (try / except).

try:

- الخطأ من نوع ValueError عند إدخال بيانات نصية.

except ValueError:

- الخطأ من نوع ZeroDivisionError عند القسمة على صفر (الطول)

except ZeroDivisionError:

3. شغل البرنامج وتأكد من خلوه من الأخطاء.

```
while True:

        # Get user input for weight and height
        weight = float(input("Enter your weight in kilograms: "))
        height = float(input("Enter your height in meters: "))
        # Calculate BMI
        bmi = weight / (height ** 2)
        # Display the BMI
        print(f"Your BMI is: {bmi:.2f}")
        # Determine BMI category
        if bmi < 18.5:
            print("You are underweight.")
        elif 18.5 <= bmi <= 24.9:
            print("You have a normal weight.")
        elif 25 <= bmi <= 29.9:
            print("You are overweight.")
        else:
            print("You are obese.")

        print("Error: Please enter valid numeric values for weight and height.")

        print("Error: You can't divide by zero!")

```

الوحدة الأولى برمجة Python

المكتبات
Libraries

1. مدخل إلى المكتبات
2. أشهر المكتبات وأقسامها
3. التعامل مع المكتبات الخارجية
4. توثيق المكتبات

المكتبات Libraries

نتاجـــات التعـلم

الجزء الأول (أساسيات المكتبات)

- فهم المكتبات كأدوات لإعادة استخدام المقاطع البرمجية وتنظيمها، مما يسهل بناء مشروعات برمجية بكفاءة.
- تعلم كيفية استيراد المكتبات في لغة Python، واستخدام الدوال لتوفير الوقت والجهد.
- التعامل مع المكتبات القياسية المدمجة في لغة Python.
- التعامل مع المكتبات الخارجية تثبيتها، تحديثها، وإدارتها من خلال بيئة التطوير.
- تطوير مكتبات بسيطة تحوي دوال وفئات قابلة لإعادة الاستخدام في مشروعات مختلفة، مما يعزز مبدأ تقسيم البرامج إلى وحدات صغيرة(modularity).

المكتبات Libraries

مدخل إلى المكتبات

تتميز لغة البرمجة Python بتوفر العديد من المكتبات في مختلف المجالات؛ منها تحليل البيانات Data Analysis ، الذكاء الاصطناعي Artificial Intelligence، تعلم الآلة Machine Learning، التعلم العميق Deep Learning، الأمن السيبراني Cybersecurity، وتطوير الألعاب Gaming Development.

المكتبات في لغة Python (Python Libraries)

المكتبة عبارة عن ملف يتضمن مجموعة من التعليمات البرمجية والوظائف المُعرفة، كالدوال functions، والفئات classes والمتغيرات variables، والتي تُسهم في حل مشكلات معينة، أو تنفيذ عمليات محددة، والتي يمكن استدعاؤها واستخدامها في البرامج لتسهيل تنفيذ العديد من المهام.

أهمية استخدام المكتبات

إعادة استخدام المقاطع البرمجية: المكتبات توفر مقاطع برمجية جاهزة للاستخدام بدلاً من إعادة كتابتها بشكل متكرر.

توفير الوقت والجهد: من خلال استدعاء الدوال الموجودة في المكتبات بدلاً من بناء دوال من الصفر.

سهولة الصيانة والتحديث: المكتبات تتيح استخدام حلول مجربة، ومتقدمة مع القدرة على تحديثها بسهولة عندما يتم إصدار تحديثات جديدة.

أشهر مكتبات لغة Python

- مكتبة NumPy: تتعامل مع الدوال الخاصة بالعمليات الحسابية، الجبر الخطي، والمصفوفات.
- مكتبة Pandas: استكشاف وتحليل ومعالجة البيانات.
- مكتبة Matplotlib: تمثيل البيانات Data Visualization، وتصميم الأشكال البيانية Graph.
- مكتبة OpenCV: تعمل في مجال الرؤية الحاسوبية Computer Vision، وتحليل الصور ومقاطع الفيديو.
- مكتبة TensorFlow: مكتبة رائدة في مجالات تعلم الآلة والتعلم العميق.
- مكتبة PyTorch: مكتبة شهيرة في مجال التعلم العميق ومعالجة اللغات الطبيعية والرؤية الحاسوبية.

أقسام المكتبات

يمكن تقسيم المكتبات إلى:

1 المكتبات القياسية Standard Libraries

مكتبات مدمجة مع لغة Python، تغطي العديد من الوظائف الأساسية مثل التعامل مع الملفات، العمليات الحسابية، معالجة النصوص، والتعامل مع البيانات؛ ومنها مكتبة math لإجراء العمليات الحسابية، ومكتبة datetime للتعامل مع الوقت والتاريخ، ومكتبة random لتوليد القيم العشوائية.

2 المكتبات الخارجية Third-Party Libraries

مكتبات يتم تطويرها من قبل مجتمع مطوري لغة Python، الشركات والمؤسسات، الجامعات والمراكز البحثية، المنظمات غير الربحية، والمبرمجين، ويجب تثبيتها للتعامل معها على سبيل المثال مكتبة NumPy للعمليات الحسابية وتحليل البيانات، مكتبة Pandas لتحليل ومعالجة البيانات، ومكتبة TensorFlow لتعلم الآلة والذكاء الاصطناعي.

3 المكتبات الخاصة Customize Libraries

مكتبات يتم تطويرها من قبل المبرمج (إنشاء ملف Python) تتضمن مجموعة من الدوال functions، الفئات classes، المتغيرات variables، لتلبية احتياجات محددة بالمشروع.

التعامل مع المكتبات الخارجية Third-Party Libraries

تثبيت المكتبات الخارجية Install Third-Party Libraries

مثال: تثبيت مكتبة numpy

الطريقة الأولى: استخدام الأمر pip install يليه اسم المكتبة library name

- من خلال أداة Terminal في PyCharm.
- نكتب الأمر: pip install numpy

لمعلوماتك

الطريقة الثانية: استخدام الأداة Python Package .

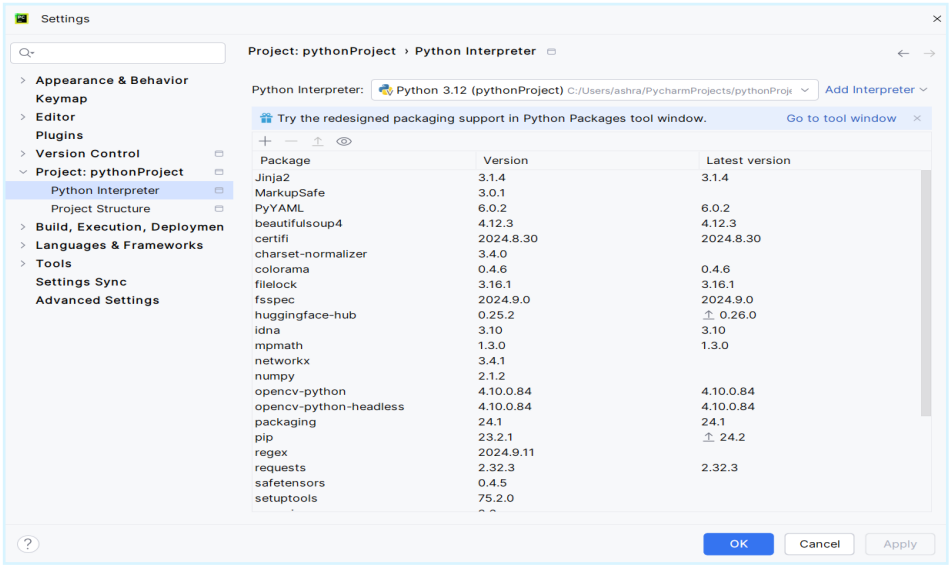
- البحث عن المكتبة المناسبة وتثبيتها من خلال الأمر install واختيار الإصدار المناسب للمكتبة.

لمعلوماتك

الطريقة الثالثة: استخدام نافذة الأمر Python Interpreter => Project: Project name => Settings => File

يتم الضغط على أداة install ، ثم البحث عن المكتبة المطلوبة.

كذلك يمكن استعراض المكتبات الخارجية التي تم تثبيتها على المشروع من نفس النافذة.



إلغاء تثبيت المكتبات الخارجية Uninstall Third-Party Libraries

يوجد العديد من الطرق لإلغاء تثبيت المكتبات الخارجية منها:

الطريقة الأولى: استخدام الأمر pip uninstall library name، من خلال أداة Terminal في PyCharm.

مثال: إلغاء تثبيت مكتبة (numpy)

```
pip uninstall numpy
```

لمعلوماتك

الطريقة الثانية: استخدام نافذة الأمر Python Interpreter => Project: Project name => Settings => File

يتم تحديد المكتبة ثم الضغط على أداة uninstall .

استيراد المكتبات Import Libraries

1. استيراد المكتبة كاملة

import library name

مثال 1: استيراد مكتبة datetime المخصصة للتعامل مع التاريخ والوقت (إظهار تاريخ اليوم).

```
1  import datetime
2  print(datetime.date.today())
3  print(datetime.date.today().day)
4  print(datetime.date.today().month)
5  print(datetime.date.today().year)
```

output

```
2024-10-22
22
10
2024
```

مثال 2: استيراد مكتبة random.

```
1  import random
2  random_num = random.randint(1,10)
3  print(f"The random intger is = {random_num}")
```

يتم استيراد الدالة random.randint المخصصة لتوليد أحد الأعداد الصحيحة بشكل عشوائي من المجال التالي (1,2,3,4,5,6,7,8,9,10).

عند استدعاء المكتبة كاملة باستخدام الأمر import، يجب كتابة اسم المكتبة random في كل مرة نستخدم فيها أحد دوالها متبوعاً بنقطة ثم اسم الدالة المطلوبة مثل (randam.randint).

2. استيراد عنصر محدد (التعليمات البرمجية الدوال functions، والفئات classes، و المتغيرات variables)، من مكتبة محددة

From library name import part of library

مثال 1: استيراد الدالة asctime فقط، من مكتبة time، والتعامل معها.

```
1  from time import asctime
2  # asctime( )  لطباعة التاريخ والوقت الحالي باستخدام دالة #
3  print (f"The current date and time is = {asctime ( ) }")
```

```
The current date and is = Thu Sep 19 15: 34 : 34 : 2024
```

مثال 2: استيراد أكثر من دالة من نفس المكتبة random

```
1  from random import randint, randrange
2  random_num1 = randint(1,10) # توليد أعداد صحيحة بشكل عشوائي (1,2,3,4,5,6,7,8,9,10)
3  random_num2 = randrange(10) # توليد أعداد صحيحة بشكل عشوائي (0,1,2,3,4,5,6,7,8,9)
```

مثال 3: استيراد أكثر من دالة من مكتبات متعددة

```
1  from dateutil.relativedelta import relativedelta
2  from datetime import datetime
3  today_date = datetime.today() # تاريخ اليوم
4  birth_date = datetime(2000, 12, 17) # تاريخ الميلاد
5  # كائن delta من النوع relativedelta، والذي يمثل الفرق بين التاريخين
6  delta = relativedelta(today_date, birth_date) # حساب العمر (الفرق بين تاريخ الميلاد وتاريخ اليوم)
7  print(f"Age: {delta.years} years, {delta.months} months, {delta.days} days")
```

- استيراد الدالة relativedelta من مكتبة dateutil.relativedelta: حساب الفرق بين تاريخين بدقة.
- استيراد الدالة datetime.today من مكتبة datetime: الحصول على التاريخ الحالي.
- تأكد من تثبيت مكتبة dateutil (في Terminal) باستخدام الأمر pip install python-dateutil.

3. استيراد المكتبة والتعامل معها باسم مختصر Import library name as shortcut name

مثال: التعبير عن اسم المكتبة datetime بالاسم المختصر dt

```
1  import datetime as dt
2  print(dt.date.today())
3  print(dt.date.today().day)
4  print(dt.date.today().month)
5  print(dt.date.today().year)
```

4. استيراد جميع الكائنات (مثل الدوال، المتغيرات، الفئات) من مكتبة معينة

مثال 1: استدعاء جميع محتويات المكتبة datetime والتعامل معها.

```
1  from datetime import *
2  print(date.today())
3  print(date.today().day)
4  print(date.today().month)
5  print(date.today().year)
```

مثال 2: استدعاء جميع محتويات المكتبة random والتعامل معها.

Import_library.py

```
1  from random import *
2  random_num1 = randint(1,10)          # توليد أعداد صحيحة بشكل عشوائي (1,2,3,4,5,6,7,8,9,10)
3  random_num2 = randrange(10)          # توليد أعداد صحيحة بشكل عشوائي (0,1,2,3,4,5,6,7,8,9)
4  random_chr = choice("Python")        # اختيار حرف عشوائي من الكلمة بين القوسين Python
```

يمثل رمز * Asterisk إمكانية التعامل مع جميع محتويات المكتبة، دون الحاجة إلى كتابة اسم المكتبة قبلها.

لمعلوماتك

توثيق المكتبات

دليل تفصيلي يساعد المطورين على كيفية استخدام محتوى المكتبة بشكل صحيح.

وللحصول على ملف توثيقي للمكتبة يمكن استخدام:

- استخدام أداة Terminal في PyCharm كتابة الأمر pydoc library name
  مثال: pydoc random
- استخدام مفتاح Ctrl والضغط على الزر الأيسر للفأرة على اسم المكتبة بمنطقة كتابة الأوامر.

```
"""Random variable generators.

    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:
```

شاشة تُظهر محتوى ملف توثيق المكتبة Random

أوراق العمل

ورقة عمل (1):

اسم البرنامج: توليد كلمة مرور قوية Strong Password

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

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

إنشاء كلمة مرور قوية تشمل عدد محدد من الحروف والرموز والأرقام.

```
#Password Generator Project
# import radom library

# lists of Letters, Numbers and Symbols
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 = ['-', '@', '&', '*', '_']

print("Welcome to the PyPassword Generator!")
# Enter numbers of each (Letters, Numbers and Symbols)
gen_letters = int(input("How many letters would you like in your password?: "))
gen_symbols = int(input(f"How many symbols would you like?: "))
gen_numbers = int(input(f"How many numbers would you like?: "))

# Initial Value of Variable
password = ""

# Generate Complex Password
for char in range(1, gen_letters + 1):
  password += random.choice(letters)
for char in range(1, gen_symbols + 1):


for char in range(1, gen_numbers + 1):


print("Password is:", password)

```

output

```
Welcome to the PyPassword Generator!
How many letters would you like in your password?: 5
How many symbols would you like?: 5
How many numbers would you like?: 5
Password is: ewhLH@@_--63089
```

تطوير البرنامج:

طور البرنامج ليتم توليد كلمة مرور مركبة لا يقل عدد عناصرها عن ٨، ولا يزيد عن ١٢.

ورقة عمل (2):

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.

```
import random
import datetime
questions = ["str()", "int()", "input()", "print()", "sum()", "del()", "exit()"]
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"]
score = 0
print(f'Today Date: {datetime.datetime.today()}')
while len(questions) > 2:
    random_index = random.randrange(len(questions))
    user_question = questions[random_index]
    print(f"Question: What is the meaning of: {questions[random_index]} function")
    for i in range(len(answers)):
        print(f"{i+1}-{answers[i]}")
    user_answer = int(input("choice correct answer: "))
    if (user_answer-1) == random_index:
        score += 1
        print(f"Correct answer\n{'=' * 50}")
    else:
        print(f"Wrong answer\n{'='*50}")
    questions.pop(random_index)
    answers.pop(random_index)
print(f"your score is:{score}")

```

- ادرس التعليمات البرمجية ثم استكمل البيانات المطلوبة:

- اسم المكتبات المستخدمة:
  - .....
  - .....
- اسم الدوال المستخدمة ووظيفة كل منها:
  - .....
  - .....

ورقة عمل (3):

اسم البرنامج: احسب عمرك

مشكلة البرنامج: توظيف المكتبات للحصول على العُمر الخاص بتاريخ الميلاد (يوم/ شهر /سنة).

المطلوب:

- استدع الملف calc_age.py.
- استكمل التعليمات البرمجية اللازمة.
- اختبر البرنامج وتأكد من خلوه من الأخطاء.
- أضف التعليمة البرمجية التالية، ثم شغل البرنامج واكتب وظيفتها.

```
print(f"You were born on a {birth_date.strftime("%A")}.")
```

.....

- استبدل %A بالحروف التالية وسجل وظيفة كل منها.

%B ..... %C.....

```
import
# dateutil.relativedelta من مكتبة relativedelta استيراد دالة
from                    import relativedelta
# استقبال من المستخدم يوم شهر/سنة الميلاد
user_day = int(input("Enter your birthday, Day: "))  # 13
user_month = int(input("Enter your birthday, Month: "))  # 8 or 08
user_year = int(input("Enter your birthday, Year 4 Digits: "))  # 2008
# التحقق من صحة تاريخ الميلاد #
try:
# 2008-08-13 يتم تحويل البيانات المُدخلة إلى كائن يُمثل تاريخ الميلاد
    birth_date = datetime.date(user_year, user_month, user_day)
except:
    print("Error: Invalid date entered. Please check the day, month, and year.")
    exit() # دالة تُستخدم للخروج من البرنامج
# الحصول على تاريخ اليوم
today_date = datetime.date.
# (الفرق بين تاريخ الميلاد وتاريخ اليوم) حساب العُمر
delta = relativedelta(today_date, birth_date)
print(f"Your age is: {delta.years} years, {delta.months} months, and {delta.days} days.")
# (الفرق بين تاريخ الميلاد وتاريخ اليوم) حساب العُمر
print(f"You were born on a {birth_date.strftime("%A")}.")

```

- تأكد من تثبيت مكتبة dateutil باستخدام الأمر pip install python-dateutil.

برمجة Python

الوحدة الأولى

واجهة المستخدم الرسومية GUI
Graphic User Interface

1 مدخل إلى واجهة المستخدم الرسومية GUI

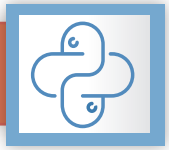
2 توظيف المكتبات في الواجهة الرسومية

واجهة المستخدم الرسومية GUI

نتاجـــات التعـلم

- الإلمام بمكونات واجهة المستخدم الرسومية من خلال عرض أبرز البرامج القائمة عليها لبيان أهميتها.
- تعلم أبرز المكتبات في لغة الـ Python المتخصصة في إنشاء واجهة رسومية واستخدام دوالها الجاهزة.
- تعلم خطوات التصميم الرسومي من التخطيط للبرنامج إلى كتابة وتصدير البرنامج.
- تعلم كيفية إضافة مكتبة (ICT_KW) إلى المشروع واستدعائها.
- التعامل مع بعض عناصر الواجهة الرسومية Widgets للمكتبة (ICT_KW) وضبط خصائصها.
- تصميم المشروعات التي تبرز أهمية توظيف واجهة المستخدم الرسومية.
- تحفيز المتعلم للبحث عن مكتبات قياسية تساعده على تصميم واجهات رسومية لتطوير مشروعه الخاص.

واجهة المستخدم الرسومية GUI



مدخل إلى واجهة المستخدم الرسومية GUI

إن معظم البرامج تعتمد على واجهات مرئية مشوقة تظهر نصوصاً ملونة وصوراً وأزرار وقوائم مختلفة تتطلب تفاعل المستخدم معها فتنقله من شاشة إلى أخرى وتغير مجرى الأمور بناءً على اختياراته فيصبح التعامل أسهل وأكثر تشويقاً من واجهة سطر الأوامر التي تعاملنا معها في الفصل الدراسي الأول. كما أن التطور التكنولوجي السريع شجع المبرمجون على تطوير الواجهات المرئية كونها الخطوة الأولى لتصميم مشاريع تعتمد على الذكاء الاصطناعي.

يستعرض هذا الجزء من الكتاب كيفية تصميم واجهة مرئية تسمى واجهة المستخدم الرسومية GUI (Graphical User Interface)، وبما أن تصميم الواجهة الرسومية يعتمد على المهارات الأساسية لبرمجة لغة Python بعضها سبق وتعلمناها في الكتاب الأول والباقي سنتعلمه تدريجياً بما يتناسب مع مكونات الواجهة الرسومية لنتمكن من تصميم وإنشاء وتصدير برنامج متكامل ومن المهارات التي سنتعلمها:

المكتبات والوحدات (Libraries- Modules).

الدوال (Functions).

البيانات غير الأولية (Non-Premitive Data).

سنبدأ في تعلم مفهوم واجهة المستخدم الرسومية ثم ننطلق لتعلم كل مهارة مما سبق.

مكونات واجهة المستخدم الرسومية GUI

واجهة المستخدم الرسومية (Graphical User Interface) GUI، هي واجهة تفاعلية تسمح للمستخدم التفاعل مع التطبيقات من خلال عناصر رسومية مرئية، مثل النوافذ، الأزرار ، والقوائم، التي تعمل على تبسيط تجربة المستخدم مما يجعل التطبيقات أكثر سهولة وملاءمة للاستخدام.

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

ينتظر البرنامج من المستخدم إدخال رقمه المدني في العنصر الفارغ:

واجهة المستخدم الرسومية GUI

عند الضغط على العنصر (اعرض تاريخ ميلادك) يتم معالجة القيمة المُدخلة وإظهار الناتج كما في الشكل التالي:

ومن هنا نرى أن واجهات المستخدم الرسومية تجعل البرامج أكثر سهولة في الاستخدام، وتساعد المستخدمين على أداء المهام بسرعة وفعالية، ومن أبرز أمثلة برامج واجهات المستخدم الرسومية:

1. أنظمة التشغيل Operating Systems: (macOS, Windows, Linux Distributions)
2. متصفحات الإنترنت Web Browsers: (Google Chrome, Mozilla Firefox, Microsoft Edge)
3. مشغلات الوسائط Media Players: (VLC Media Player,iTunes)
4. تطبيقات الهواتف الذكية Mobile Applications: (Whatsapp,Facebook, Instagram)
5. حزمة تطبيقات Office Suites: (Word, Power point, Excel, Teams)
6. الألعاب Games.

أهمية الواجهات الرسومية (GUI)

1. سهولة الاستخدام: تجعل الواجهة الرسومية التفاعل مع الأجهزة والبرامج أكثر سهولة وسلاسة، حيث يمكن للمستخدمين الضغط على العناصر المرئية بدلاً من كتابة الأوامر النصية.

2. تعزيز الإنتاجية: توفر الواجهات الرسومية عناصر مرئية تساعد المستخدمين على أداء المهام بسرعة وكفاءة، مما يعزز الإنتاجية.

3. سرعة التعلم: بفضل التصميم البسيط للواجهات الرسومية، يمكن للمستخدمين الجدد تعلم كيفية استخدام البرامج والأجهزة بسرعة دون الحاجة إلى تدريب مكثف.

4. جاذبية التصميم: تساهم الواجهات الرسومية في جعل التطبيقات أكثر جاذبية وجمالاً، مما يزيد من تفاعل المستخدمين ورضاهم.

5. التفاعل الفوري: توفر الواجهات الرسومية استجابات فورية من خلال التغييرات المرئية مثل تغيير الألوان أو الأشكال عند التفاعل مع العناصر، مما يساعد المستخدمين على فهم تأثير أفعالهم بشكل أفضل.

6. دعم المهام المتعددة: تسهل الواجهات الرسومية إدارة المهام المتعددة من خلال النوافذ المتعددة والأدوات التي تساعد في تنظيم العمل والتنقل بين المهام بسهولة.

7. إمكانية الوصول: يمكن تصميم واجهات المستخدم الرسومية لتكون في متناول الأشخاص ذوي الاحتياجات الخاصة، مما يجعل التكنولوجيا أكثر شمولاً.

واجهة المستخدم الرسومية GUI

أشهر المكتبات المستخدمة للتعامل مع (GUI):

- Tkinter.
- PyQt.
- PySide.
- Flask/Django.
- Kivy.
- wxPython.

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

م	اسم المكتبة	الاستخدام
1	.....	....
2	.....	....

توظيف المكتبات في الواجهة الرسومية

إنشاء الواجهة الرسومية:

1. تصميم الواجهة الرسومية (Front-end):

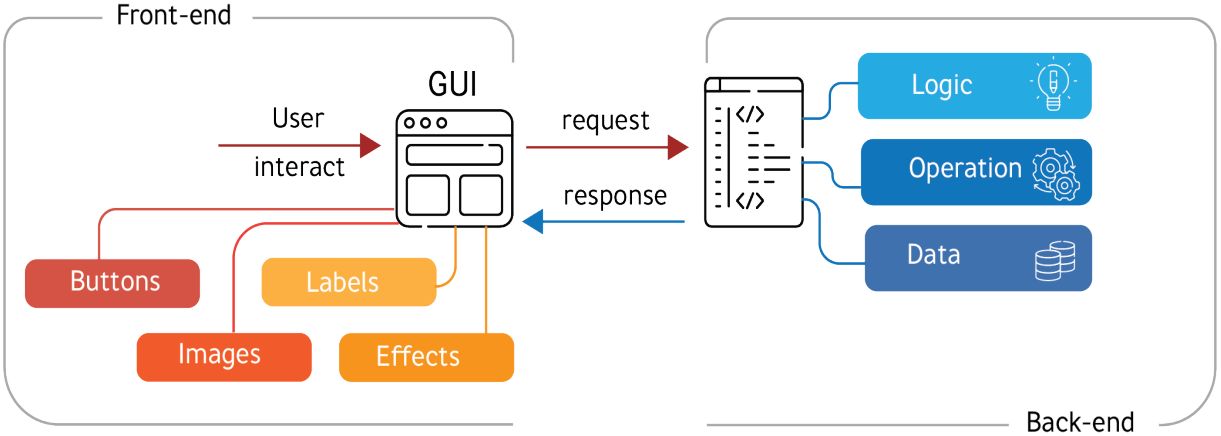
عملية بناء عناصر مرئية كالأزرار والعناوين والصور والقوائم وغيرها؛ التي يتفاعل معها المستخدم وتتمثل الخطوات في النقاط التالية:

- تَصوُر الشكل النهائي برسم واجهة البرنامج الرسومية التي سوف يتعامل معها المستخدمون.
- تحديد العناصر widgets المناسبة 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 خصائصها properties مستعيناً بأحد المكتبات المتخصصة في التصميم الرسومي.

2. تصميم الاستجابة للأحداث (Back-end):

هي الاستجابة Response لطلبات Requests الواجهة الرسومية التي تتمثل بتنفيذ مهام معين.

