وزارة التربية
Ministry of Education
State of Kuwait | دولـــة الكويـت

تقنية المعلومات 10

الصـف العاشر

الفصـل الدراســي الثاني

المرحلة الثانوية

تقنية المعلومات 10

إشراف

أ. منى سالم عوض سالم (رئيس اللجنة)

تأليف

أ. رأفت صابر عبداللاه أحمد　　د. أشرف رضوان رضوان سليمان

د. يوسف منصور يوسف الخليفي　　أ. منيرة خالد محمد أبو شيبة

أ. إبراهيم عبدالله إبراهيم المياس

تصميم

أ. إيمان عبدالعزيز أحمد الفارسي　　أ. سنيه محمد علي المؤمن

الطبعة الأولى

1447 هـ

2026/2025 م

الطبعة الأولى 2025/2024م

الطبعة الثانية 2026/2025م

لجنة المواءمة

إشراف

أ. منى سالم عوض سالم (رئيس اللجنة)

أ. حسام الدين علي عبدالقادر

أ. رأفت صابر عبداللاه أحمد

د.يوسف منصور يوسف الخليفي

أ. منى مرزوق مخلد العازمي

أ. أحمد محمد عبد الله عيسى

د. أشرف رضوان رضوان سليمان

أ. منار مصطفى عبدالحميد جمال

أ. إبراهيم عبدالله إبراهيم المياس

تصميم

أ. ساره ياسين عبد الله الأمير

إخراج

د. أشرف رضوان رضوان سليمان

المراجعة العلمية

د. أشرف رضوان رضوان سليمان

أ. فاطمة نجم جاسم الهولي

أ. غيداء حسن عبدالله السبتي

أ. محمد عبد الرزاق ملا محمد علي

الفريق المُساند

أ. سنية محمد علي المؤمن - تصميم

طبع في:

أودع بمكتبة الوزارة تحت رقم (418) بتاريخ 2025/11/30م


بسم الله الرحمن الرحيم

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

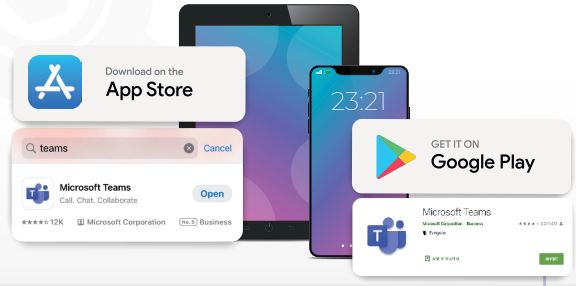
سمو الشيخ صباح خالد الحمد الصباح
ولي عهد دولة الكويت

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

الفصل الرقـــمي Digital Classroom

أولًا :تحميل وتثبيت تطبيق Microsoft Teams

الدخول على متجر تطبيقات الأجهزة الرقمية.



من الأجهزة الذكية

من جهاز الحاسوب

احرص على تحديث تطبيق Microsoft Teams بشكل دوري.

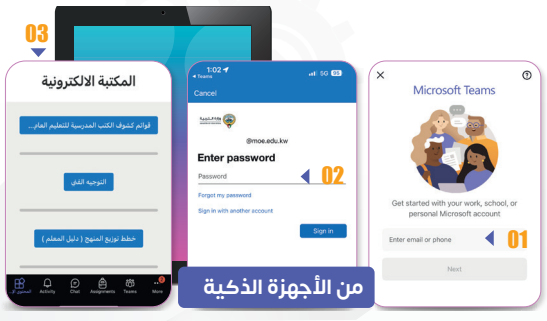
ثانياً: تفعيل Microsoft Teams على الجهاز الرقمي

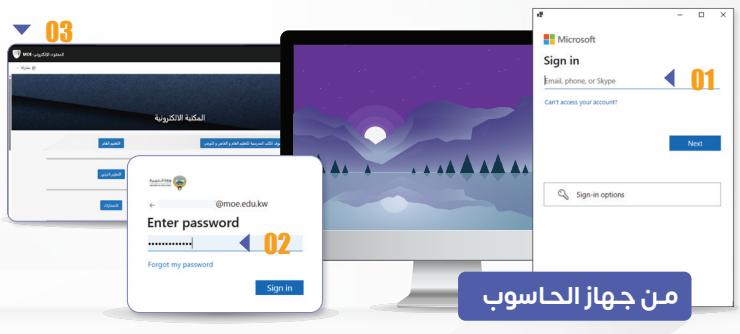
لتشغيل التطبيق اتبع الخطوات التالية:

01 أدخل اسم المستخدم: البريد الإلكتروني الرسمي الخاص بحساب Teams.

02 أدخل كلمة المرور الخاصة بحساب Teams.

03 تنقل بين واجهات التطبيق مثل المحتوى الإلكتروني، وفرق المواد الدراسية.





احتفظ بكلمة المرور في مكان آمن لتسهيل تسجيل الدخول لاحقًا.

لا تشارك بياناتك مع أي شخص غير موثوق به.

مـن هـو حـمد ؟...

هـو مُساعد تعليمـي رقمـي ذكـي يعتمـد علـى تقنيـات الـذكاء الاصطناعـي، مُصمـم لتقديـم الدعـم التعليمـي التفاعلـي لكافـة المناهـج الدراسـية.