الشكل التالي يوضح العلاقة بين الواجهة الرسومية والاستجابة للأحداث:



3. اختبار البرنامج من حيث: أ- التأكد من استجابة الأزرار والقوائم ب- معالجة البيانات والأخطاء.
4. تصدير البرامج بإنشاء ملف تنفيذي.

سنعتمد في هذا الكتاب على المكتبة الخاصة (ICT_KW)، والتي أعدت خصيصاً لتصميم واجهة مستخدم رسومية بسيطة.

واجهة المستخدم الرسومية GUI

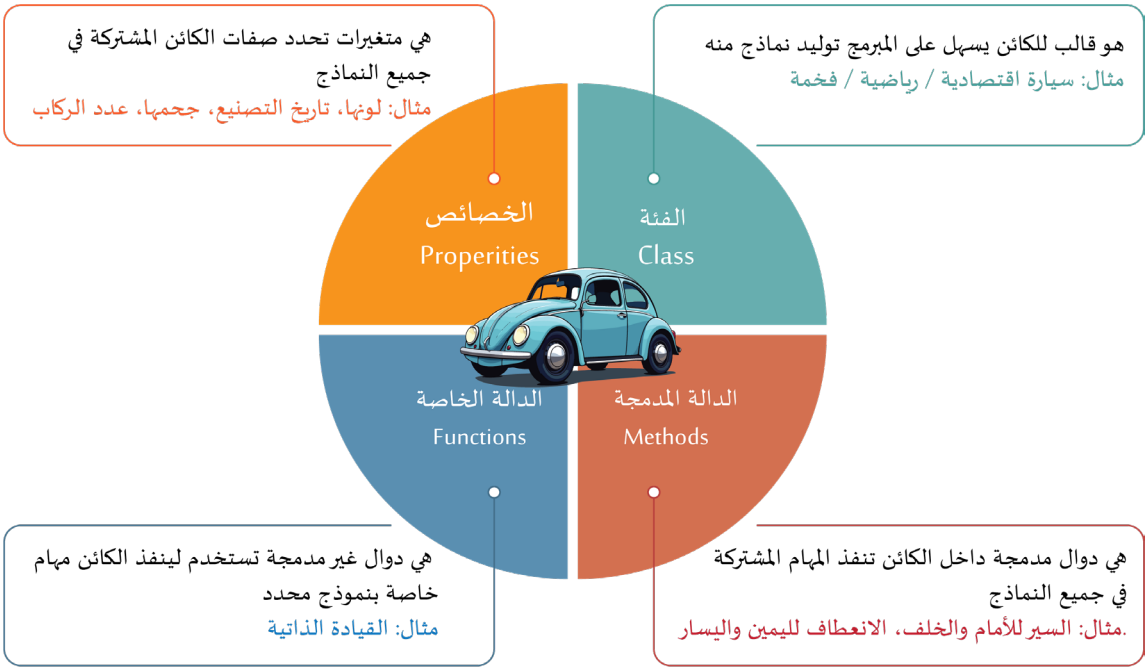
مكتبة (ICT_KW).

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، ولنتعرف على البرمجية الشيئية من خلال التالي:

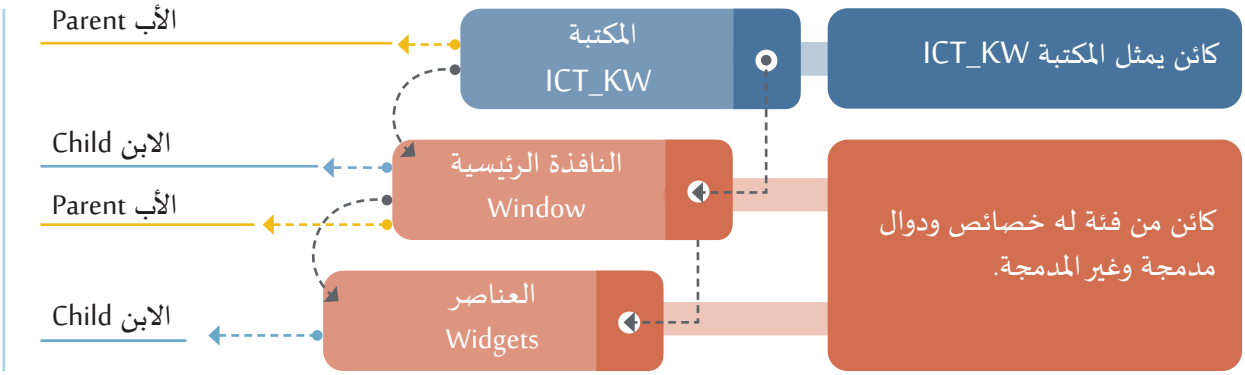
البرمجة الشيئية (Object Oriented Programming) OOP:

تعتبر لغة Python من اللغات التي تدعم البرمجة الشيئية، وهي نموذج برمجي يعتمد على الكائنات وتفاعلها مع بعضها البعض، يمثل الكائن بفئة class، يضم مجموعة من البيانات (المتغيرات)، والتي تعرف بالخصائص properties، وكذلك نوعين من الدوال: الدوال المدمجة methods والغير مدمجة functions.

مثال على ذلك: مصنع لصناعة السيارات، يصنع أنواع مختلفة من السيارات، لكل صنف خواص مشتركة ومهام للتنفيذ بعضها مشتركة والأخرى خاصة.



في لغة تتمثل العلاقات بين العناصر كعلاقة الوراثة inheritance بين الأب والأبناء حيث يرث الأبناء بعض الخصائص من الآباء، كما ترث العناصر خصائصها، ودوالها من النموذج البرمجي الأساسي؛ مما يعزز إعادة استخدام التعليمات البرمجية، ويُحسّن تنظيم البرنامج موضحًا بالشكل التالي:

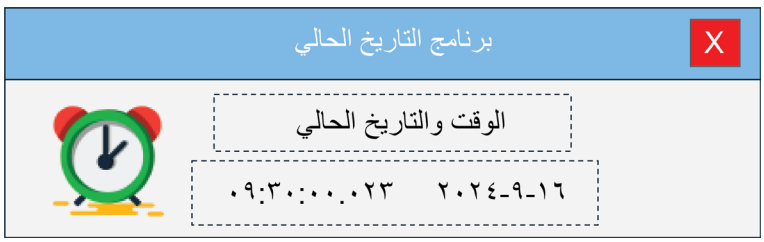


وبعد أن تعرفنا على مفهوم البرمجة الشيئية بشكل بسيط، سنبدأ في بناء مشروع رسومي «الوقت والتاريخ»، للتعرف على كيفية استخدام مكتبة ICT_KW، تصميم النافذة الرئيسية window، ثم إضافة مجموعة من العناصر إليها مثل العنوان Label، والصورة Image.

الخطوة الأولى: تصميم الواجهة الرسومية

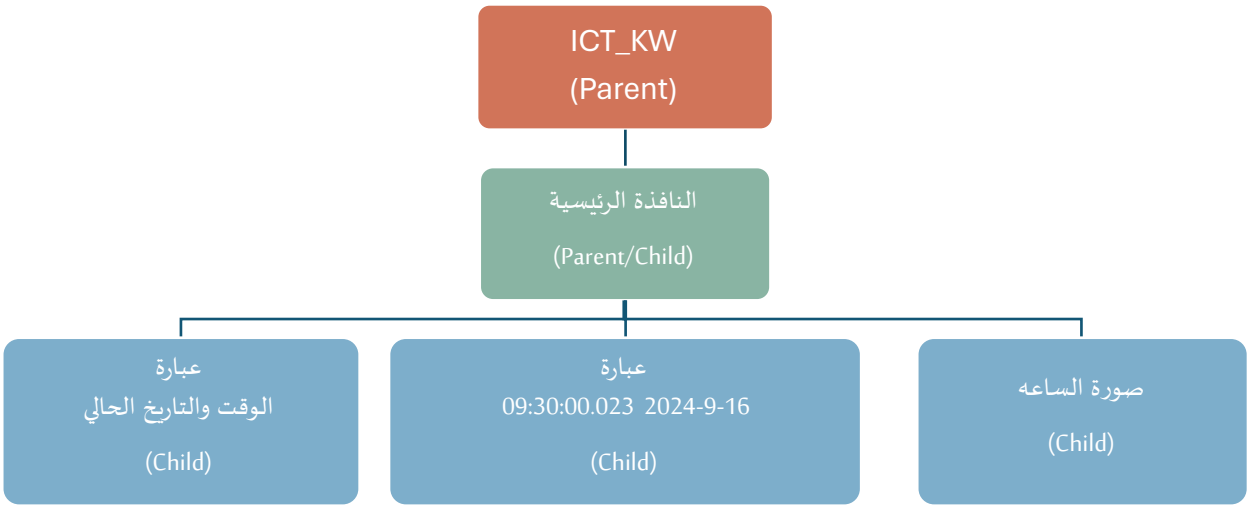
1: رسم واجهة المستخدم الرسومية.

تصور الشكل النهائي برسم مخطط للواجهة المطلوبة:



2 : تحديد العناصر widgets.

رسم مخطط هيكلي يوضح العناصر وخصائصها:



3 : كتابة التعليمات البرمجية.

أ- إنشاء مشروع جديد New Project في برنامج PyCharm باسم FirstGUI.

ب- إنشاء ملف Python باسم time_date_GUI.

ج- تثبيت المكتبة ICT_KW باستخدام الأمر pip install ICT_KW.

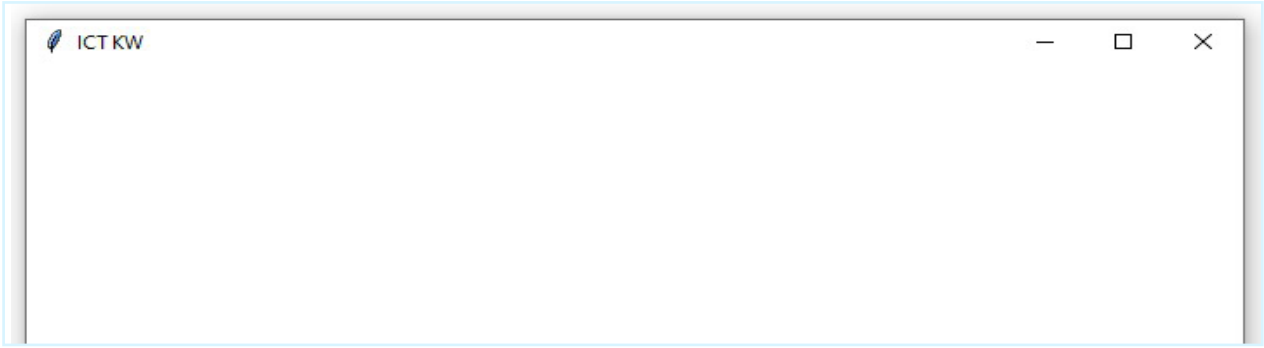
د- استدعاء مكتبة ICT_KW باستخدام الأمر import كالتالي:

```
import ICT_KW
```

هـ- تشغيل النافذة الرئيسية بكتابة التعليمية البرمجية التالية لعرض النافذة الرئيسية.

```
ICT_KW. run()
```

و- تشغيل البرنامج، لتظهر النافذة التالية:

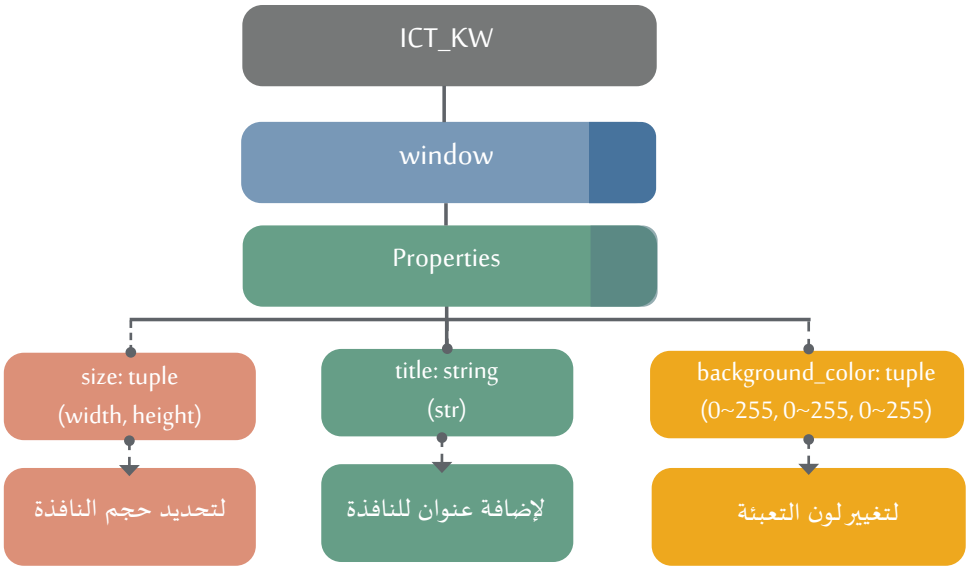


التحكم في خصائص النافذة الرئيسية Window

النافذة الرئيسية Window:

هي الإطار الأساسي الذي يحوي جميع عناصر الواجهة الرسومية.

الصيغة Syntax: ICT_KW. window(Properties)

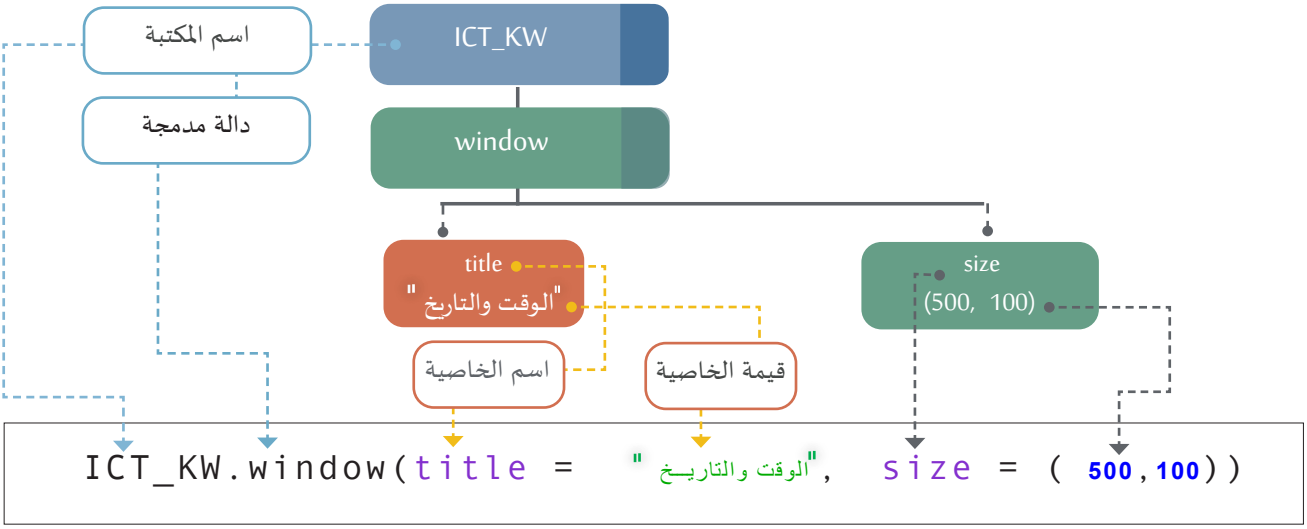


شكل يوضح الهيكل الرسومي للنافذة الرئيسية window وخصائصها

ومن خلال الهيكل الرسومي السابق يتم تمثيل كل عنصر بتعليمات برمجية لكي يعمل بشكل صحيح، وفق البرنامج المحدد له.

التعليمات البرمجية:

ويتم تحويل عناصر الهيكل الرسومي إلى تعليمات برمجية كما في الشكل التالي:



لاحظ :	الشكل السابق يوضح حجم النافذة الرئيسية بالبكسل Pixel

إضافة عناصر Widgets إلى النافذة الرئيسية

ويتم ذلك من خلال إحدى الطرق التالية:

1. إنشاء عنصر بطريقة مباشرة، والذي لا يمكن تعديل خصائصه.
2. إنشاء عنصر بطريقة غير مباشرة، وإسناده لمتغير.

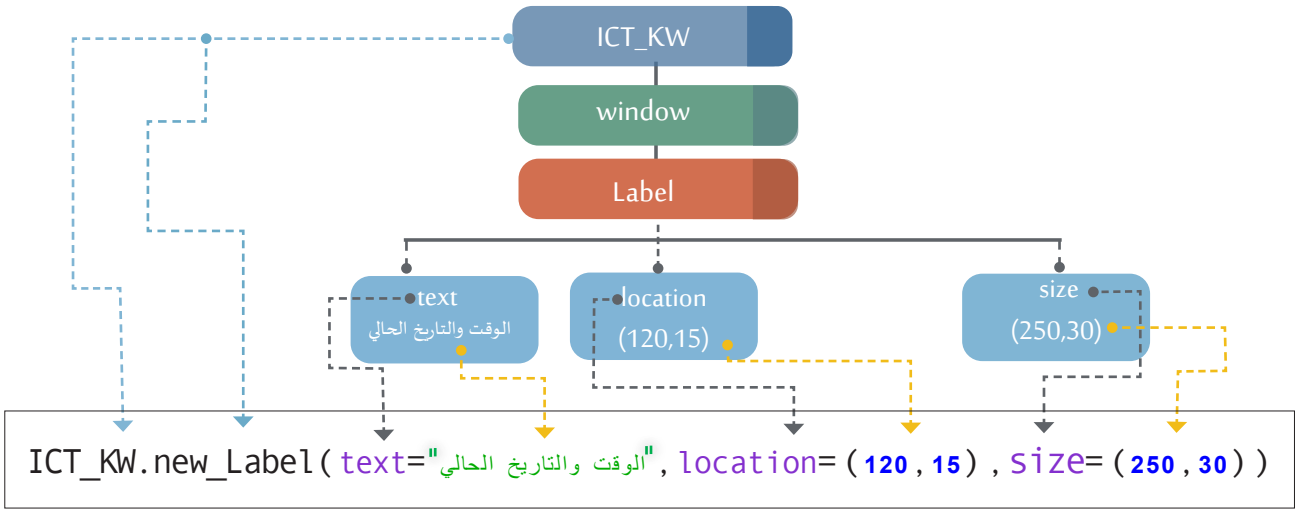
ا- العنوان Label

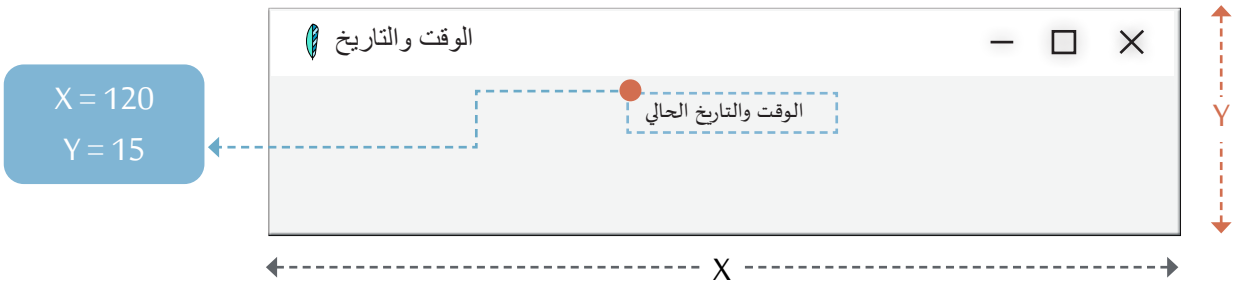
عنصر مرئي يعرض سلسلة نصية.

الصيغة Syntax: Label_name(Optional) = ICT_KW. new_Label (Properties)

Label	Properties	
	location: tuple (x, y)	موقع العنصر في النافذة الرئيسية
	size: tuple (width, height)	حجم العنصر في النافذة الرئيسية
	text: string (str)	النص المعروض للمستخدم
	align: ‹left›, ‹right›, ‹center›	محاذاة النص
	background_color: tuple (0~255,0~255,0~255)	لون التعبئة
	Fore_color: tuple (0~255,0~255,0~255)	لون خط النص
	font_style: string (str)	نوع خط النص
	font_size: int	حجم خط النص
	bold: bool	سمك الخط (نعم/لا)

العنوان الأول Label1: التعليمات البرمجية لإنشاء عنوان باستخدام (الطريقة المباشرة)، حيث يُظهر النص «الوقت والتاريخ الحالي»، معتمدًا على الهيكل الرسومي التالي:





لاحظ

موقع العنصر Location داخل النافذة يحدد من بداية الزاوية العلوية اليسرى حسب الإحداثي السيني والصادي للنافذة (X,Y).

ويمكن إنشاء عنصر (Label2) باستخدام (الطريقة غير المباشرة) وإسناده لمتغير، مثال:

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

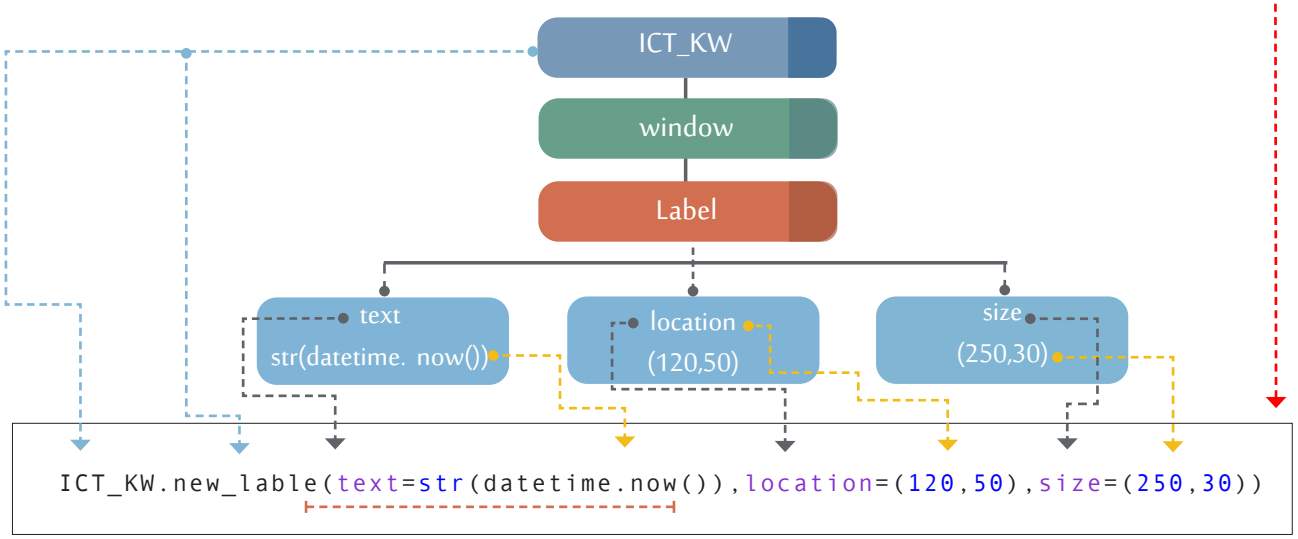
العنوان الثاني Lable2:

لإضافة عنوان داخل النافذة الرئيسية يظهر فيه التاريخ والوقت باستخدام دالة ( )datetime.now من مكتبة datetime التي سبق وتعلمتها، باتباع الخطوات التالية:

- استيراد المكتبة datetime باستخدام الأمر:

```
from datetime import datetime
```

- لإنشاء عنوان label يعرض قيمة الدالة ( )datetime. now في الخاصية Text، وتحويل قيمتها إلى نص نستخدم الدالة ( )str كما هو موضح بالشكل التالي: .



عند تشغيل البرنامج تظهر النافذة الرئيسية بالشكل التالي:

2 - الصورة Image

عنصر مرئي يعرض صورة.

الصيغة Syntax: Image_name(Optional) = ICT_KW. new_Image(Properties)

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

إضافة ملف "صورة" إلى مجلد المشروع FirstGUI

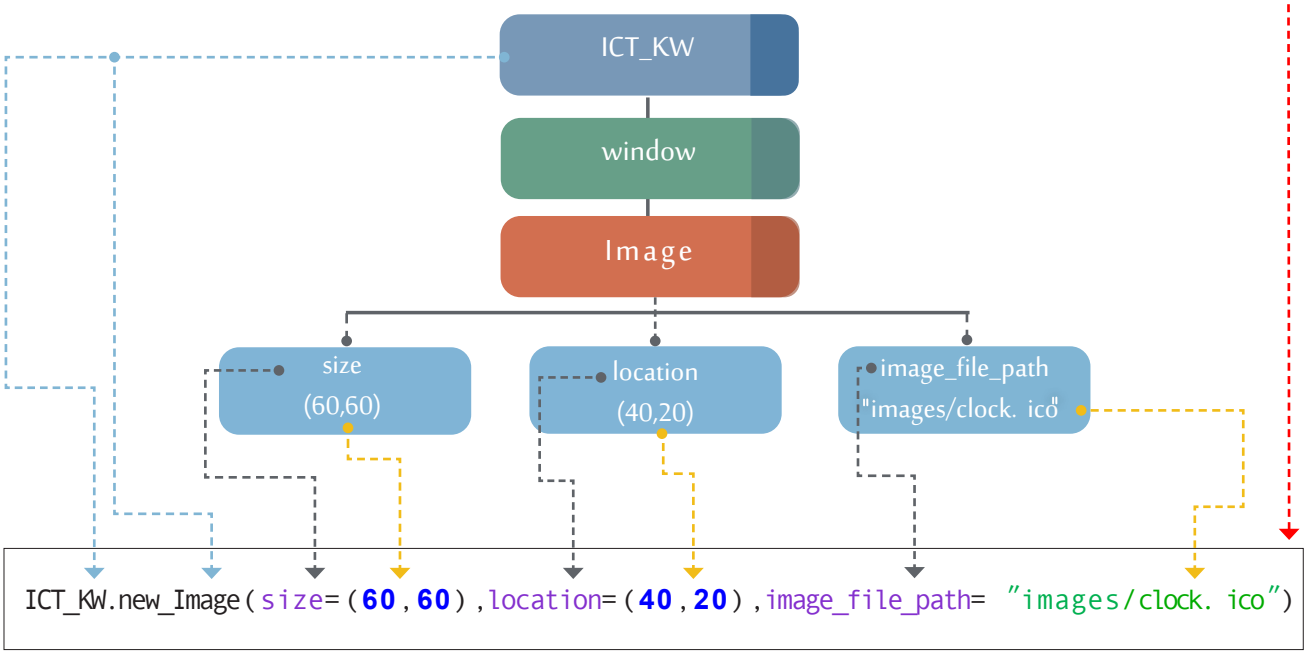
استيراد ملف الصورة clock.ico من مجلد الصور إلى مجلد المشروع باتباع الخطوات التالية:

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.

سبق وأن تم تنفيذ نفس الخطوات في درس استيراد مكتبة ICT_KW.

التعليمات البرمجية لإضافة عنصر الصورة:

يتم تحويل الهيكل الرسومي إلى تعليمات برمجية



لاحظ : من أمثلةملفات الصور المدعومة بالمكتبة (.ico ،.png ، .jpg).

عند تشغيل البرنامج تظهر النافذة الرئيسية بالشكل التالي:

الخطوة الثانية: تصميم الاستجابة للأحداث

لا يحتوي المشروع الحالي على عناصر تستجيب لمدخلات المستخدم ولكن سنتعلمه لاحقاً في درس الدوال Functions.

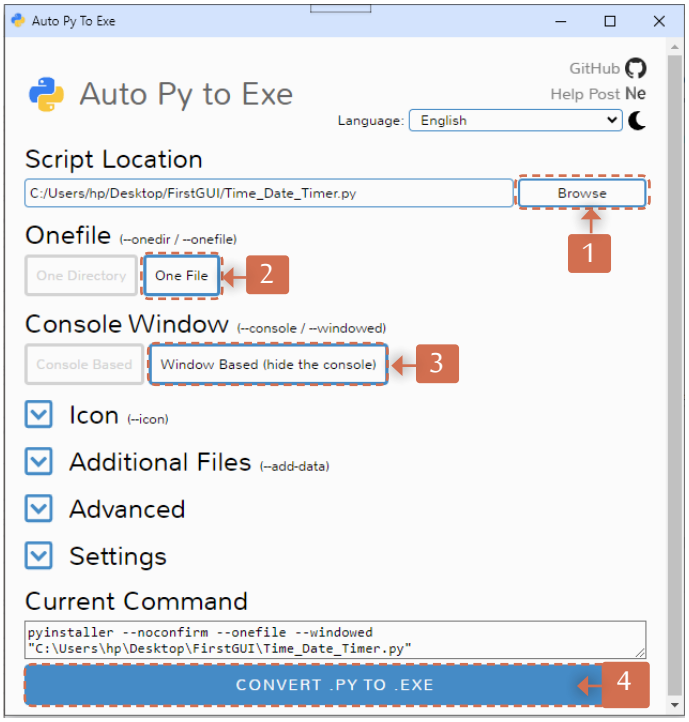
الخطوة الثالثة: اختبار البرنامج

عند تنفيذ البرنامج تظهر النافذة التالية:

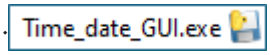
الخطوة الرابعة: تصدير البرنامج

يتم تحويل ملف (py.) إلى ملف تنفيذي (exe.)، وتشغيله مباشرة دون الحاجة للدخول إلى التعليمات البرمجية في (Pycharm)، هناك العديد من الطرق ومنها مكتبة (auto-py-to-exe) حيث توفر واجهة رسومية تتيح للمستخدم تحديد اختياراته، وذلك من خلال اتباع الخطوات التالية:

- استخدام الأداة Terminal .
- تحميل مكتبة auto-py-to-exe بكتابة الأمر pip install auto-py-to-exe>> ثم ضغط مفتاح الإدخال.
- كتابة auto-py-to-exe ثم ضغط مفتاح الإدخال.
- يظهر صندوق المحاورة التالي:



1. تحديد مسار المشروع.
2. تحويل الملف وجميع محتويات المشروع في ملف واحد.
3. إخفاء شاشة محرر الأوامر.
4. ضغط زر Convert.

- يظهر الملف التنفيذي في داخل مجلد المشروع 

.
- تشغيل البرنامج بالضغط على الملف Time_date_GUI.exe.

لاحظ

تأكد من وجود مجلد الصور images في مجلد output الخاص بمشروعك.

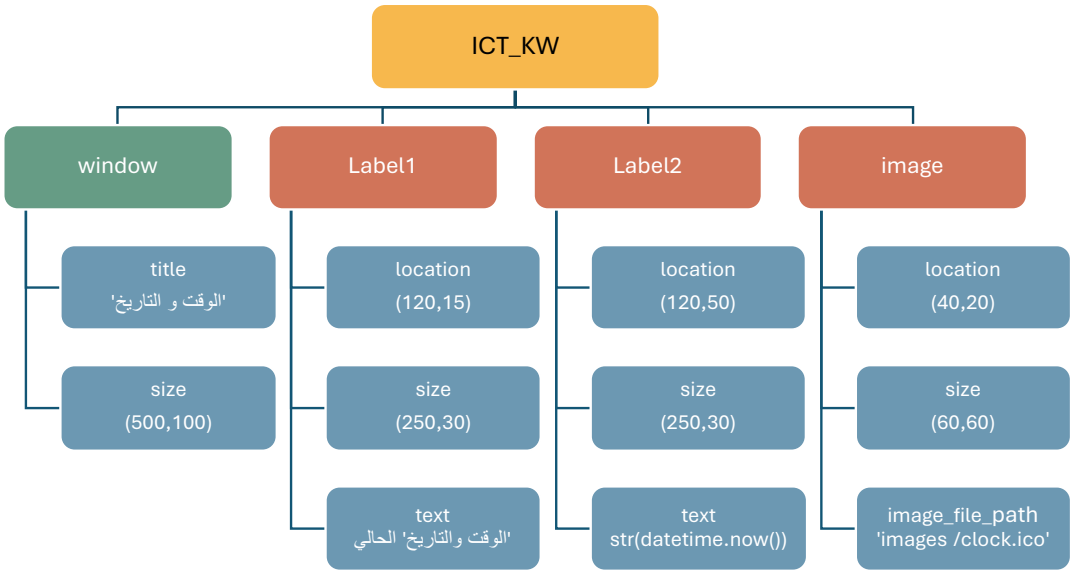
أوراق العمل

ورقة عمل (1):

اسم البرنامج: الوقت والتاريخ.

فكرة البرنامج: تصميم برنامج يعرض الوقت والتاريخ في واجهة مستخدم رسومية، متبعًا الخطوات التالية:

- شغل الملف التنفيذي Time&date.exe وعاين المنتج النهائي.
- شغل برنامج PyCharm.
- افتح المشروع First_GUI، ثم أنشيء 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.



- اختبر البرنامج.
- صدر البرنامج.

اكتب التعليمات البرمجية

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

13. .....

14. .....

15. .....

16. .....

17. .....

18. .....

19. .....

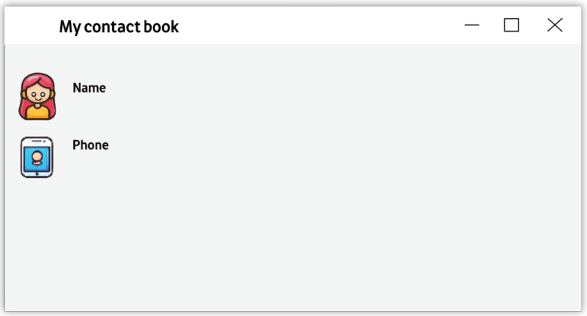
20. .....

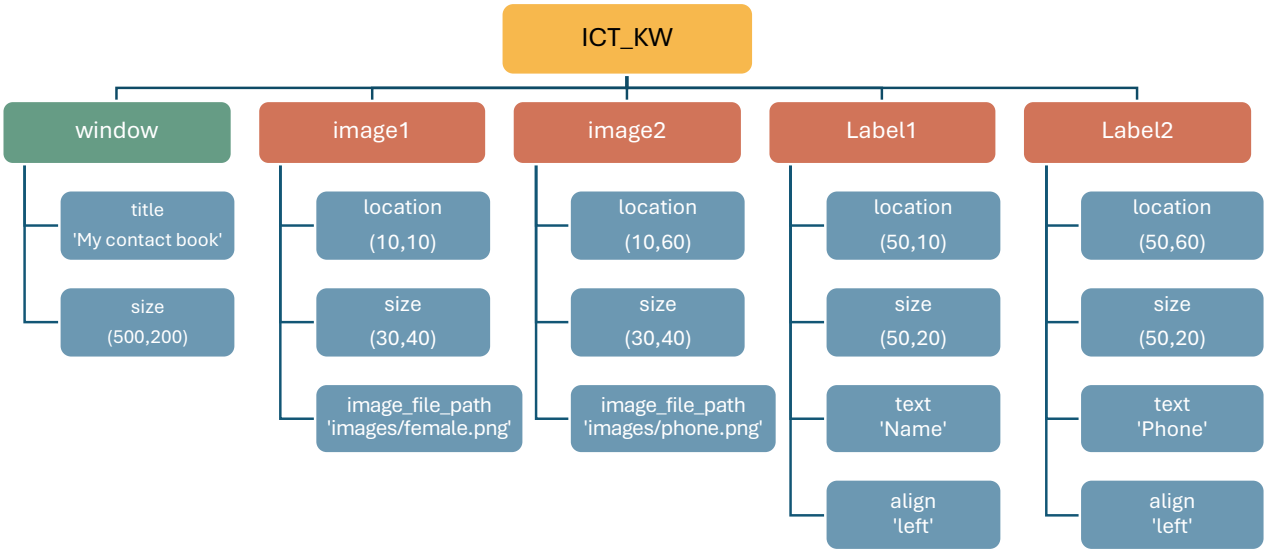
ورقة عمل (2):

اسم البرنامج: My Contact Book

فكرة البرنامج: تصميم برنامج جهة الاتصال يعرض اسم المتصل ورقم هاتفه في واجهة مستخدم رسومية، متبعًا الخطوات التالية.

- شغل الملف التنفيذي My Contact Book.exe، ثم عاين المنتج النهائي.
- شغِّل برنامج PyCharm.
- افتح المشروع My Contact Book، ثم افتح ملف main.py.
- استدع مكتبة ICT_KW.
- صمم الواجهة الرسومية التالية بكتابة التعليمات البرمجية المناسبة مستعيناً بالمخطط في الأسفل.





- اختبر البرنامج
- صدر البرنامج

اكتب التعليمات البرمجية

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

13. .....

14. .....

15. .....

16. .....

17. .....

18. .....

19. .....

20. .....

برمجة Python

الوحدة الأولى

الـــــدوال
Functions

1. مدخل إلى الدوال
2. مفهوم الدالة
3. أنواع الدوال ومميزاتها
4. إنشاء الدوال
5. استدعاء دالة

الدوال Functions

نتاجـــات التعـلم

الجزء الأول (أساسيات الدوال)

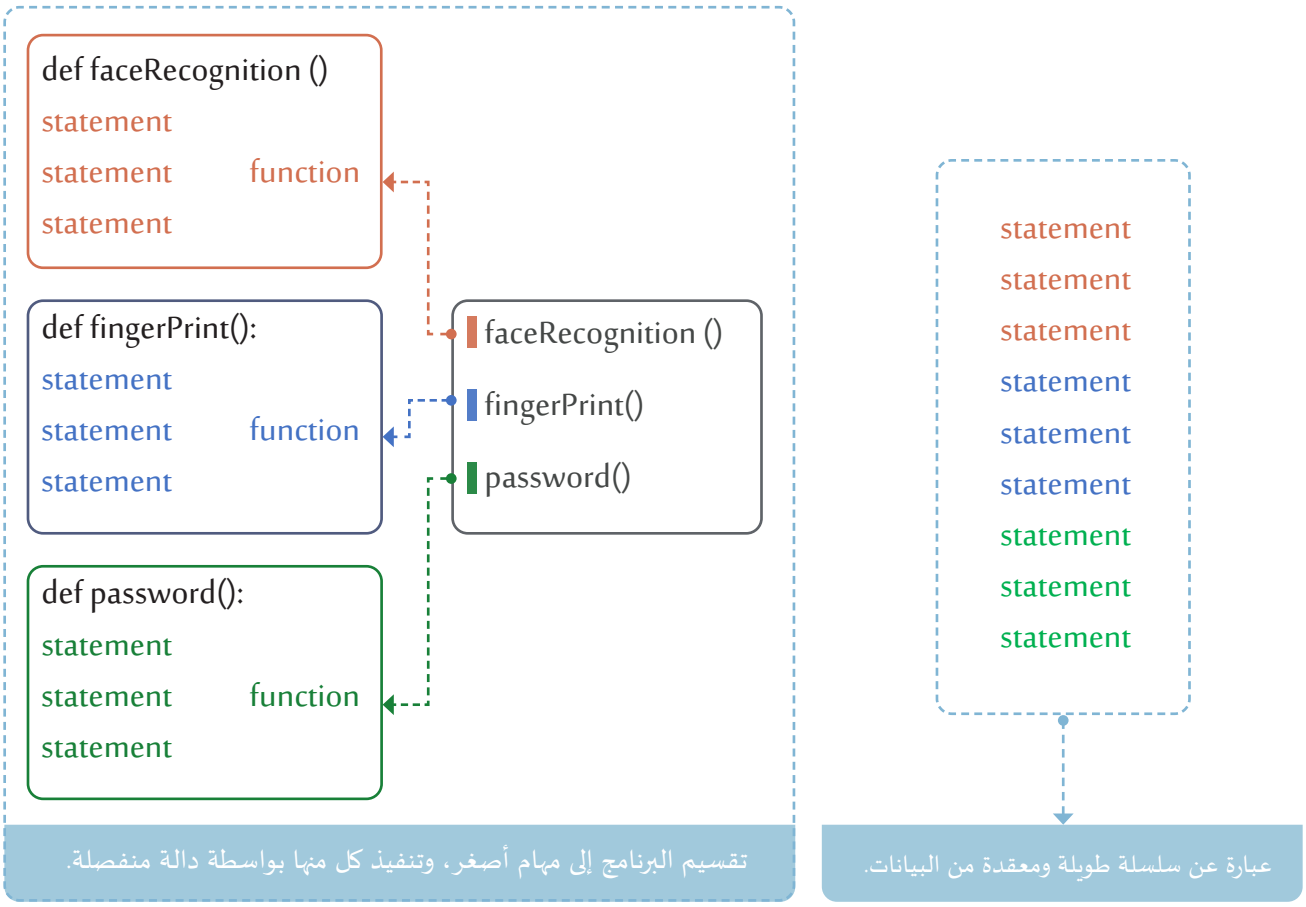
- التعرف على أهمية الدوال، واستخدامها لتنظيم التعليمات البرمجية.
- كتابة دالة بسيطة باستخدام الكود المناسب.
- استخدام المتغيرات المحلية داخل الدوال والمتغيرات العالمية.
- تمرير الوسائط إلى الدوال واستخدامها داخل التعليمات البرمجية.
- إرجاع القيم من الدوال وكيفية التعامل مع هذه القيم.
- التعرف على بعض الدوال المدمجة في Python.

مدخل إلى الدوال

تعتبر الدوال (Functions) أحد أدوات البرمجة الأساسية في Python، التي تساعد على كتابة التعليمات البرمجية بصورة منظمة لسهولة التعامل معها.

مفهوم الدالة Function Concept

مجموعة من التعليمات البرمجية لها مهام محددة تنفذ عند استدعائها، وهناك بعض البرامج تؤدي مهامًا كبيرة بما يحتاج لتقسيمها إلى مهام فرعية (دوال)، وتكرارها في عدة مواضع في البرنامج؛ مما يسهل تعديلها وتطويرها بشكل مستمر.



شكل رقم 1 - مفهوم الدالة (استخدام الدوال في كتابة وتنظيم البرنامج).

أنواع الدوال Functions Types

الدوال المدمجة Built-in functions: الدوال التي تأتي مع Python بشكل افتراضي، وتستخدم دون الحاجة إلى استيرادها مثل: print( ) – input( ) – type( )– len( ) – max( ).

لمعلوماتك

يمكن التعرف على كافة الدوال المدمجة في Python من خلال مسح QR المقابل:

الدوال المعرَّفة من المستخدم User-defined functions: الدوال التي ينشِئُها المستخدم والتي تمثل مجموعة من الأسطر البرمجية التي تنفذ مهام محددة.

الدوال المجهولة Lambda functions: الدوال التي تُستخدم لتنفيذ وظائف بسيطة ومباشرة، وتُكتب في سطر واحد دون استخدام def للتعريف عنها.

مميزات استخدام الدوال:

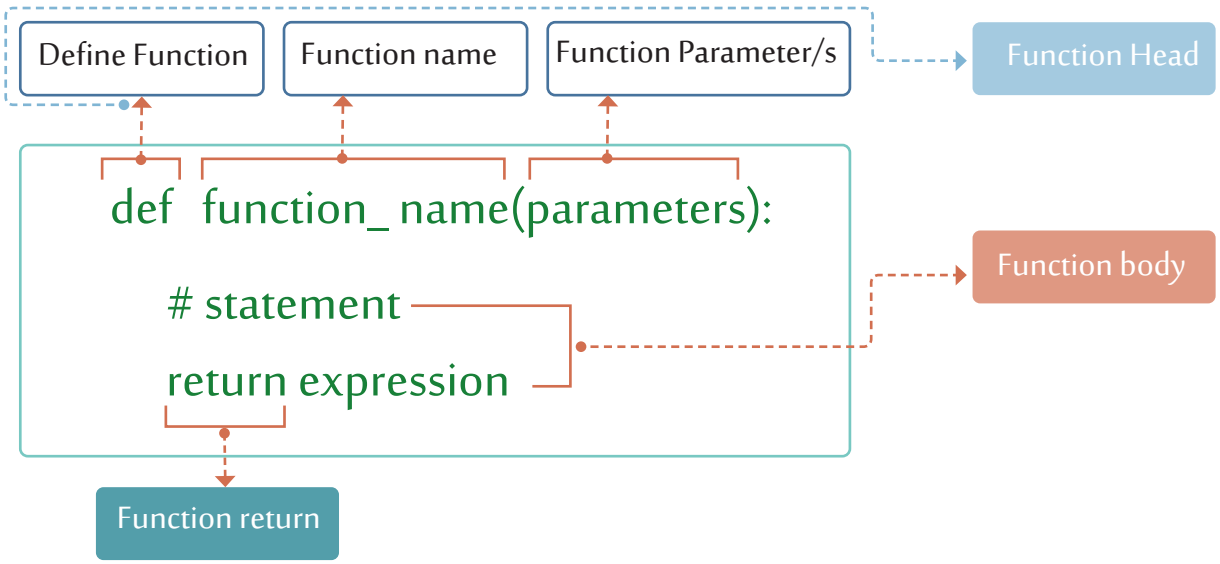
توفر الدوال في Python العديد من المميزات التي تساعد المستخدمين على:

- تسهيل كتابة وتطوير البرامج، حيث تكون التعليمات البرمجية أبسط وأسهل في الفهم والاستخدام عندما يتم تقسيمها إلى دوال متعددة.
- إعادة استخدام الدوال أكثر من مرة في البرنامج.
- اختبار البرنامج بشكل أفضل، وهذا يجعل من السهل التعرف على الأخطاء وإصلاحها.

إنشاء الدوال function

إنشاء الدوال في Python يعد من المهارات الأساسية لتنظيم التعليمات البرمجية وإعادة استخدامها، والتي يمكن استدعاؤها في أي وقت عند الحاجة.

1. صيغة الدالة Function Syntax



Function Head

- تعريف الدالة Define a function: نستخدم الكلمة المفتاحية def وهي اختصارلكلمة (definition) لتعريف الدالة، وهي من الكلمات المحجوزة في Python.

- اسم الدالة Function name: اسم يُعبرعنها، ويعكس وظيفتها، ويتم اتباع قواعد تسمية المتغيرات عند تسمية الدوال في Python، راجع صفحة 56 في كتاب الجزء الأول.

- معاملات الدالة Function Parameter / s: هي متغيرات يتم تعريفها في الدالة، ويتم تعيين قيم لها عند استدعاءها، والتي يتم تمريرها للدالة ويمكن أن تكون فارغة.
ويلي اسم الدالة أقواس دائرية Parentheses حيث يُعرَّف معامِلات الدالة بداخل الأقواس، ثم يُتبَع بالنقطتين (:) colon.

Function Body

يشمل مجموعة من التعليمات البرمجية داخل الدالة (لاحظ المحاذاة Indentation)، لأن مفسّر Python Interpreter يستخدمها للإعلان عن بداية ونهاية التعليمات البرمجية.

- التعليمات Statements: مجموعة من الأوامر البرمجية التي تُنفذ عند استدعاء الدالة.
- الإرجاع Return Expression: يمكن استخدام العديد من الدوال والتعليمات منها:

return statement: تُستخدم لإنهاء تنفيذ استدعاء الدالة و«إرجاع» نتيجتها (قيمة التعبير الذي يلي return expression).

دالة print: تُستخدم لطباعة القيم، ولكن بدون إعادة قيمة يمكن استخدامها في باقي التعليمة البرمجية.

استدعاء دالة Function Calling

لاستدعاء الدالة لتنفيذ التعليمات البرمجية:

- كتابة اسم الدالة متبوع بالأقواس الدائرية ( ) Parentheses.
- يمكن تمرير قيم (Arguments) للدوال التي تتضمن معاملات (Parameters).

Parameters (المعاملات): المتغيرات التي تعرف في رأس الدالة. وتمثل أماكن القيم التي يتم تمريرها إلى الدالة لاحقًا، ويتم تحديدها عند إنشاء الدالة.

Arguments (القيم المُرسَلة): القيم الفعلية التي تُمرر إلى الدالة عند استدعائها. هذه القيم تحل محل Parameters داخل الدالة.

- تنفيذ التعليمات البرمجية الموجودة في جسم الدالة.
- الانتقال مرة أخرى إلى جزء البرنامج الذي استدعى الدالة.
- استئناف البرنامج التنفيذ عند هذه النقطة.

مثال 1: إنشاء دالة (بدون معاملات)

التفسير: إنشاء دالة لطباعة جملة ترحيبية عند استدعائها.

Def_Greeting

```
1    def greet(): # Function Head
2        print("Hello, Kuwait!") # Function Body
3
4    greet() # Function Calling
5
```

Program Output

```
Hello, Kuwait!
```

مثال 2: إنشاء دالة (مع معاملات)

التفسير: إنشاء دالة لطباعة جملة ترحيبية عند استدعائها، وإدخال قيمة المعامل (Parameters).

Def_Argument

```
1    def add_numbers(num1, num2): # Parameters (num1, num2)
2        print (num1+num2)
3
4    add_numbers(10,30)# Arguments (10, 30)
```

Program Output

```
40
```

مثال 3: استدعاء برنامج يحتوى على أكثر من دالة.

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

Program 2

```
1   # Create an empty shopping cart
2   shopping_cart = [ ]
3
4   def add_item(shopping_cart, item):
5       # Add item to cart
6       shopping cart.append(item)
7       print(f"This item: {item} added to shopping cart")
8
9   def view_cart(shopping_cart):
10      # Show all items in shopping cart
11      if not shopping_cart:
12          print("Your shopping cart is empty.")
13      else:
14          print("Shopping cart contains:")
15          for item in shopping_cart:
16              print(f"- {item}")
17
18
19  # Add some items
20  add_item(shopping_cart, "Apple")
21  add_item(shopping_cart, "Banana")
22  add_item(shopping_cart, "Bread")
23
24
25  #View shopping cart
26  view_cart(shopping_cart)
```

Program Output

```
This item:  Apple added to shopping cart
This item:  Banana added to shopping cart
This item:  Bread added to shopping cart
Shopping cart contains:
- Apple
- Banana
- Bread
```

مثال 4: دالة ترجع قيمة

التفسير: إنشاء دالة لجمع عَدَدَيْن وإرجاع الناتج باستخدام Return Expression.

Def_Sum

```
1  def add_numbers(a,  b) :
2      return  a + b
3
4  result = add_numbers(5, 3)
5
6  print (result)
```

Program Output

```
8
```

يمكنك تعريف دوال تحتوي على معامل واحد أو أكثر.
تستخدم return لإرجاع نتيجة معالجة الدالة عند استدعائها.

إثرائي (مثال5): إنشاء دالة (بمعامِلات غير محددة * Args)، والإرجاع return بأكثر من متغير.

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

```
1   def student_result(*degrees):
2       total_sum = 0
3       number_degrees = 0
4
5       for degree in degrees:
6           number_degrees += 1
7           total_sum += degree
8       average_degrees = total_sum / number_degrees
9       return average_degrees, total_sum
10  average_degrees, total_sum = student_result(80, 85, 99, 70, 88)
11  print (f"Total student degree: {total_sum}")
12  print (f"Average student degree: {average_degrees}")
13
```

Program Output

```
Total student degrees:   422
Average student degrees:   84.4
```

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

لاحظ: ترتيب إدخال المعاملات

```
def student_result(grade, *degrees):
    total_degrees = sum(degrees)
    average_degrees = total_degrees / len(degrees)
    return average_degrees, total_degrees, grade

average_degrees, total_degrees, grade = student_result(12, 80, 85, 99, 70, 88)

print(f"Student Grade: {grade}\nTotal Student Degrees: {total_degrees}")
print(f"Average Student Degrees: {average_degrees}")

```

Program Output

Student Grade: 12

Total student degrees: 422

Average student degrees: 84.4

ادرس التعليمات البرمجية التالية، وحدد وظيفة الجزء المحدد منها:

```
def student_result(*degrees):
    # total degrees
    total_degrees = sum(degrees)
    max_total = len(degrees)*100
    print(f" Sum of student's degrees: {total_degrees} from {max_total} degrees")
    rate = (total_degrees / max_total) * 100
    # Average of degrees
    print(f" Rate of student's degrees   : {rate:.2f} %")
    # Success Status
    print("Congratulations, You succeeded") if rate >= 50  else print("Sorry! You failed")

student_result(100, 80, 90, 95, 50.5)

```

إثرائي (مثال6): القيم الافتراضية لمعاملات الدوال default parameters

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

```
1   def student_grade(name, grade = "Excellent"):
2       print (f"Hi {name} your grade is {grade}")
3
4   student_grade("Ali") # Hi Ali your grade is Excellent
5   student_grade("Ali", "Good") # Hi Ali your grade is Good
```

Program Output

Hi Ali your grade is Excellent

Hi Ali your grade is Good

المعاملات الافتراضية لا تسبق في ترتيب كتابتها المعاملات التي تتطلب قيم argument.