حمد دائمًا معك... لتتعلّم بثقة وتنجح بامتياز.

مع حـمد Chat

مـا هـي خـدمـة حـمـد شـات؟

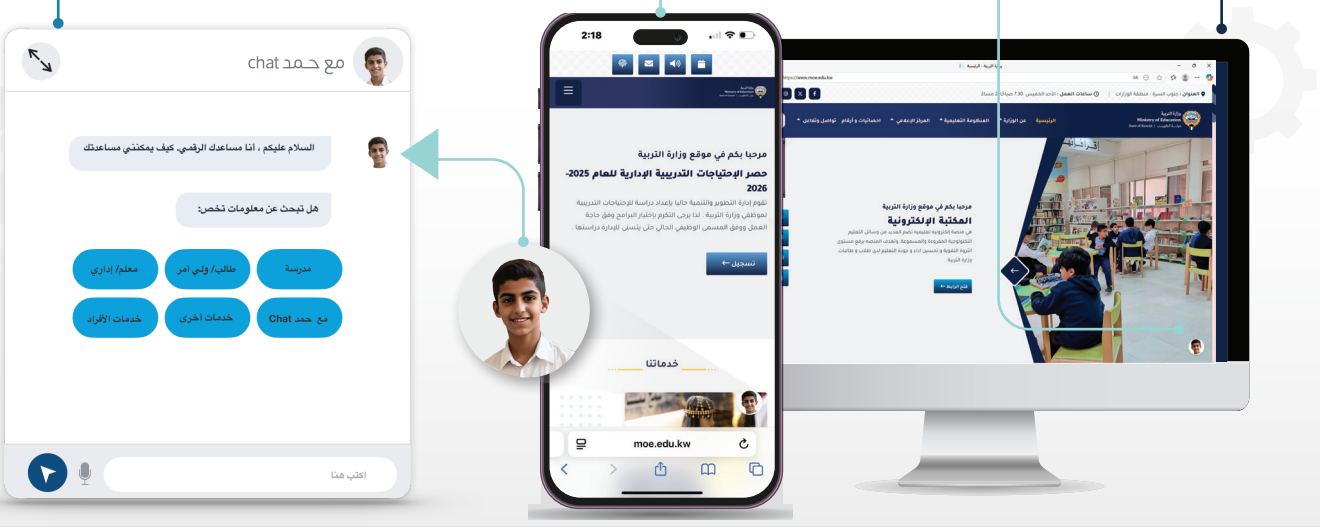
هـي خدمـة المحادثـة الذكيـة المقدمـة مـن وزارة التربيـة فـي دولـة الكويـت، تعتمـد علـى الـذكاء الاصطـناعي، تتمثـل وظيفتـه الأساسـية فـي تسـهيل عمليـة الفهـم والاسـتيعاب مـن خلال تقديـم الشـروحات المبسـطة، والإجابـة علـى الاستفسـارات، وتوجيـه المتعلميـن خلال مسـيرتهم التعليميـة.

أيـــن أجـــده؟

01 - زيارة الموقع الرسمي www.moe.edu.Kw - أو تطبيق وزارة التربية

02 الضغط عـلى صورة حمد شات

03 اكتب استفساراتك، وستتلقى الإجابات بشكل فوري.



المحـــتوى

المقدمة

في ظل الثورة الرقمية التي أحدثت تغييرات جذرية في مختلف المجالات، أصبح إتقان البرمجة ضرورة ملحة لتحقيق الطموحات ومواكبة التطورات التقنية. وقد برزت لغة Python كأداة مثالية للدخول إلى عالم الذكاء الاصطناعي، نظرًا لسهولة تعلمها وشيوع استخدامها في التطبيقات الحديثة.

يهدف هذا الكتاب إلى تزويد المتعلم بمعرفة شاملة حول توظيف لغة Python في تطوير حلول فعالة في مجال الذكاء الاصطناعي، بدءًا من المفاهيم الأساسية التي تم تناولها في الفصل الدراسي الأول وصولاً إلى التطبيقات المتقدمة. نستكشف من خلاله إمكانيات لغة Python عبر وحدات تعليمية متدرجة، تبدأ من الوحدة الأولى التي تتناول هياكل البيانات غير الأولية مثل القوائم، الصفوف، القواميس، والدوال، بما يرسّخ الأساس البرمجي لدى المتعلم.

وفي الوحدة الثانية، ينتقل الكتاب إلى أساسيات الوحدات والحزم، موضحًا دورها في تنظيم الشيفرة وتسهيل تطوير التطبيقات البرمجية، مع ربط المفاهيم النظرية بتطبيقات عملية في مجالات متنوعة مثل الرؤية الحاسوبية، تحليل البيانات، وتطوير الألعاب.

كما يتضمن الكتاب مقدمة شاملة في الذكاء الاصطناعي، تستعرض تطوره، أنواعه، ووظائف تعلم الآلة، مع التركيز على التعلم العميق والذكاء التوليدي وأبرز استخداماتهما في العصر الرقمي. ويتناول العلاقة بين الذكاء الاصطناعي وإنترنت الأشياء (IoT)، موضحًا كيف يسهم هذا الترابط في بناء أنظمة ذكية قادرة على التفاعل مع البيئة واتخاذ قرارات ذاتية.