لمعلوماتك

دوال lambda (مجهولة الاسم Anonymous) واستخدامها في Python

- تسمى هذه الدوال بالدوال المجهولة، وهي دوال بسيطة تكتب من سطر واحد.
- لا يتم الإعلان عنها بالطريقة القياسية باستخدام الكلمة المحجوزة (def).
- يمكن أن تأخذ أي عدد من (Arguments) ولكنها تُرجع قيمة واحدة فقط في شكل تعبير قصير.

والصيغة العامة لإنشاء دالة (Lambda) في بايثون تكون: lambda arguments : expression

وتوضح الأمثلة التالية طريقة إنشاء دوال Lambda.

إثرائي (مثال 7): إنشاء دالة Lambda لجمع عددين

```
1  # Create add lambda function
2  add = lambda x, y: x + y
3
4  print(add(5, 10))
5
```

Program Output

```
15
```

إثرائي (مثال 8): إنشاء دالة Lambda لحساب الأُسّ التربيعي لعدد

```
1  # Create exponention lambda function
2  exponention = lambda x: x ** 2
3
4  print(exponention(5))
```

Program Output

```
25
```

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

التفسير: برنامج ينشئ دالة لاستقبال عدد من المستخدم، وطباعة ما إذا كان فرديًا أم زوجيًا.

```
1  # Create even/odd lambda function
2  even_odd = lambda  x : print ("even") if  x%2  ==  0 else print("odd")
3
4  even_odd (float(input("Enter a number: ")))
```

Program Output

```
Enter a number:  25
odd
```

نطاق المتغيرات Scope

مفهوم يحدد مكان وكيفية الوصول إلى المتغيرات داخل البرنامج.

المتغير المحلي Local Variable: المتغيرات التي تُعرَّف داخل دالة من التعليمات تكون محلية، مما يعني أنها يمكن الوصول إليها فقط داخل تلك الدالة.

مثال 10: مجال المتغير المحلي في الدالة.

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

```
1  def my_function():
2      local_var = 10
3      print(local_var)
4
5  my_function()
6
7  print(local_var)
```

Program Output

NameError: name 'local_var' is not defined (رسالة خطأ)

مثال 11: مجال المتغير العام في الدالة

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

```
1  def my_function():
2      local_var = 10
3      global  global_var
4      global_var  =  500
5      print(f"Local Variable is: {local_var}")
6
7  my_function()
8
9  print(f"Global Variable is: {global_var}")
```

Program Output

Local Variable is: 10

Global Variable is: 500

إثرائي مثال 12: متوالية فيبوناتشي Fibonacci Generator

هي سلسلة الأرقام: 0، 1، 1، 2، 3، 5، 8، 13، 21، 34، . . .

حيث يتم إيجاد الرقم التالي في السلسلة عن طريق جمع الرقمين السابقين له، على سبيل المثال:

- الحصول على الرقم 2 (العنصر الرابع) بجمع الرقمين السابقين له (1+1).
- الحصول على الرقم 3 (العنصر الخامس) بجمع الرقمين السابقين له (1+2).
- الحصول على الرقم 5 (العنصر السادس) بجمع الرقمين السابقين له (2+3).
- . . . . . .
- ومن ثم للحصول على الرقم 55 بجمع الرقمين السابقين له (21+34)، وهكذا.

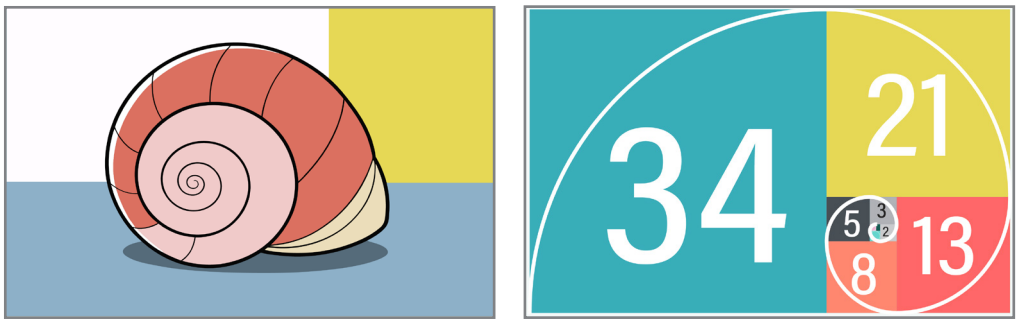
النسبة الذهبية Golden Ratio: قيمة رياضية تساوي تقريبًا 1.618، ويرمز إليها بالحرف اليوناني (Φ)، وترتبط ارتباطًا وثيقًا بمتتالية فيبوناتشي، حيث يقترب حاصل قسمة أي عددين متتاليين في السلسلة من قيمة النسبة الذهبية كلما زادت الأعداد.

يمكن الحصول على النسبة الذهبية في سلسلة فيبوناتشي بقسمة أي عددين متتاليين (العدد الأكبر / العدد الأصغر)، أو جمع أي عددين متتاليين وتقسيم الناتج على أكبرهما، ويقترب ناتج القسمة من قيمة النسبة الذهبية 1.618 كلما زادت أعداد السلسلة.

أمثلة على النسبة الذهبية في الحياة

- الهندسة والفن: استخدم الفنانون القدماء، مثل ليوناردو دافنشي، النسبة الذهبية في لوحاتهم (مثل لوحة الموناليزا) لتحقيق توازن جمالي.
- العمارة: يمكن ملاحظة النسبة في تصميم البارثينون في اليونان وبعض المباني الحديثة.
- الطبيعة: تظهر النسبة في نمو النباتات، حيث تتبع الأوراق ترتيبًا معينًا (فيبوناتشي)، وفي قواقع البحر، وحتى في ترتيب الكواكب.

تابع مثال 12: متوالية فيبوناتشي Fibonacci Generator



شكل رقم 2 - (أمثلة على النسبة الذهبية في الحياة).

البرنامج التالي يوضح كيفية الحصول على سلسلة فيبوناتشي مع حساب النسبة الذهبية، قدم حلول برمجية بديلة للحصول على سلسلة فيبوناتشي وحساب النسبة الذهبية.

```
1   def  fibonacci (number_fibonacci):
2       number_1 = 0
3       number_2 = 1
4       sum_number  = 1
5       if number_fibonacci  >  3:
6           print (number_1, number_2, sum_number, end=" ")
7           for i  in range(3 ,  number_fibonacci):
8               number_1, number_2 = number_2, sum_number # Swap variables values
9               sum_number = number_1 + number_2
10              print(sum_number, end=" ")
11              # the golden ratio is the sum of two numbers divided by the larger
12          golden_ratio = sum_number / number_2
13          print(f"\nThe Golden Ratio is= {golden_ratio: .2f}")
14      else:
15          print("Enter number more than 3")

fibonacci(int(input("Enter number of Fibonacci Sequence :")))
```

أوراق العمل

ورقة عمل 1:

اسم البرنامج: آلة حاسبة بسيطة باستخدام الدوال.

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

المطلوب: 1- استدعاء الملف Simple_Calc.py من مجلد أوراق العمل.

```
1   def addition(a, b):
2       return a + b
3   def subtraction(a, b):
4       return a - b
5   def multiplication(a, b):
6
7   def division (a, b):
8       if
9           return ("Sorry,Cannot divide by zero!!")
10      return a / b
11
12  def simple_calculator():
13          number1=float(input("Enter first  number : "))
14          number2=float(input("Enter second number : "))
15
16          user_input=input("Enter your choice : +  -  *  / :")
17          if user_input == "+":
18              result = addition(number1, number2)
19          elif user_input == "-":
20              result =subtraction(number1, number2)
21
22
23          elif user_input == "/":
24              result =division(number1, number2)
25
26          else:
27              print("Invalid input.")
28              exit() #دالة تُستخدم لإنهاء البرنامج
29
30          print(result)
31  simple_calculator()
32
```

تابع : ورقة عمل 1:

2- اكتب التعليمات البرمجية اللازمة:

- إجراء جميع العمليات الحسابية (الجمع، الطرح، الضرب، القسمة) بشكل صحيح.
- التأكد من ظهور رسالة تنبيهة في حال القسمة على صفر.

Output

```
Enter first number : 9
Enter second number : 0
Enter your choise : + * - / : /
Sorry, Cannot divide by zero !!
```

اكتب التعليمات البرمجية

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

تطوير البرنامج:

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

اكتب التعليمات البرمجية

01. .....
02. .....
03. .....
04. .....
05. .....
06. .....
07. .....
08. .....
09. .....
10. .....
11. .....
12. .....
13. .....
14. .....

إثرائي: ورقة عمل 2

اسم البرنامج: اكتشاف المواقع غير الموثوقة

المطلوب:

1. استدعاء الملف Suspicious_domains.py من مجلد أوراق العمل.

```
# List of suspicious domains
suspicious_domains = ["hacker.com", "fake.com", "phishingsite.com"]

# Function to view suspicious domains
def view_suspicious(suspicious_domains):
    print(suspicious_domains)

# Function to add a suspicious domain
def add_suspicious(suspicious_domains, email):
    domain = email.split('@')[-1]  # Extract domain
    if domain not in suspicious_domains:
        suspicious_domains.append(domain)  # Add domain only
        print(f"{domain} added")
    else:
        print(f"{domain} already found")

# Main loop
while True:
    print("\n[1] Detect & Add Email\n[2] Print Domain List\n[3] Exit")
    choice = input("Enter Your Choice: ")
    if choice == "1":
        email = input("Enter Email: ")
        if "@" in email:
            add_suspicious(suspicious_domains, email)
        else:
            print("Invalid email. Please include '@'.")
    elif choice == "2":
        view_suspicious(suspicious_domains)
    elif choice == "3":
        break
    else:
        print("Wrong Number")
```

تابع إثرائي: ورقة عمل 2

2. أجب عن الأسئلة التالية:

a. ما وظيفة من البرنامج السابق؟

b. سجل التعليمات البرمجية المناسبة لتفيذ المهام التالية:

- إذا تم إدخال بريد إلكتروني غير صحيح.

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

c. ما ناتج تنفيذ التعليمات البرمجية التالية:

```
def add_suspicious(suspicious_domains, 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")

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

الوحدة الأولى برمجة Python

الـــــدوال
Functions
(توظيف الدوال في الواجهة الرسومية)

1. تعديل النافذة الرئيسية وعناصرها
2. إنشاء العنصر (الزر Button)، وتفعيله.
3. إنشاء العنصر (المؤقت Timer)، وتفعيله

توظيف الدوال في الواجهة الرسومية

نتاجـــات التعـلم

الجزء الثاني (توظيف الدوال في واجهة المستخدم الرسومية)

- تعديل خصائص العناصر بعد إنشائها وإضافة خاصية جديدة ومنها:
  - لون التعبئة Background_color.
  - لون الخط Fore_color.
- التعامل مع عناصر الواجهة الرسومية Widgets للمكتبة (ICT_KW) وضبط خصائصها:
  - الأزرار Buttons .
  - المؤقت Timer.
- تصميم المشروعات التي تبرز أهمية الدوال في واجهة المستخدم الرسومية.

توظيف الدوال في الواجهة الرسومية

توظيف الدوال في الواجهة الرسومية

تعديل النافذة الرئيسية وعناصرها Widgets:

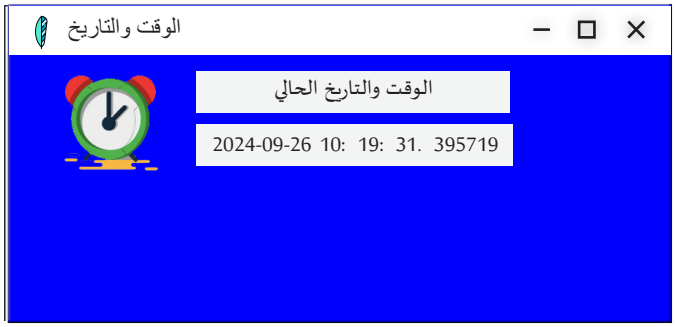
سنتعلم في هذا الفصل كيفية تعديل خصائص النافذة الرئيسية مثل (العنوان والحجم)، وإضافة خصائص جديدة، (تأكد من التنفيذ على الملف time_date_GUI.py).

أولًا: تعديل خصائص النافذة الرئيسية

يمكن تعديل خاصية الحجم وإضافة لون تعبئة للنافذة باستخدام التعليمة البرمجية التالية:

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

نلاحظ عند تشغيل البرنامج تغير حجم ولون النافذة الرئيسية إلى اللون الأزرق:



تذكّـر

توضيح: سبق وتعلمنا عن نظام الألوان RGB (Red, Green, Blue)، وهو نظام لوني يستخدم لتمثيل الألوان على الشاشات الإلكترونية مثل شاشات الحاسوب والتلفزيون.

حيث يتم دمج الألوان الأساسية الثلاثة (الأحمر، الأخضر، والأزرق) بنسب مختلفة تتراوح بين (0،255) لإنتاج مجموعة واسعة من الألوان، على سبيل المثال:

background_color = (Red,Green,Blue)

استعرض الجدول التالي، ثم اكتب اللون الناتج بعد تعديل قيم ألوان RGB في البرنامج

	أزرق Blue	أخضر GREEN	أحمر RED	اللون الناتج
نظام الألوان RGB	0	0	255	أحمر RED
	0	255	0	أخضر GREEN
	255	0	0	أزرق Blue
	0	0	0	. . . . .
	255	255	255	. . . . .

ثانيًا: تعديل خصائص العناصر

يمكن تعديل خصائص العناصر باستخدام دالتها edit مستخدمًا الصيغة التالية:

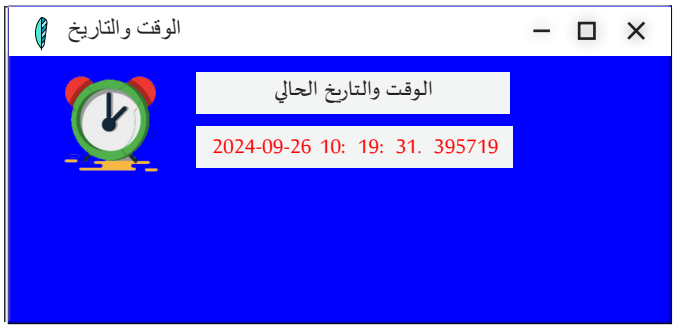
Child_name.edit(change Or add new Properties)

مثال: لتغيير خاصية اللون الأمامي للعنصر label2 نستخدم التعليمة البرمجية التالية:

```
label2.edit(fore_color = ( 255,0,0))
```

توظيف الدوال في الواجهة الرسومية

عند تشغيل البرنامج نلاحظ تغير لون خط Label2 إلى اللون الأحمر باستخدام الخاصية (fore_color):

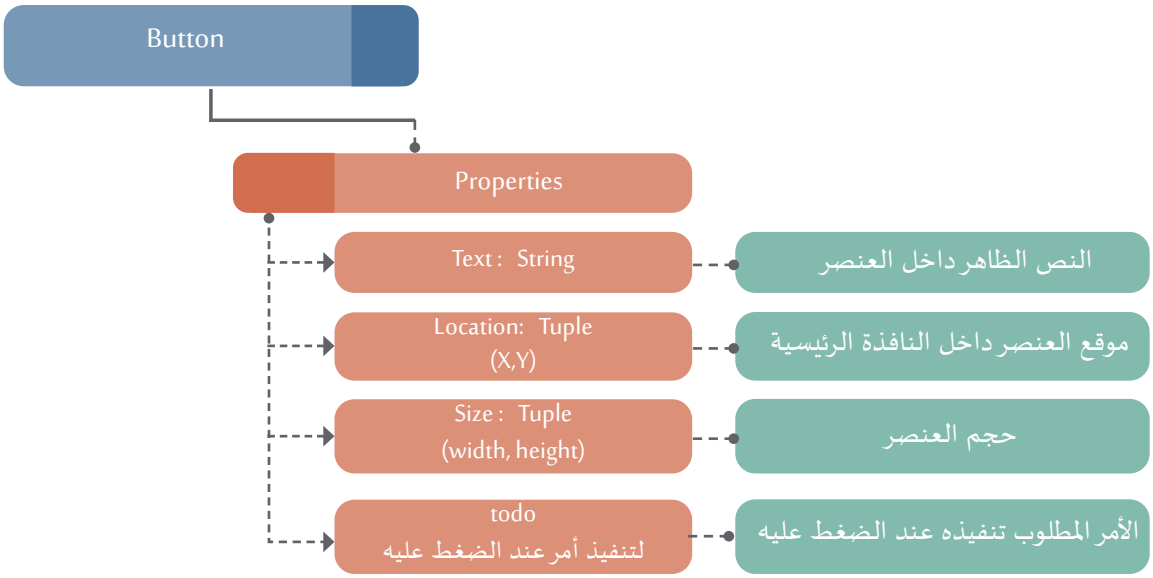


إنشاء العنصر (الزر Button)، وتفعيله.

عنصر تحكم مرئي عند الضغط عليه يستدعي دالة function لتنفيذها.

الصيغة Syntax:

Button_name(Optional) = ICT_KW.new_Button(Properies)

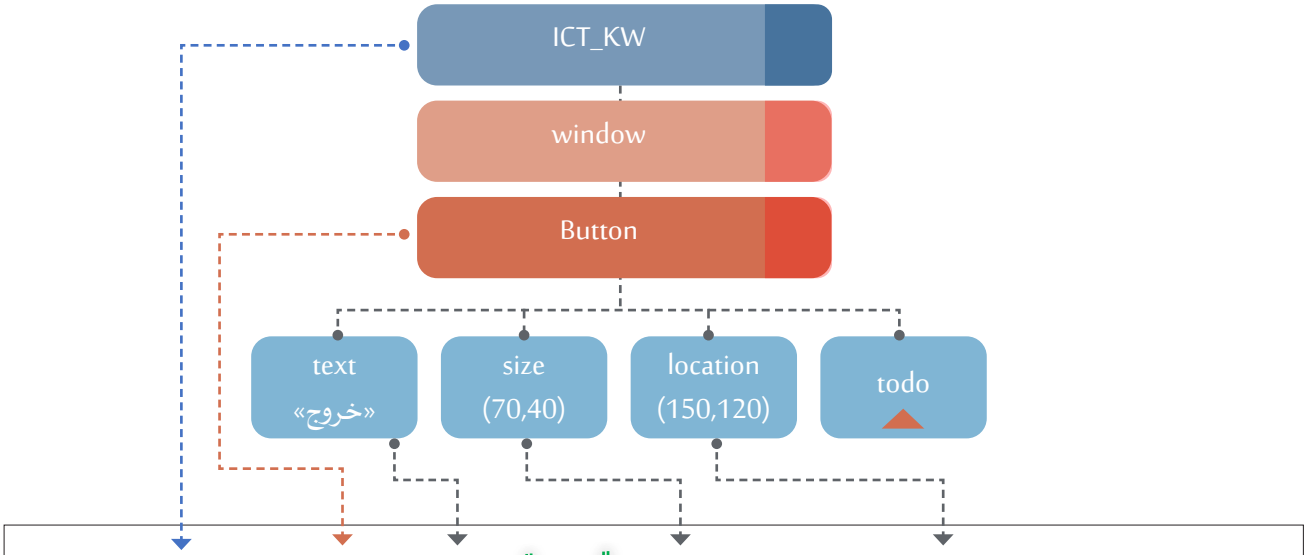


شكل يمثل الهيكل الرسومي لخصائص العنصر Button

وبعد استعراض الهيكل الرسومي لخصائص الزر Button، يمكننا كتابة التعليمات البرمجية بالاستعانة بالشكل التالي عند الضغط الزر، وذلك بطريقتين:

الطريقة الأولى: تفعيل الزر بالأمر المباشر (exit):

1. استكمال المشروع بإضافة زر الخروج وذلك من خلال الأمر التالي:




```
ICT_KW.new_Button( text= " خروج", size=(70, 40),location=(150, 120))
```

ما هي الخصائص التي تم تعيينها في الأمر السابق، دوّن إجابتك؟ . . . . . .

. . . . . .

2. عند تشغيل البرنامج نلاحظ أن الزر غير فعّال عند الضغط عليه، لذا يجب إضافة todo ليصبح الزر فعالًا، لتنفيذ أمر الخروج exit.

```
Button1= ICT_KW.new_Button(text= "خروج", size=(70, 40),location=(100,90),todo=exit)
```

لاحظ	الأمر exit هو أمر مباشر يتم تنفيذه عند الضغط على زر (خروج).

توظيف الدوال في الواجهة الرسومية

الطريقة الثانية: تفعيل الزر باستدعاء دالة function:

لاستدعاء الدالة edit_time عند الضغط على الزر Button2 نتبع الخطوات التالية:

1. إنشاء دالة باسم (edit_time):
في بداية البرنامج ضع مؤشر الكتابة بعد أوامر استدعاء المكتبات، ثم أعلن عن الدالة لتعديل التاريخ والوقت في العنصر Label2:

```
def edit_time():
    Label2.edit(text = str(datetime now()))
```

2. إنشاء زر Button2 وتفعيل خاصية todo:

```
Button2=ICT_KW.new_Button(text="تحديث التاريخ والوقت", size=(130,40),
location=(250,120),todo=edit_time)
```

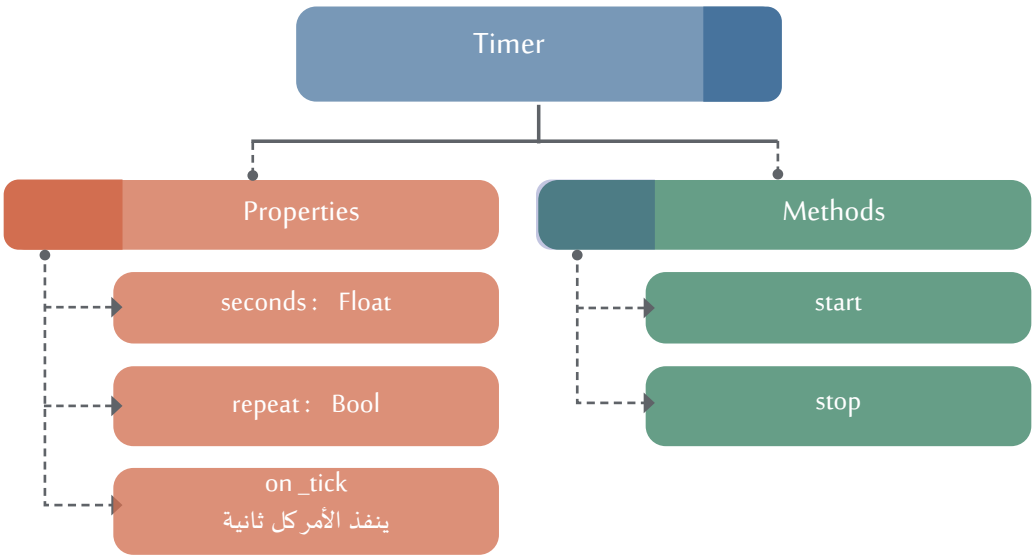
عند تشغيل البرنامج سيتم تحديث التاريخ والوقت كلما ضغط المستخدم على زر (تحديث التاريخ والوقت).

إنشاء العنصر (المؤقت Timer)، وتفعيله:

أداة تحكم غير مرئية لا تظهر على النافذة الرئيسية، تُستخدم لتكرار تعليمات برمجية خلال فترات زمنية محددة، ويمكن استخدام المؤقتات لأغراض متعددة مثل قياس مدة تنفيذ برنامج معين، إنشاء عدادات زمنية تنازلية، أو توقيت أحداث معينة.

الصيغة Syntax:

Timer_name(Optional) = ICT_KW.new_Timer(Properies).method( )



شكل يوضح الهيكل الرسومي لخصائص والدوال المدمجة للعنصر

يمكن التحكم بعمل المؤقت Timer بطريقتين:

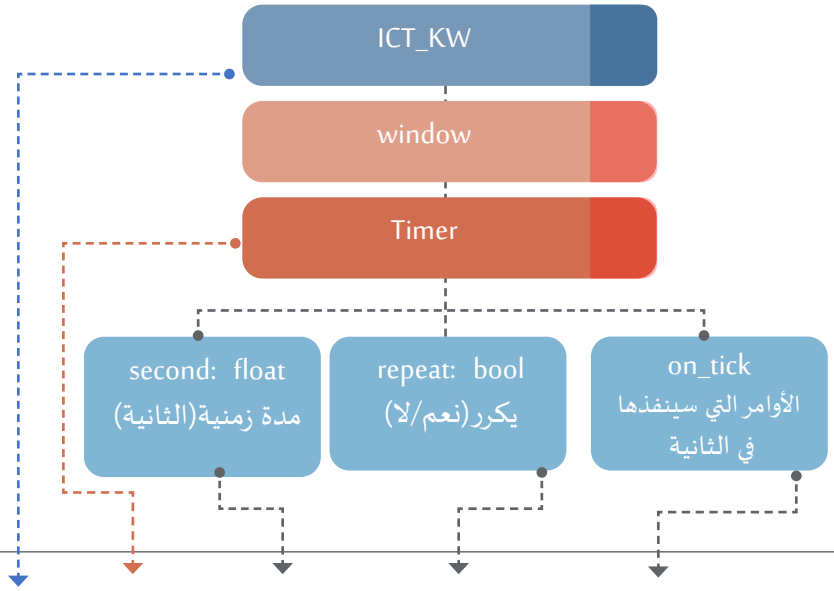
1. تشغيل: يتم بإحدى التعليمات البرمجية التالية:
   - ICT_KW.new_Timer(Properies).start
   - Child_name.start
2. إيقاف:
   - Child_name.stop

توظيف الدوال في الواجهة الرسومية

عدل في الملف time_date_GUI.py ليصبح كالتالي:

لاحظ	تم حذف الزر (تحديث التاريخ والوقت) وتغيير موقع Location الزر (خروج).

المطلوب: عند تشغيل البرنامج يتم تحديث التاريخ والوقت تلقائياً باستخدام عنصر المؤقت Timer.




```
ICT_KW.new_Timer(seconds=0.1, repeat=True,on_tick=edit_time).start()
```

عند تشغيل البرنامج نلاحظ أن عنصر المؤقت لم يظهر في النافذة الرئيسية ويتم تحديث التاريخ والوقت في عنصر العنوان Label2 باستمرار، ولإنهاء البرنامج اضغط على زر الخروج.

سيتم استكمال البرنامج السابق بإضافة زر Start و زر Stop، كما في الشكل التالي:

عند الضغط على زر start يتم تحديث الوقت والتاريخ باستمرار ويتوقف التحديث عند الضغط على زر stop، ولإنهاء البرنامج اضغط على زر الخروج.

لتنفيذ ذلك، نتبع الخطوات التالية:

1. إنشاء المؤقت Timer لتحديث الوقت والتاريخ باستخدام الدالة edit_time باستخدام التعليمة البرمجية التالية:

```
timer1=ICT_KW.new_Timer(seconds=0.1,repeat=True,on_tick=edit_time)
```

يتم تحديث الوقت والتاريخ من خلال الدالة edittime كل 0.1 جزء من الثانية

2. يتم التحكم في طريقة عمل المؤقت من خلال الزرين start وstop، كما يلي:

```
b_start=ICT_KW.new_Button(text="start",size=(70,40),location=(120,120),todo=timer1.start)
b_stop=ICT_KW.new_Button(text ="stop",size=(70,40),location=(280,120),todo=timer1.stop)
```

لاحظ

- في خاصية todo في الزر b_start نستخدم المتغير timer1 وطريقة العمل(start).
  وفي زر b_stop نستخدم طريقة العمل (stop).
- أهمية ترتيب الأوامر، والحرص على إنشاء المؤقت timer1 قبل الأزرار b_start و b_stop،لماذا؟

توظيف الدوال في الواجهة الرسومية

أوراق العمل

ورقة عمل 1:

فكرة البرنامج: صَمّم واجهة مستخدم رسومية تعرض التاريخ والوقت الحالي بحيث يمكن تحديثهما باستمرار عند الضغط على زر البدء Start وإيقاف التحديث بالضغط على الزر Stop مع إمكانية إنهاء البرنامج بالضغط على زر خروج، وذلك من خلال الخطوات التالية.

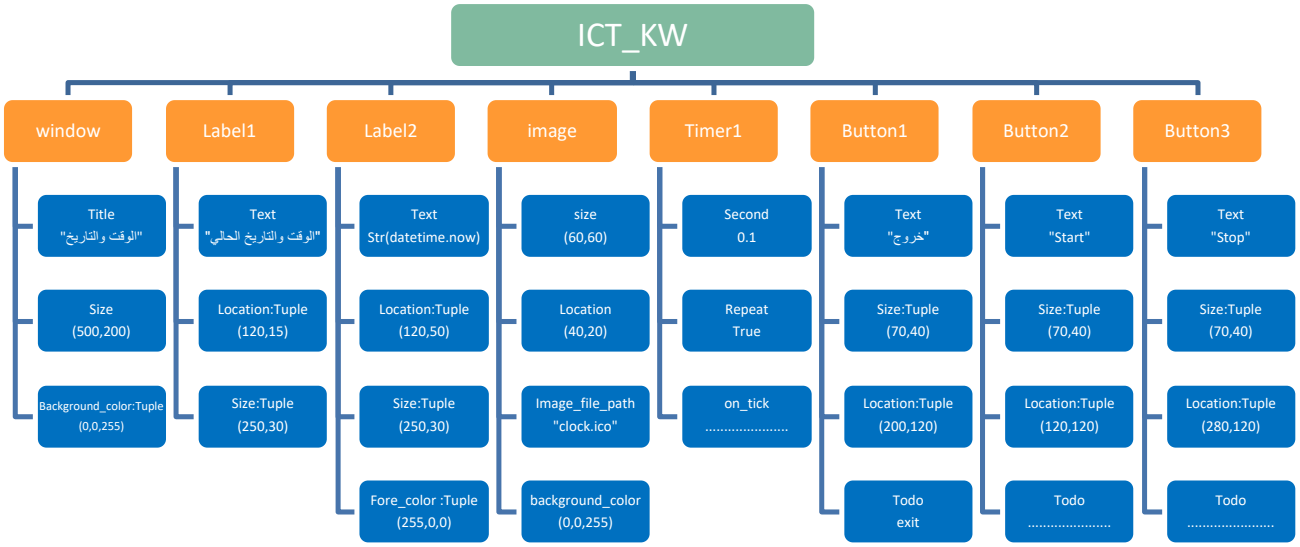
- شغل الملف التنفيذي Time&date.exe وعاين المنتج النهائي.
- أنشئ ملف جديد باسم Time_Date_Timer.py في المشروع First_GUI .
- استدع المكتبات التالية: ICT_KW و datetime.

- أعلن عن دالة باسم (edit_time) خالية من المعاملات، تحدد الوقت والتاريخ في العنوان Label2 مستعينًا بالدوال المكتبية datetime:

الخطوة الأولى: تصميم الواجهة الرسومية:

تابع - ورقة عمل 1:

صَمّم واجهة المستخدم الرسومية السابقة بكتابة التعليمات البرمجية الخاصة بإنشاء عناصرها مستعيناً بالمخطط التالي:



الخطوة الثانية: الاستجابة للأحداث:

- اكتب التعلميات البرمجية المناسبة لتحديث الوقت والتاريخ باستمرار عند الضغط على زر Start والتوقف عند الضغط على زر Stop، ثم اكمل الناقص في المخطط السابق.

توظيف الدوال في الواجهة الرسومية

اكتب التعليمات البرمجية

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

13. .....

14. .....

15. .....

16. .....

17. .....

18. .....

19. .....

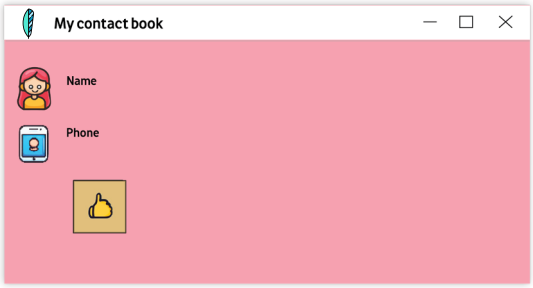
20. .....

ورقة عمل 2:

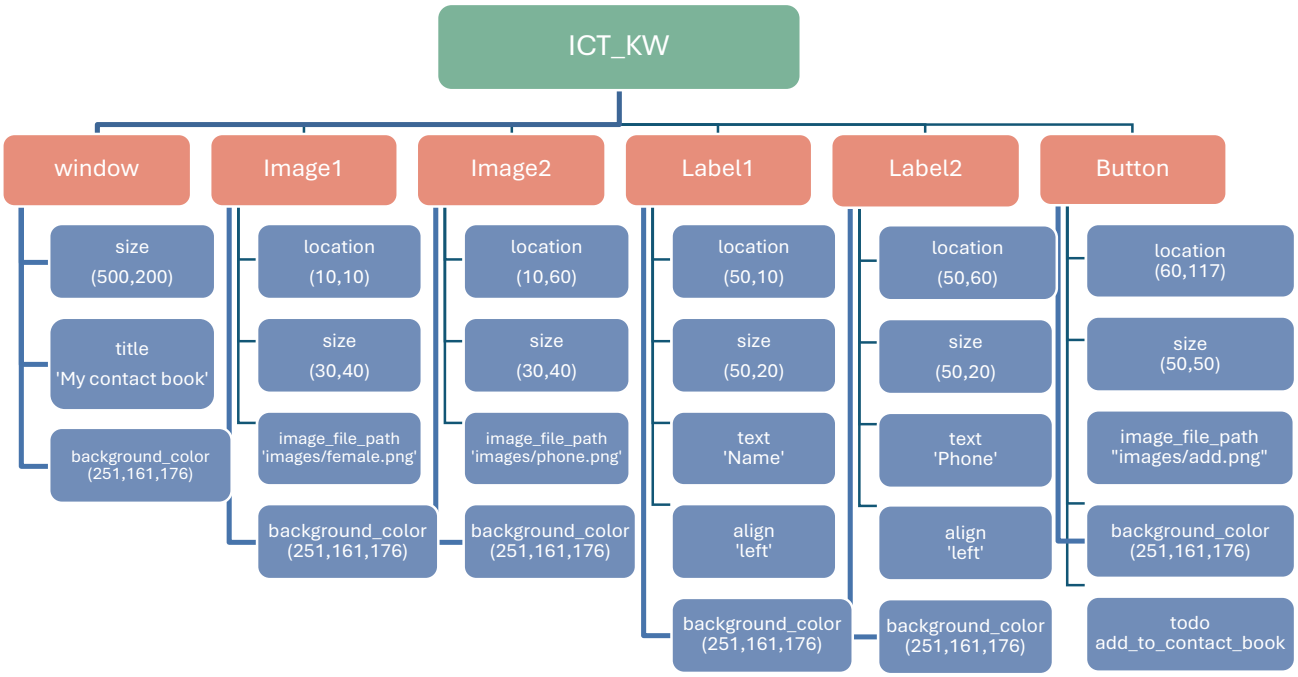
اسم البرنامج: برنامج «My Contact Book»

افتح المشروع «My Contact Book 2» ثم افتح ملف main.py، ونفذ 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:



توظيف الدوال في الواجهة الرسومية

اكتب التعليمات البرمجية

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

13. .....

14. .....

15. .....

16. .....

17. .....

18. .....

19. .....

20. .....

برمجة Python

الوحدة الأولى

القواميـــس
Dictionaries

1 مدخل إلى القواميس

2 التعامل مع القواميس

3 تطبيقات إثرائية

القواميـــس Dictionaries

نتاجـــات التعـلم

- التعرف على مفهوم القواميس، وأهميتها.
- القدرة على إنشاء القواميس، والتعرف على كيفية إضافة أو تعديل العناصر في القاموس عن طريق تعيين قيمة جديدة لمفتاح معين.
- الوصول إلى عناصر القاموس باستخدام المفاتيح للوصول إلى القيم المخزنة داخل القاموس.
- أنواع البيانات القابلة للاستخدام كمفاتيح وقيم: معرفة أن المفاتيح يجب أن تكون فريدة وغير قابلة للتغيير، وأن القيم يمكن أن تكون من أي نوع من البيانات.
- استخدام الدوال المدمجة: استخدام الدوال المدمجة للعمل مع القواميس.
- حل مسائل أكثر تعقيدًا: حل مسائل أكثر تعقيدًا تتطلب استخدام القواميس جنبًا إلى جنب مع هياكل بيانات أخرى.
- تطبيق القواميس في مشاريع حقيقية: تطبيق ما تم تعلمه من القواميس في مشاريع برمجية حقيقية مثل تخزين بيانات الطلاب، إدارة القوائم، وتحليل البيانات.

القواميس Dictionaries

مدخل إلى القواميس

تعتبر القواميس أحد أنواع البيانات الغير الأولية Non-Primitive ، والتي يمكن أن تخزن أكثر من قيمة - مثل القوائم List-الصفوف Tuple- المجموعات Sets- وتُمثل بياناتها كأزواج من المفاتيح Keys والقيم Values.

خصائص القواميس

- تنحصر عناصر القواميس بين الأقواس المعقوفة { } Curly Braces.
- كل عنصر يتضمن مفتاح Key، والقيمة الخاصة به Value .
- عناصرها مرتبة Ordered، مما يعني أن عناصرها تحتفظ بترتيبها.
- المفاتيح keys يجب أن تكون فريدة Unique (غير مكررة)، وفي حال التكرار يتم استبدال القيمة القديمة بالقيمة الجديدة.
- المفاتيح Keys غير قابلة للتعديل Immutable، ويكون من نوع (string-integer-set-tuple)، ولا يمكن أن يكون من (القوائم أو القواميس الأخرى).
- القيم Values تُمثل بأيٍ من أنواع البيانات Data Types.

نوع البيانات	قابلة للتغيير	العناصر مرتبة	الفهرسة	العناصر فريدة	أنواع مختلفة من البيانات	الصيغة
Data Type	Mutable	Ordered	Indexing	Unique Elements	Different of Data Types	Syntax
Dictionaries	نعم	نعم	باستخدام المفاتيح Keys	المفاتيح Keys فقط	القيم Values يمكن أن تكون متعددة الأنواع	{"key": "value"}

جدول رقم (١) خصائص القواميس

صيغة القاموس Dictionary Syntax

عند كتابة القاموس يجب أن يتضمن كل عنصر بالقاموس مفتاح Key، وقيمة Value.

```
student = {
    'name': 'Ahmed',
    'age': 25,
    'city': 'Kuwait'
}
```

```
# "name" is the key, "Ahmed" is the value associated with it.
# "age is the key" - 25 is the value associated with it.
# "city" is the key - "Kuwait" is the value associated with it.
```

مفتاح Key

قيمة Value

التعامل مع القواميس Dictionaries

1. إنشاء القاموس Dictionary

يمكن إنشاء القاموس باستخدام:

a. إنشاء القاموس باستخدام الأقواس المعقوفة { }.

مثال1:

```
student = {
    "name": "Ahmed",
    "grade": "Tenth",
    "age": 15,
    "city": "Jahra"
}
```

لاحظ: لاحظ: يتم وضع عناصر القاموس (المفاتيح والقيم) داخل الأقواس { }

إثرائي

b. إنشاء القاموس باستخدام الدالة ( )dict.

مثال2:

```
1  student = dict(name = "Ahmed", grade= "Tenth", age=15, city= "Jahra")
2  |
```

لاحظ: يتم وضع عناصر القاموس (المفاتيح والقيم) داخل أقواس الدالة ( ).

2. الوصول إلى العناصر (قيمة العنصر)

يمكن الوصول إلى قيمة العنصر في القاموس باستخدام المفتاح الخاص به.

مثال3: الوصول إلى عمر الطالب (15) باستخدام المفتاح الخاص به (age).

```
1  student = {"name": "Ahmed", "grade": "Tenth", "age": 15, "city": "Jahra"}
2  print(student["age"])
3  |
```

Output:

```
15
```

إثرائي

مثال4: الوصول إلى قيمة العنصر (Ahmed) باستخدام المفتاح الخاص به (name).

```
1  student = dict(name = "Ahmed", grade= "Tenth", age=15, city= "Jahra")
2  print(student["name"])
3  |
```

Output:

```
Ahmed
```

لاحظ

صيغة القاموس باستخدام الأقواس { } في المثال 3، وصيغة القاموس باستخدام الدالة dict في المثال 4 .

3. إضافة وتعديل العناصر

يمكن إضافة عناصر جديدة أو تعديل القيم الموجودة.

a. إضافة عنصر جديد.

مثال5:

```
1    student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
2    print(student)
3    student["Country"] = "Kuwait"
4    print(student)
5
```

Output:

```
{'name': 'Ahmed', 'age': 15, 'city': 'Jahra'}
{'name': 'Ahmed', 'age': 15, 'city': 'Jahra', 'Country': 'Kuwait'}
```

b. تعديل قيمة موجودة.

مثال 6:

```
1    student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
2    print(student)
3    student["name"] = "Salem"
4    print(student)
5
```

Output:

```
{'name': 'Ahmed', 'age': 15, 'city': 'Jahra'}
{'name': 'Salem', 'age': 15, 'city': 'Jahra'}
```

4. حذف العناصر

يمكن حذف عنصر من القاموس باستخدام الأمر (del) أو باستخدام الدالة ( )pop.

a. حذف عنصر باستخدام الأمر del.

مثال7:

```
1    student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
2    print(student)
3    del student["age"]
4    print(student)
5
```

Output:

```
{'name': 'Ahmed', 'age': 15, 'city': 'Jahra'}
{'name': 'Ahmed', 'city': 'Jahra'}
```

يمكن حذف القاموس بأكملة باستخدام الدالة (dictionary name)del .

b. حذف عنصر باستخدام دالة ( )pop.

مثال8:

```
1    student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
2    print(student)
3    student.pop("age")
4    print(student)
5
```

Output:

```
{'name': 'Ahmed', 'age': 15, 'city': 'Jahra'}
{'name': 'Ahmed', 'city': 'Jahra'}
```

مثال9:

```
student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
student_name = student.pop("name")
print("student_name:", student_name)
```

Output:

```
student_name: Ahmed
```

c. حذف العنصر الأخير بالقاموس باستخدام الدالة ( )popitem.

مثال10:

```
student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
print(student)
student.popitem()
print(student)
```

Output:

```
{'name': 'Ahmed', 'age': 15, 'city': 'Jahra'}
{'name': 'Ahmed', 'age': 15}
```

لاحظ

الأمر (pop) يحذف عنصر (مفتاح وقيمته) من القاموس، ويمكن إرجاع قيمة المفتاح المحذوف إلى متغير، بينما الأمر (del) لا يُرجع أي قيمة.

d. مسح جميع عناصر القاموس باستخدام دالة ( )clear.

مثال11:

```
1    student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
2    student.clear( )
3    print(student)
4
```

Output:

```
{ }
```

لاحظ:

الفرق بين استخدام كلًا من التعليمة البرمجية

student. clear( ) # مسح جميع عناصر القاموس

del(student) # حذف القاموس بالكامل من الذاكرة وأي محاولة للوصول إليه سيؤدي إلى خطأ

5. التكرار عبر عناصر القاموس باستخدام دالة ( )item

التكرار عبر العناصر (المفاتيح والقيم)

مثال 12:

```
1    student = {"name": "Ahmed", "age": 15, "city": "Jahra"}
2    for key, value in student.items():
3        print(f'{key}: {value}')
4
```

Output:

```
name: Ahmed
age: 15
city: Jahra
```

إثرائي: مثال 13: إنشاء قاموس، والوصول (لقيمة) العنصر بمعلومية (المفتاح).

طباعة المفاتيح والقيم للقاموس (بيانات درجات الطالب)، وطباعة مجموع الدرجات للمواد الدراسية.

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

Output:

```
NAME : Ahmed Salem
ARABIC : 90
ENGLISH : 85
SCIENCE : 95
COMPUTER : 98
Total degrees = 368
```

يمكن استخدام: for key in dic_student. keys() لأداء نفس المهمة.

إثرائي: مثال 14: إنشاء قاموس متعدد الأبعاد (Nested Dictionary).

تحليل بيانات الطلاب المخزنة في قواميس واستخراج نتائج معينة بناءً على درجاتهم في مواد مختلفة.

```
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)
```

طوّر البرنامج: لطباعة أسماء الطلاب وعددهم الحاصلين على درجة أكبر من 95 في مادة الحاسوب.

ورقة عمل: 1

اسم البرنامج: عن دولة الكويت

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

المطلوب:

- استدعاء الملف dictionary_Kw.py من مجلد أوراق العمل.

```
# GCC Countries Dictionary
gcc_countries = {
        "Saudi Arabia": "Riyadh",
        "Kuwait": "Kuwait City",
        "United Arab Emirates": "Abu Dhabi",
        "Qatar": "Doha",
        "Oman": "Muscat",
        "Bahrain": "Manama",
        "Yemen": "Sana' a"
}
# Kuwait info Dictionary
my_country = {
        "Country": "Kuwait",
        "Capital": "Kuwait",
        "population": 8000000
}
# Enter new value to currency key


# update key "population" value


# Print element of my_country dictionary
print(f"Info About Kuwait\n{"*"*40}") # A type of escape character that will create a new line
for x, y in my_country.items():
  print(x, y)

# Delete an element from GCC_countries


# Print element of my_country dictionary
```

- دراسة التعليمات البرمجية، ثم تنفيذ التالي:

a- على القاموس (my_country):

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

b- على القاموس (gcc_countries):

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

- مخرجات البرنامج المتوقعة.

```
Enter Kuwait Currency: Kiwaiti Dinar
Info About Kuwait
**************************************************************
Country Kuwait
Capital Kuwait
population 5000000
currency Kiwaiti Dinar
The Gulf Cooperation Council Countries
**************************************************************
Saudi Arabia Riyadh
Kuwait Kuwait City
United Arab Emirates Abu Dhabi
Qatar Doha
Oman Muscat
Bahrain Manama
```

تابع : ورقة عمل: 1

تطوير البرنامج

المطلوب:

- طور البرنامج ليشمل جميع دول مجلس التعاون الخليجي من حيث عدد السكان.

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

13. .....

14. .....

15. .....

16. .....

إثرائي: ورقة عمل 2

اسم البرنامج: درجات الطلاب

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

المطلوب:

- استدعاء الملف dictionary_students.py من مجلد أوراق العمل.

```
# create dictionaries
student1 = {"name": "ali", "result":"success", "total_degrees": 650}
student2 = {"name": "khalid", "result": "success", "total_degrees": 700}
student3 = {"name": "ahmed", "result": "failed", "total_degrees": 250}

# list = some of the dictionaries
all_students = [student1, student2, student3]

# Create new Student's Dictionary


# Add Student's Dictionary to list (all_student)


# Print keys and values
for i in all_students:
    for x, y in i.items():
        print(f"{x:<20}{y}")
    print("="*40)
```

- أضف التعليمات البرمجية اللازمة ليعمل البرنامج بشكل صحيح.
- استنتج وظيفة التعليمة البرمجية print(f"{x:<20}{y}") من ناتج تنفيذ البرنامج.

.....

.....

إثرائي: تابع: ورقة عمل 2

- ادرس التعليمات البرمجية السابقة، وناقشها مع معلمك.
- تطوير البرنامج بتنفيذ التالي:
  - إنشاء قاموس باسم student4 (يحتوي بيانات متعلم)
  - إضافة القاموس إلى القائمة all_students باستخدام الدالة المناسبة.

- مخرجات البرنامج المتوقعة.

```
Enter new name: Fahed
Enter student result(success/failed): Succes
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                Fahed
result              Success
total_degrees       500.0
====================================
```

إثرائي: تابع: ورقة عمل 2

تطوير البرنامج

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

13. .....

14. .....

15. .....

16. .....

17. .....

تطبيقات إثرائية

تعزيز مهارات متقدمة - برنامج Python

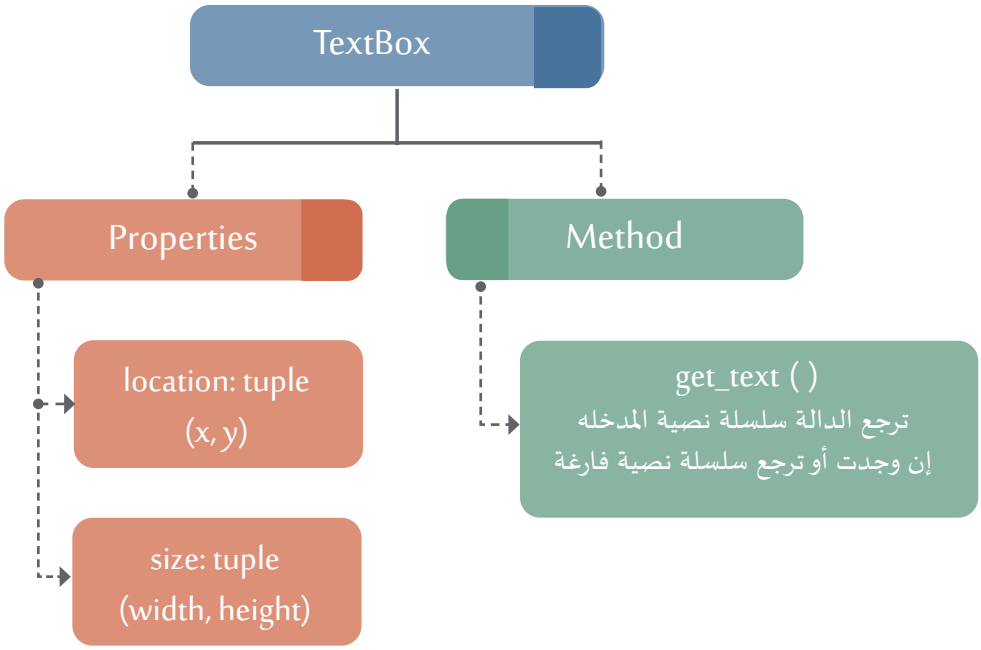
توظيف القواميس في واجهة المستخدم الرسومية

سبق وأن تعرفنا على مجموعة من العناصر التي تمكننا من عرض مجموعة من البيانات سواءً سلاسل نصية باستخدام أداة العنوان Label، أو الصور باستخدام الأداة Image، أو تنفيذ أوامر برمجية باستخدام أداة الزر Button. ولكن ماذا لو أردنا استقبال بيانات نصية من المستخدم؟ سابقاً كنا نستخدم الدالة Input ولكن ماهي الأداة الرسومية المرادفة لها؟

صندوق النصوص TextBox

عنصر رسومي مرئي يُمكن المستخدم من كتابة سلاسل نصية من خلال الواجهة الرسومية، واستدعائها لمعالجتها باستخدام أحد دوالها الخاصة ()get_text.

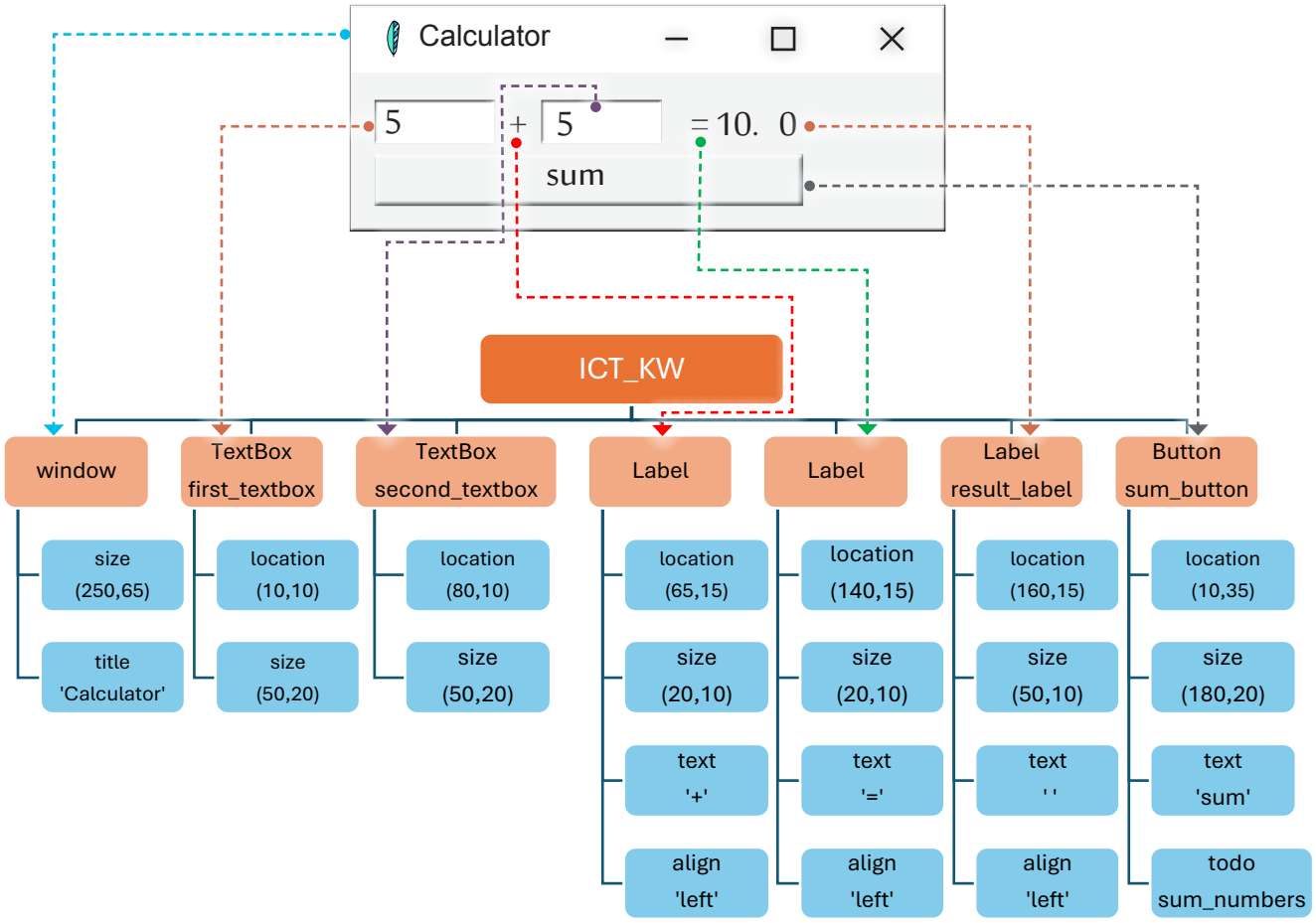
الصيغة Syntax: TextBox_name(Optional) = ICT_KW.new_TextBox(Properies)



مثال 1: تطوير برنامج آلة حاسبة بسيطة لجمع عددين

الخطوة الأولى: تصميم الواجهة الرسومية:

تصميم واجهة مستخدم رسومية تطلب من المستخدم إدخال رقمين ثم جمعهما، وإظهارهما بعد الضغط على زر محدد من أجل إظهار ناتج الجمع للمستخدم:



الخطوة الثانية: تصميم الاستجابة للأحداث:

الإعلان عن الدالة sum_numbers، وربطها بالأداة sum_button، واستدعائها عند الضغط عليها:

```
def sum_numbers ():
```

استدعاء المتغيرات من خارج نطاق الدالة
first_textbox, second_textbox, result_lable

```
global first_textbox, second_textbox, result_label
```

نعلن عن المتغير sum_result، ونعين له قيمة ناتج جمع العددين

```
a = float(first_textbox. get_text())
b = float(second_textbox. get_text())
sum_result = a + b
```

نعرض النتيجة من خلال أداة النص lable_result

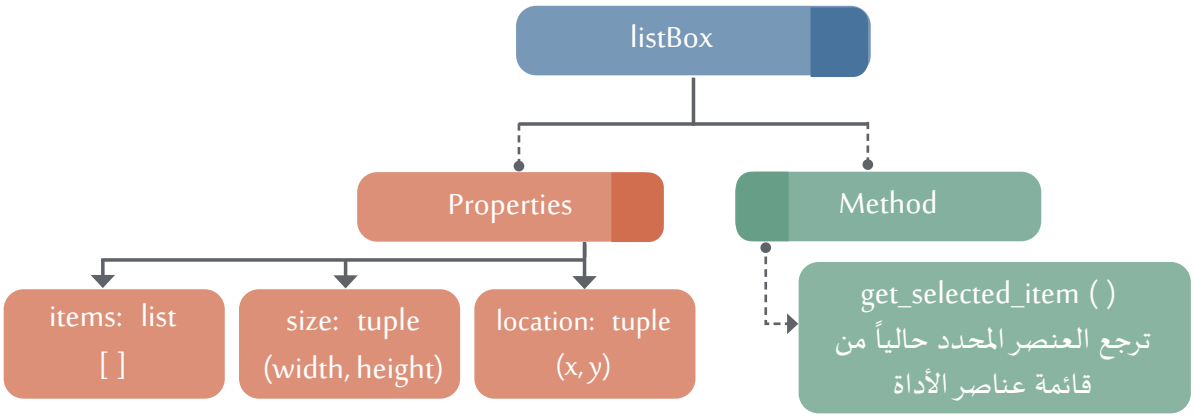
```
result_label. edit(text=str(sum_result))
```

```
import ICT_KW
def sum_numbers():
    global first_textbox, second_textbox, result_label
    a = float(first_textbox.get_text())
    b = float(second_textbox.get_text())
    sum_result = a + b
    result_label.edit(text=str(sum_result))
ICT_KW.window (size = (250, 65), title = 'Calculator')
first_textbox = ICT_KW.new_TextBox(location=(10, 10), size=(50, 20))
second_textbox = ICT_KW.new_TextBox(location=(100, 10), size=(50, 20))
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=(160, 15), size=(50, 10), text='', align='left')
sum_button = ICT_KW.new_Button(location=(10, 35), size=(180, 20), text='sum', todo = sum_numbers)
ICT_KW.run()
```

أداة صندوق القوائم ListBox

أداة رسومية مرئية تفاعلية تعرض مجموعة من البيانات من نوع قائمة list يمكن أن تنفذ أوامر عند الضغط على أحد عناصرها.

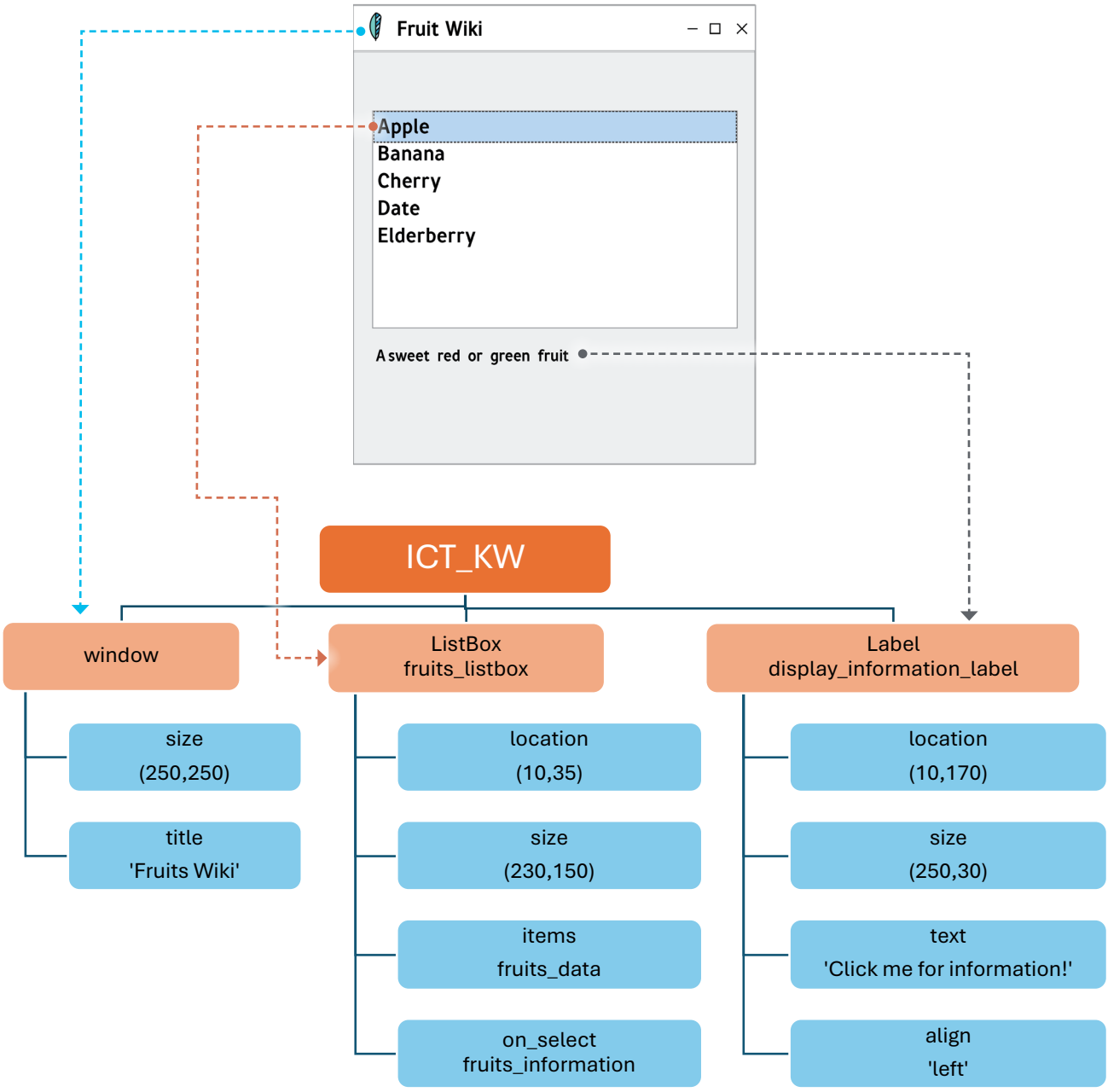
الصيغة Syntax: ListBox_name(Optional) = ICT_KW.new_ListBox(Properies)



مثال 2:

صمم واجهة مستخدم رسومية لبرنامج يعرض معلومات عن الفواكه فور اختيار أحدها من القائمة.

الخطوة الأولى: التخطيط للبرنامج:



الخطوة الثانية: كتابة البرنامج:

أعلن عن قاموس باسم fruits_data وأضف له البيانات التالية:

المفتاح Key	القيمة Value
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.

أعلن عن الدالة fruits_infromation واربطها بالأداة fruits_listbox ليتم استدعاء أحد عناصرها بالضغط عليها:

```
def fruits_information( ):
```

استدعاء المتغيرات fruits_listbox, fruits_data,display_information_lable من خارج نطاق الدالة

```
global fruiys_listbox, fruits_data, display_information_label
```

تعديل الخصائص المناسبة في أداة صندوق النصوص display_ information_lable
ليعرض معلومات عن الفاكهة من المتغير fruits_data, بناءً على العنصر المحدد من الأداة

```
display_information_label. edit(text=fruits_data[fruits_listbox.
get_selected_item( )])
```

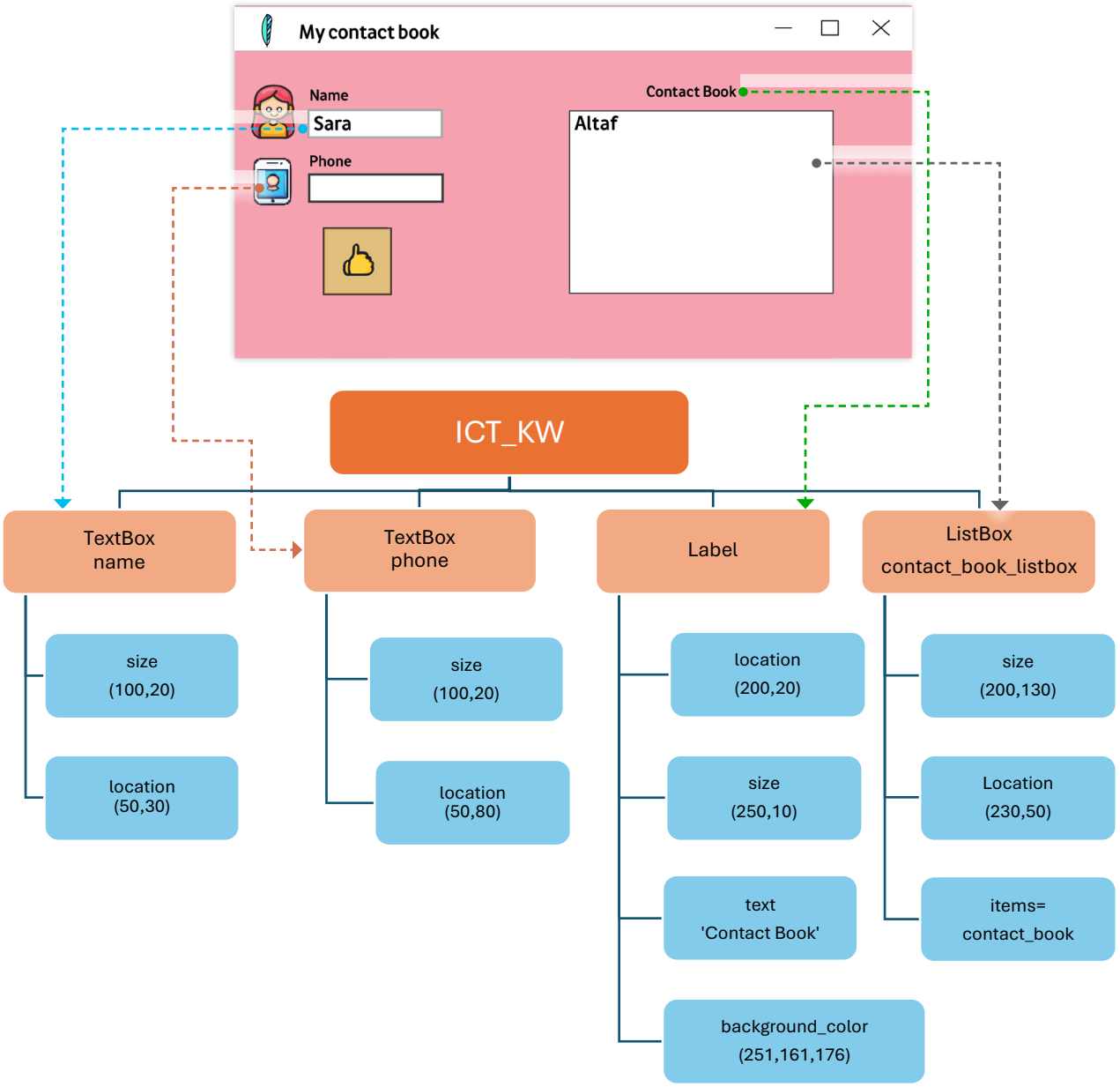
```python
import ICT_KW
fruits_data = {
    'Apple': "A sweet red or green fruit.",
    'Banana': "A long yellow fruit that's soft inside.",
    'Cherry': "A small, round, red fruit with a pit.",
    'Date': "A sweet brown fruit from the date palm tree.",
    'Elderberry': "A small,dark purple often used in syrups."
}
def fruits_information():
    fruits_listbox, fruits_data, display_information_label
    display_information_label.edit(text=fruits_data[fruits_listbox.get_selected_item()])
ICT_KW.window(size=(250, 250), title='Fruits Wiki')
fruits_listbox = ICT_KW.new_ListBox(location=(10, 35), size=(230, 150), items=fruits_data, on_select=fruits_information)
display_information_label = ICT_KW.new_Label(location=(10, 170), size=(250, 30), text='Click me for information')
ICT_KW.run()
```

ورقة عمل (1) اسم البرنامج: تصميم برنامج «My Contact Book»

افتح المشروع «My Contact Book 3» ثم افتح ملف main.py، ونفذ ما يلي:

- تصميم الواجهة الرسومية:

اكتب التعليمات المناسبة لتطوير الواجهة الرسومية لتكون على النحو التالي:



تابع : ورقة عمل (I) اسم البرنامج: تصميم برنامج «My Contact BooK»

- أعلن عن قاموس فارغ باسم «contact_book».
- في الدالة «add_to_contact_book» اكتب التعليمات البرمجية اللازمة لإضافة الاسم ورقم الهـــاتف المدخلة في أداتي «name» و «phone» للقاموس «contact_book» ثم اعرضه في أداء صندوق القائمة «contact_book_listbox»

اكتب التعليمات البرمجية

01. .....

02. .....

03. .....

04. .....

05. .....

06. .....

07. .....

08. .....

09. .....

10. .....

11. .....

12. .....

13. .....

14. .....

15. .....

16. .....

17. .....

18. .....

19. .....

20. .....

تطبيق تصميم معرض للصور.

متطلبات تصميم التطبيق

- استخدام المكتبة الرسومية ICT_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.
- استعراض المنتج النهائي بتشغيل الملف التنفيذي "Image Viewer.exe".

إنشاء التطبيق

إنشاء مشروع جديد باستخدام برنامج PyCharm باسم «Image Viewer».

إنشاء ملف project.py.

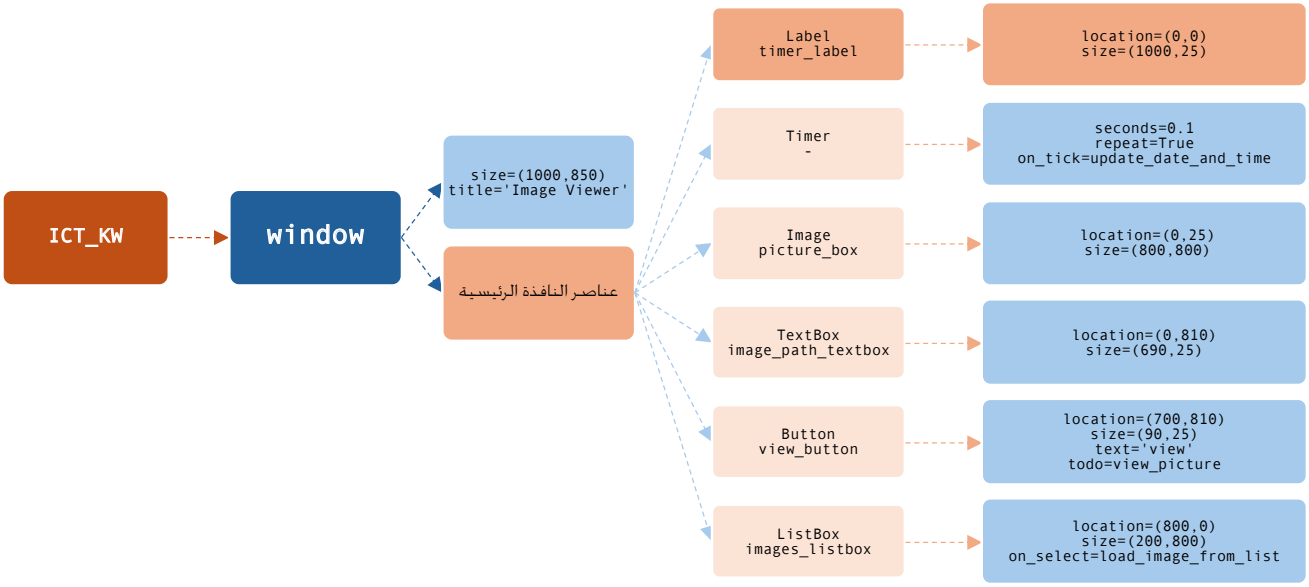
استيراد التالي إلى المشروع:

- المكتبة الرسومية ICT_KW.
- مكتبة الوقت والتاريخ datetime.

```
import ICT_KW
from datetime import datetime
```

الخطوة الأولى: تصميم الواجهة الرسومية

مستعينا بالمخطط التالي، أنشئ واجهة رسومية



اكتب التعليمة البرمجية لإنشاء النافذة الرئيسية:

اكتب التعليمة البرمجية لإنشاء عنوان timer_label:

اكتب التعليمة البرمجية لإنشاء مؤقت Timer:

اكتب التعليمة البرمجية لإنشاء صندوق الصور picture_box:

اكتب التعليمة البرمجية لإنشاء صندوق النصوص image_path_textbox:

اكتب التعليمة البرمجية لإنشاء زر view_button:

اكتب التعليمة البرمجية لإنشاء صندوق القوائم images_listbox:

الخطوة الثانية: تصميم الاستجابة للأحداث

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

```
def update_date_and_time)(:
```

أعلن عن قاموس فارغ image_dict لتخزين اسم ومسار الصور:

```
#key=image name , value=image path
```

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)(:
```

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

```
def load_image_from_list)(   :
```

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

تصدير البرنامج.

الوحدة الثانية

الذكاء الاصطناعي

Artificial Intelligence

1. مدخل إلى الذكاء الاصطناعي
2. الذكاء الاصطناعي التوليدي: الإبداع والابتكار في إنشاء المحتوى.
3. أدوات الذكاء الاصطناعي، ومجالات استخداماتها.
4. المبادئ الأخلاقية في التعامل مع الذكاء الاصطناعي.

الذكاء الاصطناعي

Artificial Intelligence

نتاجـــات التعـلم

1. التعرف على المفاهيم الأساسية:
   - تعريف الذكاء الاصطناعي وفروعه (مثل التعلم الآلي، التعلم العميق، وغيرها).
   - معرفة الفرق بين الذكاء الاصطناعي، التعلم الآلي، والتعلم العميق.

2. قدرة الذكاء الاصطناعي التوليدي ومزاياه:
   - استخدام أدوات وتقنيات الذكاء الاصطناعي.
   - تنفيذ خوارزميات التعلم الآلي الأساسية.

3. تعرف التحديات التي تواجه الذكاء الاصطناعي التوليدي.
4. فهم تطبيقات الذكاء الاصطناعي واتجاهاته.
5. الوصول إلى أدوات الذكاء الاصطناعي التوليدي ومجالات استخدامه، وفوائده، والتحديات التي تواجهه.

6. التطبيقات العملية:
   - تطبيق تقنيات الذكاء الاصطناعي في مجالات مختلفة مثل الرعاية الصحية، الأعمال، والتمويل.
   - حل مشاكل حقيقية باستخدام حلول الذكاء الاصطناعي.

7. أخلاقيات الذكاء الاصطناعي:
   - فهم القضايا الأخلاقية المرتبطة بالذكاء الاصطناعي، مثل التحيز والخصوصية.
   - التفكير في تأثيرات الذكاء الاصطناعي على المجتمع.

مدخل إلى الذكاء الاصطناعي

الذكاء الاصطناعي (AI) هو مجال من مجالات علوم الحاسوب يسعى إلى تطوير أنظمة وبرامج تستطيع محاكاة القدرات البشرية في التفكير والتعلم واتخاذ القرارات.

في ضوء أهمية هذا المجال، أصبح توظيف الذكاء الاصطناعي في قطاع التقنيات الحاسوبية وتطبيقه في مختلف جوانب الحياة، ويُعد من القضايا الأساسية التي يوليها الخبراء والممارسون التقنيون اهتماماً بالغاً، حيث يتطلب ذلك فهمًا شاملاً لمفهوم الذكاء الاصطناعي، متابعة تاريخ تطوره، بالإضافة إلى التعرف على أدواته وكيفية استخدامها بفعالية.

ونتيجة لهذا التطور التكنولوجي الهائل برز الذكاء الاصطناعي كأحد أهم المجالات التي تشكل مستقبل البشرية، وفي هذا السياق، ظهرت شخصيات رائدة في هذا المجال ولعبت دورًا محوريًا في دفع عجلة الابتكار والتقدم، ومن بين هؤلاء الرواد، برز الدكتور عبد الله محمد عبد الكريم المطوع كأحد الأسماء البارزة في مجال الذكاء الاصطناعي والروبوت في دولة الكويت ومنطقة الخليج العربي.

نال درجة الدكتوراة في تخصص الذكاء الاصطناعي من جامعة سيراكيوز في نيويورك، الولايات المتحدة الأمريكية، سنة 1999، يشغل حاليًا منصب أستاذًا جامعيًا في قسم هندسة الكمبيوتر بكلية الهندسة والبترول في جامعة الكويت، حيث أسس أول مختبر للروبوت والذكاء الاصطناعي على مستوى الخليج عام 2001، بهدف تطوير المعرفة وتعزيز استخدام الذكاء الاصطناعي والأنظمة الذكية.

وقد شغل الدكتور عبدالله المطوع منصب مستشارًا لمعالي وزير التربية ووزير التعليم العالي والبحث العلمي في مجال الذكاء الاصطناعي، وترأس اللجنة الوزارية المسؤولة عن إدخال موضوعات الذكاء الاصطناعي في جميع المناهج الدراسية لجميع الفصول. كما حصل الدكتور عبد الله على جائزة الشيخ سالم العلي الصباح للتميز الأكاديمي في الذكاء الاصطناعي لعام 2021، يعمل مستشارًا لمنظمة الملست الدولية، وخبيراً دولياً لمنظمة اليونسكو بالأمم المتحدة في مجال أخلاقيات الذكاء الاصطناعي، حيث ساهم في إعداد وثيقة اليونسكو التاريخية لأخلاقيات الذكاء الاصطناعي، التي تهدف إلى وضع معايير أخلاقية لاستخدام التكنولوجيا بشكل مسؤول، و قدم إسهامات كبيرة في تعليم الشباب التقنيات الحديثة، مما ساهم في نشر ثقافة الذكاء الاصطناعي على المستويين المحلي والإقليمي.

وفي هذا السياق ظهرت العديد من التعريفات الأساسية في مجال الذكاء الاصطناعي وهي على النحو التالي:

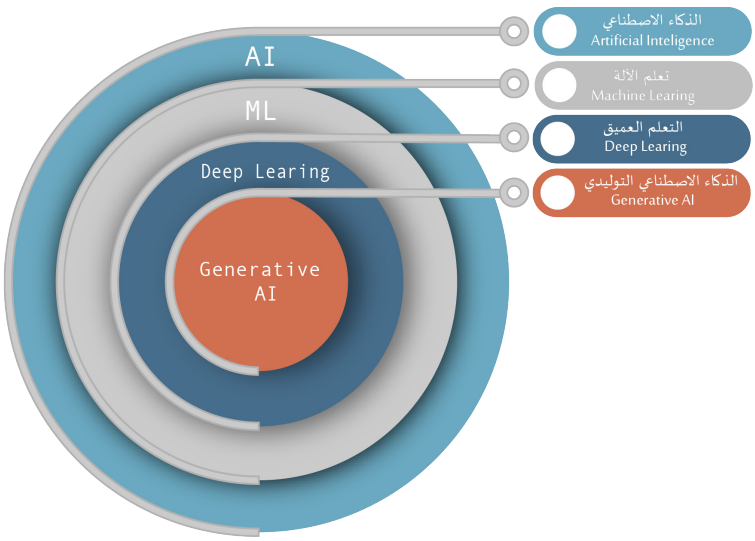
الذكاء الاصطناعي (Artificial Intelligence) أحد أنماط الذكاء المبني في الآلات لتحاكي ذكاء الانسان في اتخاذ القرارات وحسن التصرف.

تعلم الآلة (Machine Learning) فرع من الذكاء الاصطناعي يستخدم عادة لتصنيف البيانات والتنبؤ بالاتجاهات المستقبلية واتخاذ القرارات بناء على البيانات وأمثلة سابقة.

التعلم العميق (Deep Learning) إحدى طرق تعلم الآلة، وهي مستوحاة من بنية الدماغ البشري ووظائفه، أي الربط بين العديد من الخلايا العصبية.