يسلط الكتاب الضوء أيضًا على أهمية مهارات تصميم الأوامر في تحسين دقة المخرجات من أدوات الذكاء التوليدي، ويعرض المبادئ الأخلاقية المرتبطة بالاستخدام الآمن والمسؤول للتقنيات الحديثة. كما يستعرض أدوات برمجية ذكية وتطبيقات عملية مبسطة تمنح المتعلم فهمًا تطبيقيًا مباشرًا.

وفي الجانب العملي، يقدّم مشروعًا تعليميًا في تقنية التعرف على الوجوه، يوظف أدوات الذكاء الاصطناعي وتعلم الآلة لتطوير أنظمة قادرة على تحليل الصور والتعرف على الوجوه بدقة، بالاعتماد على مكتبة OpenCV المفتوحة المصدر. ويُطبّق هذا المشروع من خلال تجربتين تعليميتين: اكتشاف الوجوه واكتشاف حركة الأشخاص، لتدريب المتعلمين على تحليل الصور ومقاطع الفيديو باستخدام تقنيات الرؤية الحاسوبية، وفهم آليات الذكاء الاصطناعي في معالجة البيانات البصرية.

المؤلفون

الوحدة الأولى

المُعالجة الرقمية

Digital Processing

برمجة Python
Python Programming

1 القوائم Lists

2 الصفوف Tuples

3 القواميس Dictionaries

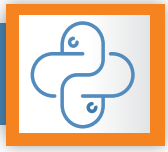
4 الدوال Functions

القوائم Lists

نتاجات التعلم

- التعرّف على هياكل البيانات غير الأولية مثل القوائم Lists، واستخداماتها في تنظيم البيانات.
- تمييز خصائص القوائم (مثل: القابلية للتغيير، الترتيب، الفهرسة) مقارنة ببقية الهياكل مثل tuples وdictionaries.
- إنشاء Lists باستخدام الصيغة الصحيحة في لغة Python، وإدخال البيانات إليها بطرق مختلفة (يدوية أو من المستخدم).
- الوصول إلى عناصر Lists باستخدام الفهرسة الإيجابية والسلبية وتفسير النتائج.
- تعديل عناصر Lists من خلال إسناد قيم جديدة إلى فهارس محددة.
- إجراء عمليات الإضافة والحذف على Lists باستخدام دوال مثل append وinsert وremove وpop.
- معالجة الأخطاء البرمجية الناتجة عن التعامل مع Lists فارغة أو أخطاء في الفهرسة لضمان سلامة البرنامج.

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



هياكل البيانات غير الأولية Non-Primitive Data Structures

تُعد هياكل البيانات غير الأولية Non-Primitive Data Structures من العناصر الأساسية في البرمجة، إذ تُستخدم لتنظيم وتخزين مجموعات من القيم بطريقة مترابطة تتيح معالجة البيانات بكفاءة وسهولة، وتتميز بقدرتها على التعامل مع أنواع مختلفة من البيانات داخل بنية واحدة، مما يجعلها أداة مهمة في تطوير البرامج والتطبيقات المتقدمة، ومن أنواعها:

- القوائم Lists.
- الصفوف Tuples.
- القواميس Dictionaries.

القوائم Lists

تمثل مجموعة مرتبة من البيانات تحتوي على أي نوع من البيانات، (الأعداد الصحيحة والعشرية والسلاسل النصية، ...).

نوع البيانات Data Type	قابلة للتغيير Mutable	العناصر مرتبة Ordered	الفهرسة Indexing	العناصر الفريدة Unique Elements	أنواع مختلفة من البيانات Different Types of Data	المحددات Delimiters
List	نعم	نعم	نعم	لا	نعم	[ ] Square Brackets

الصيغة العامة لإنشاء Lists

إنشاء القوائم Lists باستخدام الأقواس المربعة [ ] ، ويفصل بين عناصرها علامة الفاصلة Comma ,.

list_name = [item1 , item2 , ...]

العمليات على القوائم Lists

1. إنشاء القائمة List :

مثال1: إنشاء List فارغة

```
1    list_1 = []
```

مثال 2: إنشاء List تحتوي العديد من أنواع البيانات

```
1    numbers = [1,2,3,4,5]                          # قائمة من الأعداد الصحيحة
2    decimals = [1.0,2.5,3.14]                      # قائمة من الأعداد العشرية
3    strings = ['Hello','world!','I am','a Student'] # قائمة من السلاسل النصية
4    multi = ['Hello','world!','I am',16,'years old'] # قائمة من أنواع مختلفة من البيانات
```

مثال 3: إنشاء List تحتوي على عدة Lists مختلفة

```
1    name_degree = [['Fahed','Ahmed!','Ali'] , [ 13 , 15, 20]]
```

2. الوصول إلى عناصر List :

يُمكن الوصول إلى عناصر محددة في List باستخدام الفهارس Indexing.

list_name[index]