وقد ظهر لاحقًا مفهوم جديد بعنوان الذكاء الاصطناعي التوليدي (Generative Artificial Intelligence)، كأحد أنواع الذكاء الاصطناعي قادر على إنشاء محتوى جديد وأصلي مثل النصوص والصور والفيديو والصوت والكود البرمجي. يعمل باستخدام خوارزميات التعلم الآلي المتقدمة لتحليل كميات هائلة من البيانات وتعلم أنماطها، ثم استخدام هذه المعرفة لإنتاج محتوى جديد يشبه البيانات الأصلية.

كذلك قادر على إنشاء محتوى جديد وأصلي مثل النصوص والصور والفيديو، معتمدًا على تحليل كميات كبيرة من البيانات وتعلم الأنماط منها. الفرق الرئيسي بينهما هو أن الذكاء الاصطناعي التقليدي يركز على تنفيذ مهام محددة، بينما يتميز الذكاء الاصطناعي التوليدي بقدرته على الإبداع وإنتاج محتوى جديد

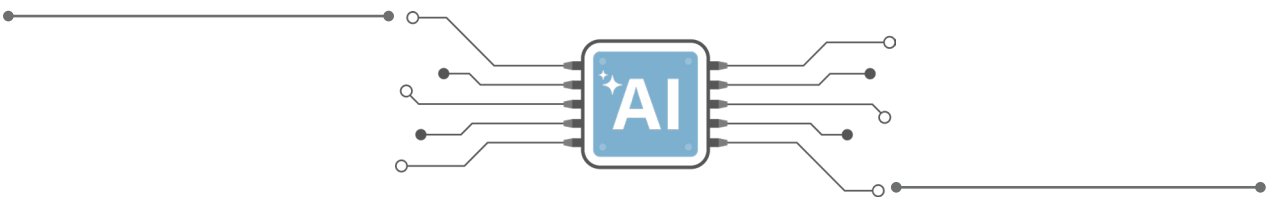


شكل يوضح مجال الذكاء الاصطناعي وعلاقته بمجالات تعلم الآلة، والتعلم العميق.

أولًا: الذكاء الاصطناعي التوليدي - الإبداع والابتكار في إنشاء المحتوى

يتميز الذكاء الاصطناعي التوليدي فيما يلي:

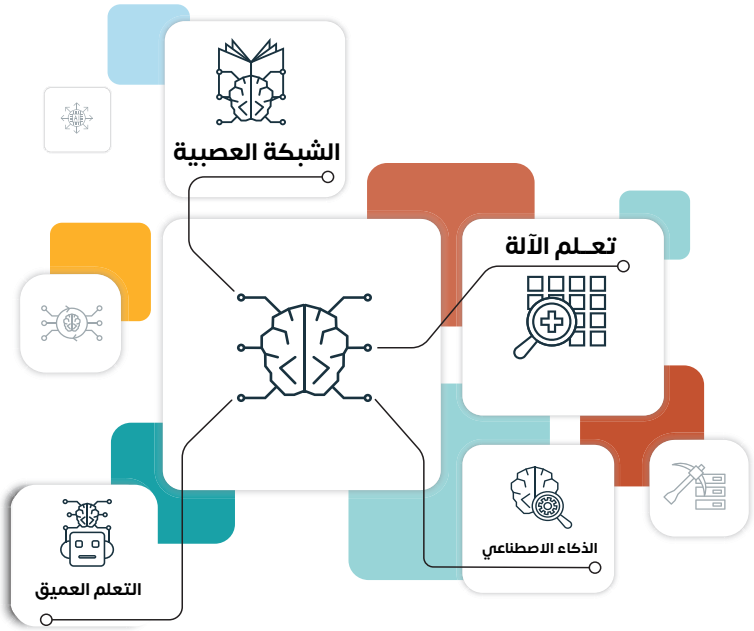
- الذكاء الاصطناعي التقليدي هو نوع من الذكاء الاصطناعي مصمم لأداء مهام محددة بناءً على خوارزميات وقواعد مبرمجة مسبقًا، بينما الذكاء الاصطناعي التوليدي شبكات عصبية عميقة، لتوليد محتوى جديد يشابه البيانات التي تم تدريبه عليها.
- تعلم الآلة يسمح للآلات بالتكيف مع البيانات الجديدة واكتشاف أنماط جديدة. الذكاء الاصطناعي التوليدي يستفيد من هذه التقنية لتوليد محتوى جديد بناءً على البيانات التي تعلم منها.
- الذكاء الاصطناعي اللغوي يمكنه فهم اللغة البشرية والاستجابة لها، بينما التوليدي يمتد لإنشاء أنواع متنوعة من البيانات تتجاوز النصوص لتشمل الصور والفيديوهات.



ويتم تدريب الذكاء الاصطناعي التوليدي باستخدام البيانات التي تم جمعها من صفحات الويب ومحادثات ووسائل التواصل الاجتماعي، وغيرها من الوسائل عبر الإنترنت بحيث يتم توليد المحتوى الجديد من خلال توظيف التحليلات الإحصائية لتوزيع الكلمات أو وحدات البيكسل أو العناصر الأخرى في البيانات، وتحديد الأنماط الشائعة بها وتكراراتها.

وبالرغم من قدرة الذكاء الاصطناعي التوليدي على إنتاج محتوى يبدو جديداً، إلا أنه يفتقر إلى توليد أفكار أو حلول مبتكرة لتحديات العالم الحقيقية، فهو لا يفهم العالم الواقعي، ولا يدرك الأشياء أو العلاقات الاجتماعية التي تكمن وراء اللغة، وبالرغم من قيامه بتوليد الأفكار والمعلومات بطلاقة وسرعة مثيرة للإعجاب، فإنه لا يمكن الوثوق به تمامًا، إلا أنه يفتقر للدقة اللازمة، و غالباً ما تمر أخطاؤه دون ملاحظة، ما لم يكن لدى المستخدم معرفة عميقة بموضوع بحثه.

ويستخدم الذكاء الاصطناعي التوليدي النصي نوعًا من الشبكات العصبية الاصطناعية Artificial Neural Networks، تسمى نموذج اللغة الكبير LLM (Large Language Model)، ويطلق عليها أيضا المحول التوليدي سابق التدريب A Generative Pretrained Transformer أو GPT ومن هنا جاء GPT في مصطلح ChatGPT



نظرا للتطور الهائل للذكاء الاصطناعي التوليدي ظهرت العديد من التحديات المتعلقة بالذكاء الاصطناعي التوليدي، ومن أهمها:

فجوة القوة الحوسبية
Computing Power Gap

يعتمد الذكاء الاصطناعي التوليدي على كميات ضخمة من البيانات وقوة حوسبة عالية، وهي موارد غير متاحة للجميع. مما يزيد الفجوة بين الدول والمجتمعات التي تملك الوصول إلى هذه الموارد وتلك التي لا تملكها.

التطور السريع
Rapid Development

التطور المستمر لهذه التقنية يفوق قدرة التشريعات والقوانين على مواكبته، مما يُنشئ تحديات قانونية وأخلاقية تتطلب من الباحثين والمعلمين الانتباه إليها، والبحث المستمر والابتكار في مجالات الأمن والأخلاقيات الرقمية.

انتهاك حقوق الملكية
Intellectual Property Violation

المحتوى المُنتج باستخدام الذكاء الاصطناعي قد ينتهك حقوق الملكية الفكرية، مما يستدعي تشريعات وضوابط فعالة تضمن حماية الحقوق الفكرية.

الحد من تنوع الآراء
Limiting Diversity of Opinions

خوارزميات الذكاء الاصطناعي قد تقلل من تنوع الآراء والمحتوى الذي يصل إلى المستخدمين، حيث تعرض بعض الأنظمة محتوىً يعتمد على تفضيلات المستخدم السابقة، وإنشاء حواجز فكرية تزيد من الانقسام بين الأشخاص ذوي الآراء المختلفة، ويقلل من فرص التواصل والتفاهم بينهم.

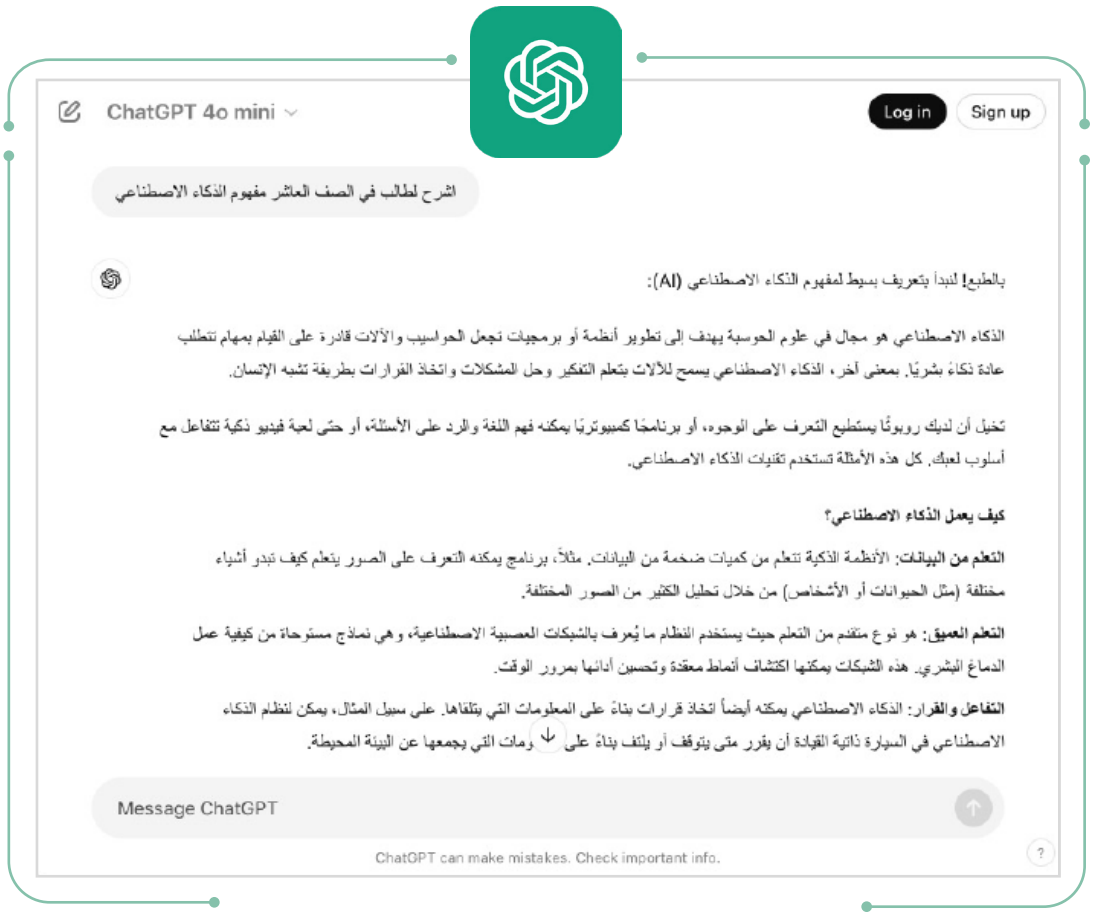
التزييف العميق
Deepfake

يمكن استخدام الذكاء الاصطناعي التوليدي لتزييف الصور والفيديوهات بطرق يصعب كشفها، مما يفتح الباب أمام إساءة الاستخدام مثل نشر المعلومات المضللة.

تناولت تطبيقات الذكاء الاصطناعي عدة اتجاهات وهي:

- توليد نص (Text Generation):

تعمل على إنتاج محتوى جديد باللغة الطبيعية حيث يمكن استخدام النماذج التوليدية لإنشاء نص إبداعي جديد على سبيل المثال يمكن تدريب نموذج لغوي مثل ChatGPT على كميات كبيرة من البيانات النصية ثم تستخدم لإنشاء نص جديد ومتماسك وصحيح نحويًا في مختلف اللغات، وفيما يلي مثال على استخدام ChatGPT للإجابة على سؤال تم طرحه على البرنامج.



شكل يوضح إجابة سؤال تم طرحه على ChatGPT

• توليد الصور (Image Generation):

عملية استخدام النماذج التوليدية في برامج الذكاء الاصطناعي التوليدي مثل شبكات تنافسية توليدية GANs (Generative Adversarial Networks)، وتقنية نماذج الانتشار (Diffusion Models) لإنشاء صور جديدة خياليه تشبه بصريا صور لأشخاص في العالم الحقيقي، ويمكن معاينة تلك الصور من خلال الموقع الإلكتروني https: //thispersondoesnotexist.com

شكل يوضح صور غير حقيقية تم انشاؤها باستخدام تقنية الذكاء الاصطناعي التوليدي

• توليد الفيديو (Video Generation):

عملية استخدام النماذج التوليدية لإنشاء مقاطع فيديو جديدة من وصف نصي مثال ذلك نموذج Dreamix من شركة جوجل حيث يقوم بتحليل فيديو بواسطة النص وتغيير محتوى الفيديو وفقًا للنص المُدخل من قبل المستخدم، ويستخدم برنامج Dreamix تقنية الانتشار العكسي Inverse Diffusion لإعادة بناء الفيديو بطريقة متسقة زمنياً والمحافظة على اللون والوضعية وحجم الأشياء وزاوية الكاميرا.

• توليد الصوت (Voice Generation):

نماذج توليدية تم تدريبها على تسجيلات صوتية في الاصوات المختلفة وبأعداد ضخمة ويمكن من خلالها تحويل النص إلى كلام.

ثانيًا: أدوات الذكاء الاصطناعي التوليدي، ومجالات استخدامها

أدوات الذكاء الاصطناعي وكيفية التعامل معها:

تُعتبر أدوات الذكاء الاصطناعي ثورة في عالم التكنولوجيا، فهي تُمكّننا من القيام بمهام معقدة بسرعة وفعالية، وتفتح آفاقًا جديدة للإبداع والابتكار.

تمـرين :

كيف يمكننا التعامل مع هذه الأدوات بكفاءة والاستفادة منها؟

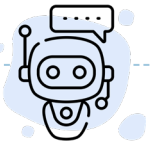
ما هي أدوات الذكاء الاصطناعي؟

هي برامج وتطبيقات تعتمد على خوارزميات الذكاء الاصطناعي لتقليد القدرات الذهنية البشرية، مثل التعلم والاستدلال وحل المشكلات. تتراوح هذه الأدوات من برامج بسيطة تقوم بمهام محددة، إلى أنظمة معقدة قادرة على التعلم والتطور بمرور الوقت.

أنواع أدوات الذكاء الاصطناعي:

1. أدوات معالجة اللغة الطبيعية (NLP): تستخدم لفهم اللغة البشرية وتوليدها، مثل الترجمة الآلية، وتحليل المشاعر، وتوليد النصوص، وغيرها.
2. أدوات التعلم الآلي: تستخدم لتحليل البيانات الكبيرة، اكتشاف الأنماط، التنبؤ، التصنيف، واكتشاف الاحتيال، وغيرها.
3. أدوات الرؤية الحاسوبية: تستخدم لتحليل الصور والفيديوهات، وتستخدم في التعرف على الوجوه، والكشف عن الأجسام، وتوليد الصور، وغيرها.
4. أدوات الروبوتات: تستخدم للتحكم في الروبوتات وتوجيهها، وتستخدم في التصنيع، والخدمات اللوجستية، والرعاية الصحية، وغيرها.

التعامل مع أدوات الذكاء الاصطناعي:



1. فهم الأساسيات: من المهم فهم المبادئ الأساسية للذكاء الاصطناعي وكيف تعمل هذه الأدوات.
2. اختيار الأداة المناسبة: يجب اختيار الأداة التي تناسب المهام المراد إنجازها.
3. تحديد الأهداف: تحديد الأهداف بوضوح قبل البدء في استخدام أي أداة.
4. جمع البيانات: جمع البيانات ذات الجودة العالية لتدريب النماذج.
5. التدريب والتجربة: تدريب النماذج وتجربتها لتحسين الأداء.
6. التقييم المستمر: تقييم أداء النماذج بشكل مستمر وإجراء التعديلات اللازمة.
7. الأمان والخصوصية: الحرص على حماية البيانات والخصوصية عند استخدام أدوات الذكاء الاصطناعي.
8. البقاء على اطلاع: مواكبة التطورات المستمرة في مجال الذكاء الاصطناعي.

أمثلة على أدوات الذكاء الاصطناعي التوليدي:

1. ChatGPT: أداة تستخدم لإنشاء وتوليد النصوص والترجمة.
2. Midjourney: أداة تستخدم لإنشاء الصور الفنية.
3. DALL-E 2: أداة لإنشاء الصور الواقعية والفنية.
4. Google Cloud AI: مجموعة من أدوات الذكاء الاصطناعي المقدمة من Google.
5. Amazon Sage Maker: منصة لبناء وتدريب ونشر نماذج التعلم الآلي.

6. Microsoft Copilot: نموذج لغوي يعمل بنظام الذكاء الاصطناعي على إنجاز المهام بسهولة ويسر ، تُمكن من كتابة المحادثة نصيًا أو عبر الإدخال الصوتي المباشر حيث انه يعمل على تجميع المعلومات من الويب وتقديم الدعم وإكمال المهام وغير ذلك الكثير، ويمكن الوصول لهذه الاداة من خلال منصة مايكروسوفت تيمز بحساب المستخدم الشخصي.

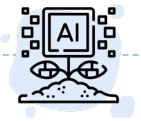
7. Google Gemini: نموذج لغوي كبير طورته غوغل قادر على فهم وتوليد النص والرمز والصوت والصورة والفيديو، وهي مصممة لتكون متعددة الوسائط ومرنة، وقادرة على العمل على أجهزة مختلفة، ويعتبر أحد أكثر نماذج الذكاء الاصطناعي تقدما من جوجل.

8. Internet of Things IoT: ويقصد بها إنترنت الأشياء للتحكم في الأجهزة، كما يتيح أيضًا جمع بيانات عن البيئة المحيطة، وبناء قرارات تخدم التفاعل الذكي، ويُقصد به الجيل الجديد من شبكة الإنترنت الذي يتيح التفاهم بين الأجهزة المترابطة مع بعضها البعض عبر بروتوكول الإنترنت.

وتشمل هذه الأجهزة الأدوات والمستشعرات والحساسات وأدوات الذكاء الاصطناعي المختلفة وغيرها، ويتخطى هذا التعريف المفهوم التقليدي، وهو تواصل الافراد مع الحواسيب والهواتف الذكية عبر شبكة عالمية واحدة ومن خلال بروتوكول الإنترنت التقليدي المعروف، وما يميز إنترنت الأشياء (IoT) أنها تتيح للإنسان التحرر من المكان، أي أن الشخص يستطيع التحكم في الأجهزة بشكل فعال من دون الحاجة إلى التواجد في مكان محدّد للتعامل مع جهاز معين عن قرب وعن بُعد.

فيستطيع المستخدم مثلاً تشغيل محرّك سيارته وجهاز التلفزيون والتكييف والتحكم فيهم من جهاز الحاسوب او الهاتف الذكي.

فوائد استخدام أدوات الذكاء الاصطناعي التوليدي:



- زيادة الإنتاجية: تتيح إنجاز المهام بسرعة ودقة وكذلك توليد كميات كبيرة من المحتوى في وقت قصير.
- تحسين اتخاذ القرارات: تساعد في تحليل البيانات واتخاذ قرارات مناسبة.
- تخصيص الخدمات: تتيح تخصيص المحتوى ليناسب احتياجات المستخدمين الفردية.
- اكتشاف فرص جديدة: تساعد في اكتشاف فرص عمل جديدة في مجالات مهنية عديدة.
- الإبداع: تتميز بقدرتها على توليد أفكار جديدة ومبتكرة.
- توفير التكاليف: تعمل على تنفيذ بعض المهام التي يقوم بها البشر.

التحديات التي تواجه استخدام أدوات الذكاء الاصطناعي:

- التكلفة: قد تكون تكلفة تطوير وتشغيل أدوات الذكاء الاصطناعي عالية.
- الخصوصية: قد تثير مخاوف بشأن خصوصية البيانات.
- الأمان: قد تكون هناك أخطار أمنية مرتبطة باستخدام الذكاء الاصطناعي.
- الوظائف: قد يؤدي انتشار الذكاء الاصطناعي إلى فقدان بعض الوظائف.
- التحيز: تخضع هذه الأدوات للتحيز لصالح البيانات التي تم التدريب عليها وقد لا تعكس الواقع الفعلي.

تمـرين تطـبيقي :

من خلال الدخول الى موقع https: //copilot. microsoft. com.
قم بالبحث عن معلومات عن إحدى الموضوعات التالية:

- الذكاء الاصطناعي التوليدي.
- المعايير الأخلاقية في الذكاء الاصطناعي.

أرسل النص إلى معلمك عبر فريق الفصل في منصة تيمز Teams.

أهم تطبيقات الذكاء الاصطناعي التوليدي:

1 تطبيق توليد النصوص : ومهامه

- كتابة المقالات والمحتوى التسويقي.
- ترجمة اللغات.
- إنشاء قصص شعرية وروائية.
- كتابة كود برمجي.
- إنشاء سيناريوهات للأفلام.

2 تطبيق توليد الصور : ومهامه

- تصميم الجرافيك واللوحات الفنية.
- إنشاء صور واقعية لأشخاص وأشياء غير موجودة.
- تصميم الأزياء والمفروشات.
- تحرير الصور وتعديلها.

3 تطبيق الصوت : ومهامه

- توليد الأصوات البشرية الطبيعية.
- إنشاء الموسيقى والألحان.
- تحسين جودة الصوت.
- دبلجة الأفلام والمسلسلات.

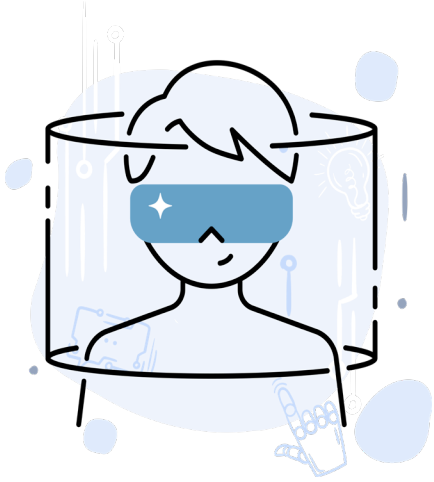
4 تطبيق الفيديو : ومهامه

- إنشاء مقاطع فيديو قصيرة.
- تحرير الفيديوهات وإضافة المؤثرات الخاصة.
- إنشاء عوالم افتراضية.

مجالات استخدام الذكاء الاصطناعي التوليدي في الحياة اليومية:

برنامج «الذكاء الاصطناعي من أجل الأرض» Artificial Intelligence for Earth، الذي أطلقته شركة مايكروسوفت بهدف حماية كوكب الأرض من خلال استخدام علم البيانات وتقنيات الذكاء الاصطناعي التوليدي، لمواجهة التحديات الراهنة والحفاظ على الموارد الطبيعية للقطاعات الأربعة الرئيسة بما يشمل: الزراعة، والمياه، والتغير المناخي، والتنوع البيولوجي.

حيث هدف البرنامج إلى دعم المشاريع المُبتكرة في المجالات المختلفة بما يشمل استخدام الذكاء الاصطناعي التوليدي لإنشاء خرائط للسكان لغاية فهم تأثير تغير المناخ على هجرة البشر



كما لم يغفل الذكاء الاصطناعي التوليدي أيضًا عن خدمة الجانب الإنساني، وذلك من خلال إعداد تطبيق «الذكاء الاصطناعي لمساعدة المكفوفين» AI(Seeing Artificial Intelligence) لخدمة الأشخاص ذوي الاحتياجات الخاصة، يقوم بوصف المشهد المحيط لهم، إضافة إلى تطبيقات أخرى لمساعدة المكفوفين، وغيرها الكثير من التطبيقات التي تهدف إلى خدمة الإنسان.

ونظرًا للتطور التقني الهائل برزت استخدامات للذكاء الاصطناعي التوليدي في مجالات عدة من أهمها:

مجال التعليم:

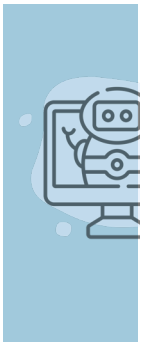
حيث أصبح البحث على الشبكة الإلكترونية جزء لا يتجزأ من التعلم، حيث حلت الأجهزة اللوحية محل الكتب المدرسية التقليدية. حيث من المتوقع أن تنتقل الفصول الدراسية التقليدية إلى فصول افتراضية باستخدام تقنيات الذكاء الاصطناعي، والاعتماد على الروبوتات للقيام بالوظائف الروتينية لتخفيف الضغط على المعلمين، مثل تصحيح الامتحانات وتقييم الواجبات المدرسية .

مجال الطب:

وتمثلت استخداماته في تشخيص الأمراض بدقة عالية من خلال تحليل صور الأشعة، وتحديد المشاكل الصحية التي يُعاني منها المريض، وجمع بيانات عن التاريخ الصحي للمرضى والاحتفاظ بها بسهولة. وأيضاً استشراف المستقبل وتحديد احتمالات التعرض للعدوى والأمراض المختلفة، علاوة على قدرتها على فحص آلاف المرضى في وقت قياسي.

مجال الطيران:

أصبحت العديد من شركات الطيران تعتمد تطبيقات الذكاء الاصطناعي لفحص أمتعة الركاب وتحديد وزنها وتسديد أي رسوم مطلوبة، إضافة إلى المساهمة في إنجاز إجراءات السفر وتسهيل حركة الركاب، من خلال تقنيات التعرف على الوجه، وأجهزة إنهاء عمليات صعود الركاب إلى الطائرة.



مجال النقل والمواصلات:

أصبح التنقل أكثر سهولة ويسر حيث أتاحت تطبيقات الذكاء الاصطناعي التوليدي خدمة طلب السيارات للتوصيل من مكان إلى مكان آخر، وفي أي وقت دون الحاجة إلى الانتظار أو البحث، ومن المتوقع أن ينتشر استخدام السيارات ذاتية القيادة حول العالم، وهي واحدة من أبرز الأمثلة لتعلم الآلة واستخدام تطبيقات الذكاء الاصطناعي التوليدي، حيث تعتمد فكرتها بالأساس على استخدام أجهزة للاستشعار تمدها بالبيانات التي تساعدها على رسم خرائط دقيقة. بالإضافة إلى أنها مزودة بكاميرات تمُكنّها من الانتقال بين الأماكن المختلفة والتعرف على الطرق بشكل ذاتي.