مثال 4: الوصول إلى عنصر في List

```
1  list_2= [1, 2, 3, 4, 5,'Hello','world!']
2  print(list_2[0])
3  print(list_2[1])
4  print(list_2[-1])
```

Program Output

```
1
2
world!
```

مثال 5: الوصول إلى عنصر داخل List فرعية

```
1  fruits = [
2      ["apple", "banana", "mango"],      # index [0]
3      ["orange", "kiwi", "grape"],       # index [1]
4      ["cherry", "pear", "watermelon"]   # index [2]
5  ]
6  print(fruits[1])
7  print(fruits[1][1])
```

Program Output

```
['orange', 'kiwi', 'grape']
kiwi
```

3. تعديل عناصر List:

مثال 6: تعديل قيمة عنصر

```
1  Liquids = ['Water','Oxygen','Milk', 'Oil']
2  print(Liquids)
3  Liquids[1] = 'Juice'
4  print(Liquids)
```

Program Output

```
['Water', 'Oxygen', 'Milk', 'Oil']
['Water', 'Juice', 'Milk', 'Oil']
```

4. إضافة عناصر List :

مثال 7: إضافة عنصر جديد في نهاية List

```
1  sports= ['Football','Volleyball', 'Tennis','Handball']
2  print(sports)
3  sports.append('Basketball')
4  print(sports)
```

Program Output

['Football', 'Volleyball', 'Tennis', 'Handball']

['Football', 'Volleyball', 'Tennis', 'Handball', 'Basketball']

مثال 8: إضافة عنصر جديد في مكان محدد بالقائمة List

```
1  sports= ['Football','Volleyball', 'Tennis','Handball']
2  sports.insert(2, 'Basketball')
3  print(sports)
```

Program Output

['Football', 'Volleyball', 'Tennis', 'Handball']

['Football', 'Volleyball', 'Basketball', 'Tennis', 'Handball']

5. حذف عناصر من List:

مثال 9: حذف عنصر محدد حسب قيمته من القائمة List

```
1  sports= ['Football','Volleyball', 'Tennis','Handball','Basketball']
2  print(sports)
3  sports.remove('Volleyball')
4  print(sports)
```

Program Output

['Football', 'Volleyball', 'Tennis', 'Handball', 'Basketball']

['Football', 'Tennis', 'Handball', 'Basketball']

مثال 10: حذف عنصر محدد حسب رقم موقعه (الفهرس) من List

```
1  sports= ['Football','Volleyball', 'Tennis','Handball','Basketball']
2  print(sports)
3  sports.pop(1)
4  print(sports)
```

Program Output

```
['Football', 'Volleyball', 'Tennis', 'Handball', 'Basketball']
['Football', 'Tennis', 'Handball', 'Basketball']
```

الفرق بين استخدام remove و pop.

مثال 11: حذف العنصر الأخير من List

```
1  sports= ['Football','Volleyball', 'Tennis','Handball','Basketball']
2  print(sports)
3  sports.pop()
4  print(sports)
```

Program Output

```
['Football', 'Volleyball', 'Tennis', 'Handball', 'Basketball']
['Football', 'Volleyball','Tennis', 'Handball']
```

يمكن حذف جميع عناصر List باستخدام دالة ( )clear.

6. الحصول على طول List:

مثال 12: طباعة طول القائمة (عدد العناصر داخل القائمة)

```
1  GCC= ['Kuwait' , 'UAE' , 'Bahrain' , 'Oman' , 'Saudi Arabia' , 'Qatar']
2  print(len(GCC))
```

Program Output

```
6
```

التطبيقات العملية على Lists

1. التحقق من وجود عنصر معين داخل القائمة:

مثال 13: التأكد من وجود عنصر في List

```
1  fruits_list = ['Apple' , 'Orange', 'Banana' , 'Strawberry' , 'Grapes']
2  fruit = input('Enter the name of the fruit: ')
3  if fruit in fruits_list:
4    print('The fruit is in the fruits list.')
5  else:
6    print('The fruit is not in the fruits list.')
```

Program Output

```
Enter the name of the fruit: Apple
The fruit is in the fruits list.
```

Program Output

```
Enter the name of the fruit: blackberry
The fruit is not in the fruits list.
```

2. تعبئة List من المستخدم:

مثال 14: تعبئة قائمة بإضافة عناصر لها مستخدمًا الحلقة التكرارية while

```
1  student_list = []
2  while True:
3    student_name = input('Enter the name of the student: ')
4    if student_name.lower() == 'stop':
5      break
6    student_list.append(student_name)
7  print(student_list)
```

Program Output

```
Enter the name of the student: Salman
Enter the name of the student: Ali
Enter the name of the student: stop
[ 'Salman' ,  'Ali' ]
```

3. التعامل مع عناصر List من خلال عملية التكرار:

مثال 15: طباعة جميع عناصر القائمة كل عنصر في سطر

```
1  animals = ['cat' , 'tiger' , 'lion']
2  for animal in animals:
3      print(animal)
```

Program Output

```
cat
tiger
lion
```

مثال 16: طباعة جميع عناصر List باستخدام الفهرسة (موقع العنصر في List)

```
1    animals = ['cat' , 'tiger' , 'lion']
2    for n in range ( len (animals)):
3        print(animals[n])
```

Program Output

cat

tiger

lion

أوراق العمل

ورقة عمل 1:

برنامج اختبر معلوماتك

مشكلة البرنامج: استرجاع رموز العناصر الكيميائية، وربطها بأسمائها.