مجال المؤسسات المصرفية والمالية:

اهتمت تطبيقات الذكاء الاصطناعي في هذا المجال بتوفير الخدمات المالية بشكل أكثر أمانًا، وحصول العملاء على الخدمات المطلوبة بسهولة ويسر. حيث نجحت العديد من الشركات الناشئة في تقديم العديد من الخدمات المالية بما يشمل ذلك خدمات الدفع الرقمي، والعملات الرقمية، وتحويل الأموال عبر الحدود، وغيرها من التقنيات المالية الحديثة حيث أصبحت متاحة لجميع فئات المجتمع، وبالأخص تلك التي لا تتعامل مع القطاع المصرفي بشكل مباشر مثل محدودي الدخل، والشباب، والمشاريع متناهية الصغر والصغيرة والمتوسطة الحجم.

ثالثًا: المبادئ الأخلاقية في التعامل مع أنظمة الذكاء الاصطناعي التوليدي

الشفافية

يجب ضمان عدم التحيز أو التمييز ضد أي فئة أثناء تطوير الأنظمة، والتأكد من أن البيانات المستخدمة شاملة وتمثل الجميع بشكل عادل.

الخصوصية

يتعين على الأنظمة الذكية حماية البيانات الشخصية للمستخدمين، وتطبيق تقنيات التشفير والضوابط الأمنية لضمان عدم الوصول غير المصرح به للبيانات، واحترام حقوق المستخدمين في جمع واستخدام بياناتهم.

العدالة وعدم التحيز

تصميم أنظمة الذكاء الاصطناعي عادلة وتعامل جميع الأفراد بالتساوي بغض النظر عن خلفياتهم الاجتماعية أو الثقافية، وأن تكون خالية من التحيزات التي تؤثر على النتائج أو القرارات، وتحليل وتقييم مستمر للخوارزميات لضمان عدم وجود تحيزات ضمنية تؤثر سلباً على المستخدمين.

المُساءلة

يجب أن تكون هناك آلية واضحة لتحميل الأفراد أو الجهات المسؤولة عن الأنظمة الذكية المسؤولية في حالة حدوث أخطاء أو تأثيرات سلبية، هذا يتطلب وجود هياكل تنظيمية وقانونية محددة تضمن الشفافية في تحديد المسؤولين والمحاسبة على تصرفاتهم.

الأمان

من الضروري أن يتم تصميم أنظمة الذكاء الاصطناعي لتكون آمنة وتقلل من المخاطر المحتملة على المستخدمين والمجتمع، هذا يتطلب تطبيق معايير وإجراءات أمان صارمة، مثل: اختبار الأنظمة بشكل دوري للبحث عن نقاط الضعف وتصحيحها، وتطوير بروتوكولات طوارئ للتعامل مع الحوادث الأمنية.

التنمية المستدامة

يهدف هذا المبدأ إلى استخدام الذكاء الاصطناعي لدعم وتحقيق أهداف التنمية المستدامة مثل: تحسين جودة الحياة وتعزيز المساواة الاجتماعية والحفاظ على البيئة. يمكن للذكاء الاصطناعي أن يلعب دوراً هاماً في تطوير حلول مبتكرة للمشكلات البيئية والاجتماعية والاقتصادية.

الوحدة الثالثة

المنتجات الرقمية

مشروع في تقنيات كشف الوجوه

1. مدخل مشروع تقنية التعرف على الوجوه
2. مكتبة OpenCV
3. توظيف الواجهة الرسومية مع مشروع التعرف على الوجوه
4. خطوات إعداد المشروعات

المنتجات الرقمية

نتاجـــات التعـلم

- الإلمام بمفهوم التعرف على الوجوه.
- تثبيت واستخدام مكتبة OpenCV، واستخدام دوالها الأساسية في مشروع Python.
- معالجة الصور وقراءتها من الملفـات، وتحويل الصـــور إلى تدرج الرمادي لتحسين أداء الكشف عنها.
- استخدام نموذج Haar Cascade كأحد خوارزميات التعلم الآلي لكشـف الوجوه.
- التعرف على خطوات إعداد المشروع.
- تنمية مهارات البحث والتعلم الذاتي.
- تنمية مهارات التعاون والعمل بروح الفريق.
- توظيف مهارات الوحدتين في مشروع متكامل يخدم شريحة معينه أو يحل مشكلة محددة.
- تنمية الإبداع والابتكار للمتعلمين.
- اكتساب مهارات الإلقاء وتعزيز الثقة في النفس.

مدخل إلى مشروع تقنية التعرف على الوجوه

أصبحت تقنيات الذكاء الاصطناعي (Artificial Intelligence)، وتعلم الآلة (Machine Learning) من أهم الأدوات المستخدمة في مجموعة واسعة من التطبيقات، تكمن قوة هذه التقنيات في قدرتها على معالجة كميات هائلة من البيانات وتعلم الأنماط المعقدة، مما يجعلها مثالية للعديد من التطبيقات.

تعلم الآلة:

أحد أفرع تطبيقات الذكاء الاصطناعي، والذي يركز على تطوير خوارزميات ونماذج تمكن الأجهزة الرقمية من التعلم من البيانات. بدلاً من برمجتها لتنفيذ مهمة معينة، ويتم تدريب النموذج على مجموعة من البيانات ليتعلم كيفية أداء المهمة بنفسه.

تقنيات كشف الوجوه:

1 خوارزمية الرؤية الحاسوبية Haar Cascade

يعتمد على تحليل الميزات البصرية في الصورة، مثل الحواف والملامح، لاكتشاف الوجوه. يستخدم بشكل واسع بسبب كفاءته وسرعته في الكشف عن الوجوه.

2 الشبكات العصبية العميقة Deep Neural Networks

تستخدم تقنيات تعلم الآلة الأكثر تطورًا لكشف الوجوه. يتم تدريب هذه الشبكات على كميات ضخمة من البيانات (صور الوجوه) لتعلم الأنماط والتفاصيل الدقيقة. توفر هذه التقنية دقة أعلى في الكشف عن الوجوه حتى في ظروف الإضاءة الصعبة أو زوايا التصوير المختلفة.

3 تقنيات التعلم العميق Customize Libraries

هو نوع من تعلم الآلة يستخدم الشبكات العصبية المعقدة التي تحتوي على العديد من الطبقات. من خلال استخدام تقنيات مثل الشبكات التلافيفية (Convolutional Neural Networks) وخصائص مثل التعرف على الملامح، يمكن لهذه النماذج تحقيق دقة عالية في كشف الوجوه والتعرف عليها.

مشروع تقنية التعرف على الوجوه

مكتبة OpenCV:

Open-Source Computer Vision Library مكتبة مفتوحة المصدر تحتوي على العديد من الخوارزميات والأدوات البرمجية لمعالجة الصور والرؤية الحاسوبية، وتدعمها مؤسسة OpenCV، وتستخدم في العديد من المجالات ومنها:

1. معالجة الصور: تشمل عمليات تحسين جودة الصور، وتصحيح الإضاءة، وتصفية العناصر المؤثرة عليها.
2. الرؤية الحاسوبية: من خلال التعرف على الوجوه، تتبع الأجسام في الفيديو، والتعرف على الأنماط الرقمية والنصية.
3. التحليل الهندسي: بتنفيذ التحويلات الهندسية مثل قياس الأبعاد والمسافات داخل الصور.
4. التعلم الآلي: تستخدم في بناء وتصميم نماذج تعلم الآلة لتحليل الصور والفيديوهات.
5. الروبوت: برمجة الروبوت للتعرف على الأشياء والتفاعل مع البيئة المحيطة من خلال الرؤية الحاسوبية.
6. تطبيقات الواقع المعزز: إنشاء تطبيقات تستخدم تقنية الواقع المعزز من خلال دمج العناصر الرقمية مع المشاهد الحقيقية.

استخدام مكتبة OpenCV مع Python للتعرف على وجوه الأشخاص Face Detection:

تستخدم مكتبة (OpenCV) في Python للتعرف على وجوه الأشخاص داخل الصورة المحددة .
من خلال هذا المشروع سنتمكن من تصميم برنامج يعالج صورة محددة، مستعيناً بأحد النماذج مسبقة التدريب (ML) للتعرف على وجوه الأشخاص وتحديدها بمستطيلات زرقاء مختلفة الأبعاد والأماكن في الصورة حسب حجم ومكان كل وجه من خلال التعليمات البرمجية التالية:

Face Detection.py

```python
import cv2
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
image_path = input("Enter the path to the image: ")
image = cv2.imread(image_path)
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray_image, scaleFactor=1.1, minNeighbors=5)
for (x, y, w, h) in faces:
    cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)
cv2.imshow('Detected Faces', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

عند تنفيذ البرنامج وكتابة مسار الصورة (Image Path) يعالج البرنامج الصورة المحددة ويتعرف على الأوجه بها وذلك بتحديدها بمستطيلات زرقاء.

الصورة قبل المعالجة

الصورة بعد المعالجة

تهيئة البرنامج

- إنشاء مشروع جديد باستخدام برنامج PyCharm باسم Face Detection.
- إنشاء ملف Python جديد باسم Face Detection.py، وإضافة التعليمات البرمجية المناسبة.
- تثبيت المكتبات اللازمة باستخدام الأمر pip من خلال أداة Terminal، وكتابة الأمر التالي:

```
pip install opencv-python opencv-python-headless
```

كتابة التعليمات البرمجية

- استيراد مكتبة cv2 (مكتبة فرعية من OpenCV)

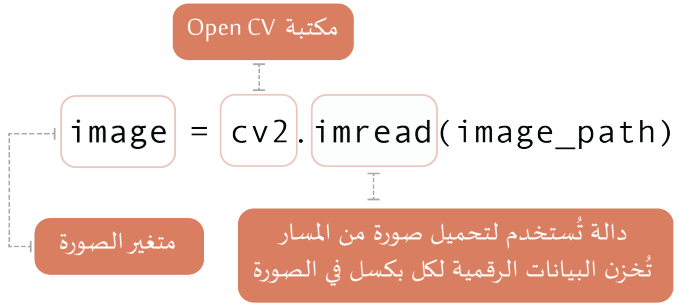
```
import cv2
```

- تحميل نموذج كشف الوجوه:

```
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades+
'haarcascade_frontalface_default.xml')
```

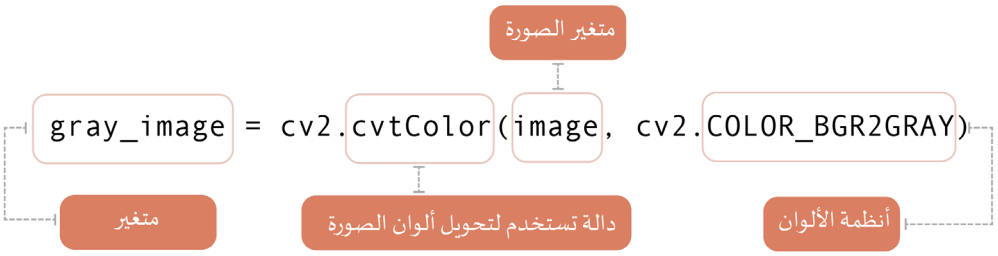
- تحميل الصورة المراد معالجتها من المسار المدخل من المستخدم:

```
image_path = input('Enter the path to the image: ')
image = cv2.imread(image_path)
```



تحويل الصورة من نظام الألوان RGB إلى نظام الألوان Grayscale تدرج الرمادي:

```
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```



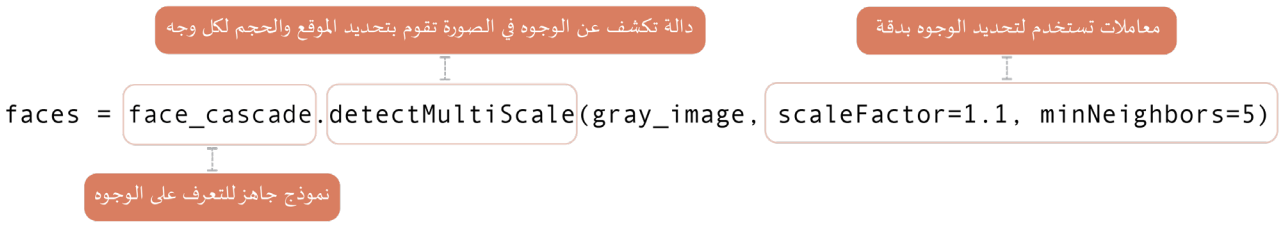
RGB Image

Grayscale Image

لاحظ: تحويل الصورة إلى نظام تدرج الرمادي لزيادة كفاءة وسرعة المعالجة، حيث تصبح القيم بين اللونين الأسود والأبيض (0 إلى 255) بدلًا من تنسيق RGB الثلاثي الألوان الذي يمتد بين (0 إلى 255) لكل لون، مما يقلل من تعقيد المعالجة بشكل كبير.

الكشف عن الوجوه في الصورة المحددة (التي تم تحويلها إلى تدرج الرمادي):

```
faces= face_cascade.detectMultiScale(gray_image,scaleFactor=1.1,minNeighbors=5)
```



لمعلوماتك

`scaleFactor=1.1`

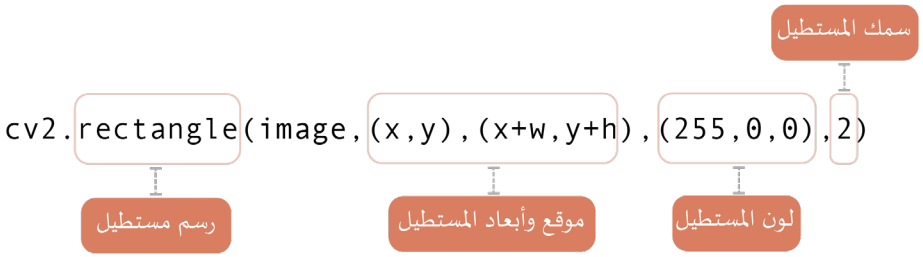
- تحدد نسبة تصغير الصورة في كل خطوة من عملية الكشف.
- قيمة 1.1 تعني أنه في كل خطوة يتم تصغير الصورة بنسبة 10٪.
- كلما كانت القيمة أقل كانت دقة الكشف أعلى، ولكن ذلك يتطلب وقت أطول للمعالجة.

`minNeighbors=5`

- تحدد عدد المناطق المجاورة للمنطقة التي يتم معالجتها حاليًا؛ ليتم اعتبار الكائن (الوجه) ككائن فعلي.
- قيمة 5 تعني أن الكائن المكتشف لديه على الأقل 5 مناطق مجاورة تم التعرف عليها أيضًا كوجوه لتأكيد عملية الكشف.
- كلما زادت القيمة زادت دقة الكشف، لكن قد يتخطى بعض الكائنات.

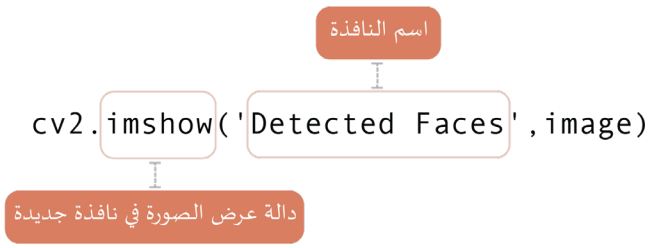
رسم مستطيلات حول الوجوه التي تم اكتشافها في الصورة:

```
for (x,y,w,h) in faces:
cv2.rectangle(image,(x,y),(x+w,y+h),(255,0,0),2)
```



عرض الصورة بعد تحديد الأوجه في نافذة جديدة:

```
cv2.imshow("Detected Faces",image)
```



إغلاق النوافذ المفتوحة

```
cv2.destroyAllWindows
```

توظيف الواجهة الرسومية مع مشروع التعرف على الوجوه

تصميم واجهة رسومية بسيطة تحتوي العناصر التالية:

- صورة Image: عرض الصورة.
- صندوق النصوص TextBox: يستقبل من المستخدم مسار الصورة التي سيتم معالجتها.
- إنشاء دالة: التعليمات البرمجية للتعرف على الوجوه.
- زر Button: بدء عملية معالجة الصورة وعرضها.

بناء البرنامج: صدر البرنامج كملف تنفيذي.

يجب التأكد من تثبيت المكتبة الخارجية pip install ICT_KW

```
import cv2, ICT_KW

def detect_faces():
    global image_textbox, picture_box
    face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
    image_path = image_textbox.get_text()
    image = cv2.imread(image_path)
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGRA2GRAY)
    faces = face_cascade.detectMultiScale(gray_image, scaleFactor=1.1, minNeighbors=5)
    for (x, y, w, h) in faces:
        cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)
    picture_box.edit(image_from_cv2=image)

ICT_KW.window(size=(500, 555), title='Face Detection')
picture_box = ICT_KW.new_Image(location=(0, 0), size=(500, 500))
ICT_KW.new_Label(location=(0, 500), size=(500, 25), align="right", font_size=12, text="أدخل مسار الصورة")
image_textbox = ICT_KW.new_TextBox(location=(0, 500), size=(400, 25))
ICT_KW.new_Button(location=(200, 530), size=(100, 25), text="Detect", todo=detect_faces)
ICT_KW.run()
```

خطوات إعداد المشروعات

سنستعرض الآن طريقة إعداد المشروعات ضمن ضوابط وإرشادات لتنسيق العمل باتباع الخطوات التالية:

مخطط اعداد مشروع في بايثون

إعداد فريق العمل

يتكون الفريق من ٤ إلى ٥ أعضاء، تقسم المهام بينهم على النحو التالية:

- قائد الفريق: المسؤول عن إدارة عمل الفريق وتجميع الملفات والمستندات المطلوبة وتقديمها للمعلم.
- مُصمم الواجهة الرسومية: المسؤول عن تصميم الواجهة الرسومية والإهتمام بالشكل الجمالي العام للنموذج البرمجي.
- مُصمم دوال الاستجابة للعناصر الرسومية: المسؤول عن تصميم الاستجابة للأحداث.
- مُعد العرض التقديمي: المسؤول عن إعداد وتصميم العرض التقديمي للمناقشة.

اكتب بيانات أعضاء الفريق موضحاً وظيفة كل عضو:

اعتماد المعلم: التاريخ:

١. فكرة المشروع

يناقش أعضاء الفريق ويحدد موضوع يثير اهتمامهم، سواءً كان تطبيقاً أو لعبة أو أداة تحليل البيانات وغيرها، على أن يكون مُلم بالشروط التالية:

- أن تكون الفكرة واضحة وقابلة للتطبيق (شرح المشكلة والحلول).
- تحديد الفئة المستفادة وتوضيح كيفية الاستفادة.
- أن تحتوي على عدة مصادر خارجية يسهل الرجوع إليها.
- أن تعتمد على تقنية اكتشاف الوجوه أو أي تقنية من تقنيات الذكاء الاصطناعي.

اكتب فكرة الشروع بعد مناقشة أعضاء الفريق:

اعتماد المعلم: التاريخ:

٢- إعداد بيئة العمل

تثبيت ما يحتاج إليه الفريق من برمجيات وأدوات لإعداد المشروع، ومنها:

- PyCharm: لتصميم النموذج البرمجي.
- PowerPoint: لإعداد العرض التقديمي.
- Windows screen recorder: لتسجيل الشاشة إستعداداً لتوثيق المشروع.
- Draw. io: لرسم خريطة التدفق للمشروع.
- Microsoft Edge: للبحث عن المعلومات.

وأي مستلزمات أخرى تخدم المشروع.

٣- إعداد هيكل المشروع

يشترط النموذج البرمجي تواجد الشروط التالية:

- أن يُعد باستخدام لغة Python.
- أن يحتوي على واجهة رسومية.
- أن يوظف تقنية التعرف على الوجوه أو أي تقنية من تقنيات الذكاء الاصطناعي.
- أن يعد خريطة تدفق واجهة رسومية.

صمم الواجهة الرسومية/ صمم الاستجابة للأحداث:

اعتماد المعلم: التاريخ:

٤- كتابة التعليمات البرمجية

يقوم مصمم الواجهة الرسومية ومصمم الاستجابة للأحداث بكتابة التعليمات البرمجية استناداً لما تم التخطيط له مسبقاً.

٥. اختبار المشروع

يقوم مصمم الواجهة الرسومية، ومصمم الاستجابة للأحداث باختبار البرنامج والتأكد من خُلُوّهِ من الأخطاء البرمجية، اللغوية، والعلمية.

٦. تطوير المشروع

يقوم مصمم الواجهة الرسومية، ومصمم الاستجابة للأحداث بتبسيط وترتيب التعليمات البرمجية إذا أصبحت صعبة القراءة، وإضافة التعليقات والبيانات الإرشادية التي توضح مهمة التعليمات البرمجية.

٧. توثيق المشروع

يقوم مصمم العرض التقديمي بإنشاء عرض تقديمي يشتمل على الجوانب التالية:

- عرض بيانات المدرسة وأعضاء الفريق.
- الترحيب بالمعلم وزملائه المتعلمين.
- عرض مقدمة عن فكرة المشروع موضحاً الهدف والمشكلة والفئة المستفادة والحل.
- عرض النموذج البرمجي.
- عرض المصادر.
- فتح باب الأسئلة والمناقشة.
- الختام.

و تسليم مجلد مجمع لجميع ملفات المشروع.

٨. مشاركة المشروع

يقوم الفريق بعرض المشروع ومناقشته مع المعلم والمتعلمين.

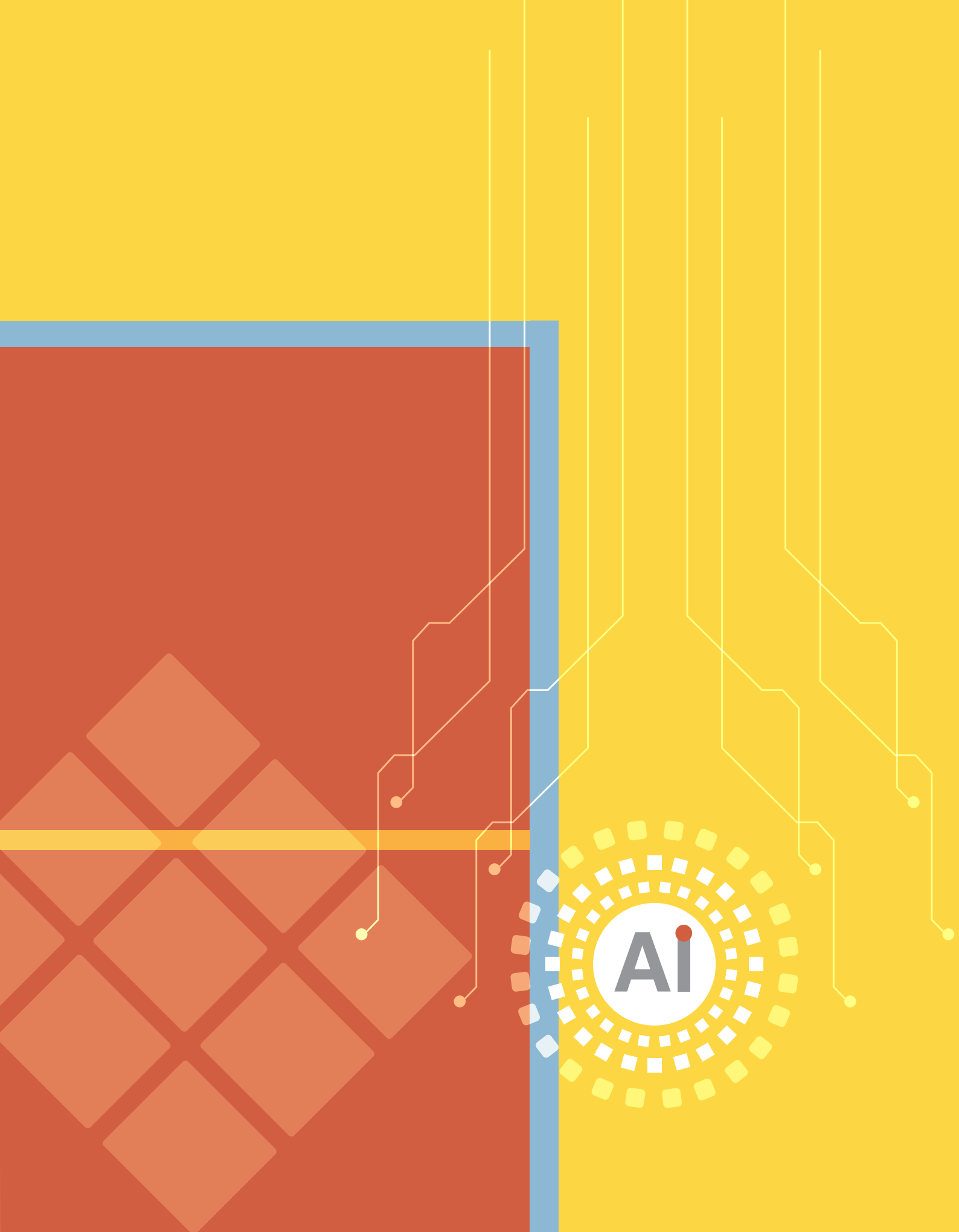
٩. الاستمرار في التعلم

التوسع في الإطلاع على مستجدات الذكاء الاصطناعي وتنمية قدرات المتعلم البرمجية من خلال الدورات التدريبية والمنتديات الحاسوبية.

مشاريع إثرائية

المراجع

- مؤسسة بايثون للبرمجيات. (2024). دليل بايثون الرسمي: إصدار 7. https://www.python.org/doc.
- منظمة الأمم المتحدة للتربية والعلم والثقافة، « الذكاء الاصطناعي في التعليم» (2017).
- خارطة الطريق للتحول الرقمي للمؤسسات الحكومية في دولة الكويت، د. عبد الله محمد عبد الكريم المطوع، مركز دراسات الخليج والجزيرة العربية. العدد 32 . 2023م.
- التوصية الخاصة بأخلاقيات الذكاء الاصطناعي- منظمة الأمم المتحدة للتربية والعلم والثقافة (اليونسكو)، 2021م.
- محمد لحلح. (2020). مدخل الى الذكاء الاصطناعي وتعلم الآلة. شركة حسوب وأكاديمية حسوب.
- نرمين مجدي. (2020). الذكاء الاصطناعي وتعلم الآلة. صندوق النقد العربي.
- منظمة الأمم المتحدة للتربية والعلم والثقافة. (2021). الذكاء الاصطناعي والتعليم. منظمة الأمم المتحدة للتربية والعلم والثقافة.
- سدايا. (سبتمبر 2023). مبادئ أخلاقيات الذكاء الاصطناعي. سدايا.


Ai