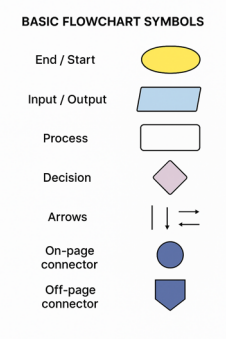
الخوارزمية:

- بداية البرنامج.
- إنشاء List تحتوي على أسماء العناصر الكيميائية element_name.
- إنشاء List تحتوي على رموز العناصر الكيميائية element_symbol بنفس ترتيب List الأولى.
- الإعلان عن المتغير score لحساب درجة المتعلم، وتخزين القيمة (صفر) كقيمة ابتدائية.
- الإعلان عن المتغير I لتخزين عدد عناصر القائمة element_name.
- إنشاء حلقة تكرارية بعدد عناصر List:
  - طباعة العبارة: The Element is، متبوعة بإظهار عنصر من القائمة element_name.
  - عرض عبارة: Enter its symbol لإدخال رمز العنصر من المستخدم وتخزينه في المتغير es.
  - التحقق من تطابق الرمز المُدخل es مع نظيره في القائمة element_symbol، ثم:
    - إذا تطابق الرمز مع نظيره: طباعة عبارة Correct، و زيادة قيمة المتغير score بمقدار 1.
    - إذا لم يتطابق الرمز مع نظيره: طباعة عبارة Incorrect، دون تغيير قيمة المتغير score.

- طباعة فاصل أسطر من خلال تكرار الرمز (-) 30 مرة.
- طباعة العبارة :Student's score is، متبوعة بدرجة المتعلم الكلية.
- نهاية البرنامج.

المطلوب:

ارسم خريطة التدفق



■ البرنامج.

- افتح الملف chemistry.py وقارنه بالخوارزمية السابقة.
- استكمل التعليمات البرمجية ليعمل البرنامج بشكل صحيح.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

```
1   element_name = ['Sodium' , 'Potassium' , 'Copper' , 'Iron' , 'Mercury' , 'Lead']
2   element_symbol =
3   score= 0
4
5   for x
6
7
8       if
9           print
10
11      else:
12          print
13      print("-" * 30)
14  print("Student's score is: " , score, "/" ,l)
```

Lead	Mercury	Iron	Copper	Potassium	Sodium	العنصر الكيميائي
Pb	Hg	Fe	Cu	K	Na	الرمز

جدول (1) قائمة ببعض العناصر الكيميائية ورموزها.

ورقة عمل 2:

برنامج حساب المتوسط الحسابي

مشكلة البرنامج:

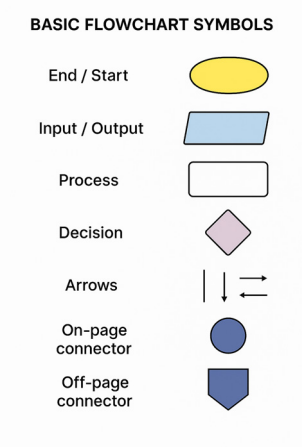
حاجة المستخدم إلى برنامج يُدخل من خلاله أعدادًا موجبة أو صفرية، ثم يُحسب المتوسط الحسابي تلقائيًا بعد الانتهاء من الإدخال.

الخوارزمية:

- بداية البرنامج.
- إنشاء List فارغة باسم numbers_list.
- الإعلان عن المتغير total، وتخزين القيمة (صفر) كقيمة ابتدائية.
- إنشاء حلقة تكرارية لانهائية، تُنفذ التالي:
  - عرض العبارة Enter a number (negative to stop) لإدخال عدد من المستخدم، وتخزينه في المتغير input_number، وإضافته إلى عناصر القائمة numbers_list.
  - إذا كان العدد المُدخل أصغر من صفر، يتم الخروج من التكرار.
  - إضافة العدد المُدخل إلى عناصر القائمة numbers_list.
  - جمع العدد المُدخل إلى قيمة المتغير total.
- حساب المتوسط الحسابي total / len(numbers_list).
- طباعة عبارة: The numbers you entered are.
- طباعة عناصر القائمة numbers_list.
- طباعة عبارة Average value، متبوعة بنتيجة المتوسط الحسابي.
- نهاية البرنامج.

المطلوب:

ارسم خريطة التدفق



■ البرنامج:

- افتح الملف average.py، وقارنه بالخوارزمية السابقة.
- استكمل التعليمات البرمجية ليعمل البرنامج بشكل صحيح.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

```
numbers_List = []
total = 0
while True:
  input_number = int (input("Enter a number (negative to stop): "))
  if input_number < 0:
     # Exit loop
    ▸
  # Insert the new element (number) to numbers_List
 ▸
  total = total + input_number
average = total / len(numbers_List)
print("Numbers you entered are: ")
print(numbers_List)
print(f"Average value {average}")
```

تطوير البرنامج:

- التحقق من احتواء List على عناصر قبل حساب المتوسط الحسابي، وذلك لتفادي حدوث أخطاء ناتجة عن محاولة القسمة على صفر.

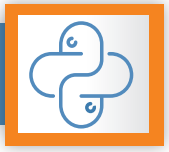
اكتب التعليمات البرمجية

01. .....
02. .....
03. .....
04. .....
05. .....
06. .....
07. .....
08. .....
09. .....
10. .....
11. .....
12. .....
13. .....
14. .....
15. .....
16. .....

الصفوف Tuples
نتاجـــات التعـلم

- التعرف على مفهوم Tuple كنوع من أنواع البيانات غير القابلة للتغيير Immutable في Python.
- التمييز بين خصائص Tuples والأنواع الأخرى من البيانات، مثل Lists.
- إنشاء Tuple يحتوي على عناصر مرتبة باستخدام الأقواس الدائرية.
- الوصول إلى عناصر Tuple باستخدام الفهرسة الموجبة والسالبة.
- تحويل Tuple إلى List بهدف تعديل العناصر، ثم إعادته إلى Tuple.
- تخصيص عناصر Tuple إلى متغيرات متعددة باستخدام تقنية Sequence Unpacking.
- كتابة برامج بسيطة تتعامل مع Tuples لتمثيل بيانات متعددة الأنواع.
- استخدام التكرار لطباعة محتويات Tuples وتنظيم إخراج البيانات.
- اختبار البرامج والتأكد من خلوها من الأخطاء البرمجية أثناء التنفيذ.

يمثـل رمـز الاسـتجابة السـريعة QR رابـط
لملفـات أوراق العمـل، ومصـادر التعلـم



الصفوف Tuples

تعريف Tuples

نوع من أنواع البيانات غير القابلة للتغيير immutable يُستخدم لتخزين سلسلة sequence من العناصر في متغير واحد.

خصائص Tuples

- تنحصر عناصر الصفوف Tuples بين أقواس دائرية ( ) Parentheses.
- يحتفظ بترتيب العناصر كما تم إدخالها ويمكن الوصول إليها باستخدام الفهرسة.
- لا يمكن إضافة أو حذف أو تعديل لعناصره بعد الإنشاء.
- يمكن أن يحتوي على قيم مكررة.
- يقبل أنواع بيانات مختلفة.
- سريعة في التنفيذ لأن بياناتها ثابتة ولا تتغير، فهي تستهلك ذاكرة أقل.

نوع البيانات Data Type	قابلة للتغيير Mutable	العناصر مرتبة Ordered	الفهرسة Indexing	العناصر الفريدة Unique Elements	أنواع مختلفة من البيانات Different Types of Data	المحددات Delimiters
Tuple	لا	نعم	نعم	لا	نعم	( ) Parentheses

الصيغة العامة لإنشاء Tuples

إنشاء Tuple باستخدام الأقواس الدائرية ( ) يحتوي مجموعة من العناصر:

tuple_name = (Item1, Item2, ...)

العمليات على Tuples

1. إنشاء Tuple:

مثال 1: Tuple يحتوي على عناصر من نوع واحد

إنشاء Tuple يحتوي على أيام الأسبوع.

```
1  week_days = ('Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday')
2  print(week_days)
```

Program Output

('Sunday' , 'Monday' , 'Tuesday', 'Wednesday','Thursday','Friday','Saturday')

مثال 2: Tuple يحتوي على عناصر من أنواع مختلفة

إنشاء Tuple يحتوي على بيانات الموقع الجغرافي لعاصمة دولة الكويت.

```
1  location = ('Kuwait', 29.3759, 47.9774)
2  print(location)
```

Program Output

('Kuwait', 29.3759, 47.9774)

مثال 3: Tuple يحتوي على عنصر واحد فقط

إنشاء Tuple يحتوى على اسم المؤلف.

```
1  author = ('ICT Department')
2  print(type (author))
3
4  author = ('ICT Department' , )
5  print(type (author))
```

Program Output

```
<class 'str'>
<class 'tuple'>
```

لاحظ

في حال عدم إضافة الفاصلة (,) لا يعتبر المتغير صفاً Tuple.

2. الفهرسة:

يمكن الوصول إلى العنصر في Tuple بواسطة رقم الفهرس [index].

مثال 4: الوصول لقيمة عنصر داخل Tuple

طباعة اليوم الأول واليوم الأخير في الأسبوع.

week days	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
Index	0	1	2	3	4	5	6
	-7	-6	-5	-4	-3	-2	-1

```
1  week_days = ('Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday')
2  print(week_days[0])
3  print(week_days[-1])
```

Program Output

Sunday
Saturday

3. تعديل Tuple من خلال تحويله إلى List:

يمكن تحويل عناصر Tuple إلى List لإجراء التعديلات المطلوبة عليها، ثم إعادتها إلى Tuple من جديد عند الحاجة.

مثال 5 : حذف عناصر من Tuple

تعديل أيام العمل لتحتوي على أيام الأحد والاثنين والثلاثاء والأربعاء والخميس فقط، وذلك بحذف يومي الجمعة والسبت.

```
1  work_days = ('Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday')
2  work_days = list(work_days)      # Convert Tuple to List
3  work_days.pop()                  #Removes the last item from the list.
4  work_days.pop()                  #Removes the new last item after the previous pop.
5  work_days = tuple(work_days)     # Convert List to Tuple
6  print(work_days)
```

Program Output

('Sunday' , 'Monday' , 'Tuesday', 'Wednesday','Thursday')

4. تقسيم بيانات Tuple على متغيرات Sequence Unpacking:

مثال 7 : تعريف مجموعة من المتغيرات باستخدام Tuple

```
1  country_info = ("Kuwait", "Kuwait City", "Kuwaiti Dinar", "17,818 sq km")
2  country_name, capital, currency, area = country_info # Assign each tuple item to its own variable.
3  print(f"Country: {country_name}")
4  print(f"Capital: {capital}")
5  print(f"Currency: {currency}")
6  print(f"Area: {area}")
```

Program Output

```
Country: Kuwait
Capital: Kuwait City
Currency: Kuwaiti Dinar
Area: 17,818 sq km
```

أوراق العمل

ورقة عمل 3:

نظام الألوان الضوئية الأساسية.

مشكلة البرنامج: إنشاء مصفوفة تمثل الألوان الضوئية الأساسية وطباعتها.

الخوارزمية:

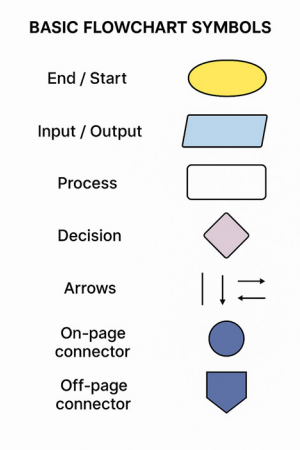
- بداية البرنامج.
- إنشاء ثلاثة Tuples يمثل كل Tuple لون واحد من الألوان الأساسية الضوئية، كما في الجدول التالي:

Tuple name	RED	GREEN	BLUE
red	255	0	0
green	0	255	0
blue	0	0	255

- إنشاء Tuple رئيسي color_rgb يحتوي على الألوان الأساسية الضوئية.
- طباعة العبارة " : Primary Colors" لتوضيح بداية عرض الألوان.
- استخدام التكرار لطباعة كل لون في الصف color_rgb.
- نهاية البرنامج.

المطلوب:

ارسم خريطة التدفق



- البرنامج.
  - أنشئ ملف Python باسم my_color_rgb.py.
  - اكتب التعليمات البرمجية اللازمة.
  - اختبر البرنامج، وتأكد من خلوه من الأخطاء.

اكتب التعليمات البرمجية

01. .....
02. .....
03. .....
04. .....
05. .....
06. .....
07. .....
08. .....
09. .....
10. .....
11. .....
12. .....
13. .....
14. .....
15. .....
16. .....
17. .....

القواميـــس Dictionaries

نتاجـــات التعـلم

- التعرف على Dictionaries كنوع من أنواع البيانات غير الأولية القابلة للتغيير Mutable.
- التمييز بين خصائص Dictionaries مقارنة بالقوائم Lists والصفوف Tuples.
- إنشاء Dictionarie باستخدام الأقواس المعقوفة { } يحتوي على أزواج من المفاتيح والقيم.
- استخدام الفهرسة بواسطة المفاتيح للوصول إلى القيم المرتبطة بها.
- إضافة وتعديل عناصر داخل Dictionarie باستخدام التعيين المباشر.
- حذف عناصر من Dictionarie باستخدام الدوال مثل clear() - popitem() - pop().
- طباعة مفاتيح وقيم Dictionarie باستخدام دالة items() والتكرار for.
- تحليـل البيانـات داخـل Dictionaries واسـتخلاص نتائـج مثـل مجمـوع الدرجـات أو أسـماء الطـلاب الناجحين/الراسـبين.
- توظيـف Dictionarie فـي سـيناريوهات عمليـة، مثـل تخزيـن بيانـات الطالـب أو درجاتـه فـي المـواد المختلفـة.
- اختبار البرامج والتأكد من خلوها من الأخطاء البرمجية أثناء التنفيذ.

يمثـل رمـز الاسـتجابة السـريعة QR رابـط
لملفـات أوراق العمـل، ومصـادر التعلـم

مدخل إلى القواميس Dictionaries

تعتبر القواميس Dictionaries أحد أنواع البيانات غير الأولية، والتي يمكن أن تخزن أكثر من قيمة مثل (القوائم Lists - الصفوف Tuples) وتُمثل بياناتها كأزواج من المفاتيح Keys والقيم Values.

خصائص القاموس:

- تُكتب عناصر Dictionary بين الأقواس المعقوفة { } Curly Braces.
- كل عنصر يتضمن مفتاح Key، والقيمة الخاصة به Value.
- عناصره مرتبة (تحافظ على ترتيب الإدخال)، ومفهرسة باستخدام المفاتيح Keys.
- المفاتيح Keys يجب أن تكون فريدة Unique (غير مكررة)، وفي حال التكرار يتم استبدال القيمة القديمة بالقيمة الجديدة.
- المفاتيح Keys غير قابلة للتعديل Immutable، ويجب أن تكون من أنواع مثل: (string - integer - tuple)، ولا يمكن أن تكون من نوع list أو dictionary.
- القيم Values تُمثل بأي من أنواع البيانات.
- يفرّق بين حالة الأحرف (Capital / Small)، فالمفاتيح المكتوبة بأحرف كبيرة تختلف عن تلك المكتوبة بأحرف صغيرة وتُعد مفاتيح مستقلة.

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

الصيغة العامة لإنشاء Dictionaries

عند إنشاء Dictionary يجب أن يتضمن كل عنصر بالقاموس على: مفتاح Key، وقيمة Value، حيث يُستخدم المفتاح للوصول إلى القيمة المرتبطة به

```
dictionary_name = {
        'key1': value1,
        'key2': value2,
        .....
            }
```

العمليات على Dictionaries

1. إنشاء Dictionary:

يمكن إنشاء Dictionary باستخدام الأقواس المعقوفة { }، وهي الطريقة الأكثر شيوعًا لإنشاء Dictionaries مباشرة.

مثال 1: إنشاء Dictionaries

```
1  student = {
2   "name" : "Ahmed",
3   "grade": "Tenth" ,
4   "age": 15,
5   "city": "Jahra"
6  }
```

لاحظ

يتم وضع عناصر Dictionaries (المفاتيح والقيم) داخل الأقواس { } .

2. الفهرسة:

يمكن استخدام المفاتيح للوصول إلى القيم keys المرتبطة بها values.

مثال2: للوصول إلى عمر الطالب (15) باستخدام المفتاح الخاص به (age)

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

Program Output

```
15
```

3. إضافة وتعديل العناصر في Dictionary:

يمكن إضافة عناصر جديدة أو تعديل القيم الموجودة في Dictionary.

مثال3: إضافة عنصر جديد

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

Program Output

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

مثال 4: تعديل قيمة موجودة

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

Program Output

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

4. حذف العناصر من Dictionary:

يمكن حذف عنصر من القاموس باستخدام الدالة pop().

مثال 5: حذف العنصر الذي مفتاحه age من Dictionary.

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

Program Output

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

مثال 6: حذف المفتاح name من Dictionary وإرجاع (تخزين) قيمته في متغير

```
1  student = { "name" : "Ahmed", "grade": "Tenth" , "age": 15, "city": "Jahra" }
2  student_name = student .pop("name") # Create a variable and remove the key 'name' from the dictionary.
3  print (student)
4  print ("Student name:" , student_name)
```

Program Output

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


لاحظ

الأمر (pop) يحذف عنصر (مفتاح وقيمته) من Dictionary، ويُرجع القيمة المرتبطة بالمفتاح بحيث يمكن تخزينها في متغير، بعد حذفها من Dictionary.

حذف العنصر الأخير من Dictionary باستخدام الدالة ( )popitem.


مثال 7: حذف العنصر الأخير من Dictionary

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

Program Output

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

مسح جميع عناصر القاموس باستخدام دالة ( )clear.


مثال 8: مسح جميع عناصر Dictionary

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

Program Output

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


لاحظ

```
student. clear( )    # مسح جميع العناصر داخل القاموس (المفاتيح والقيم) دون حذف القاموس
del(student)         # حذف القاموس بالكامل من الذاكرة وأي محاولة للوصول إليه سيؤدي إلى خطأ
```

5. التكرار عبر عناصر Dictionary:

يمكن التكرار عبر عناصر Dictionary (المفاتيح والقيم) باستخدام الدالة items()، حيث تُرجع كل عنصر على شكل Tuple يحتوي على المفتاح والقيمة المرتبطة به.

مثال 9: طباعة جميع محتويات Dictionary بطريقة منظمة

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

Program Output

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

طباعة المفاتيح والقيم Dictionary، ثم حساب مجموع الدرجات للمواد الدراسية وطباعتها.

مثال 10: إنشاء Dictionary، والوصول (لقيمة) العنصر بمعلومية (المفتاح)

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

Program Output

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

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

أوراق العمل

ورقة عمل 4:

دول مجلس التعاون الخليجي GCC

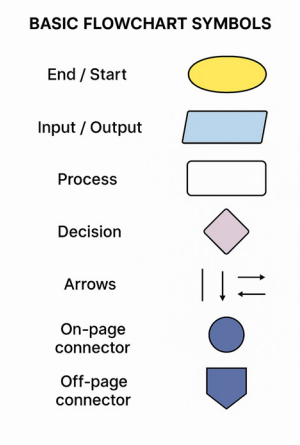
مشكلة البرنامج: تنفيذ عمليات الإنشاء وإضافة العناصر وتعديلها وحذفها على Dictionaries.

الخوارزمية:

- بداية البرنامج.
- إنشاء قاموس gcc_countries يحتوي على أسماء دول مجلس التعاون الخليجي (الدول كمفاتيح والعواصم كقيم).
- إنشاء قاموس my_country يحتوي على معلومات عن دولة الكويت (عناوين المعلومات كمفاتيح ومحتواها كقيم).
- تحديث قيمة المفتاح population في القاموس my_country لتصبح 5000000.
- إضافة مفتاح جديد Currency إلى القاموس my_country، وإدخال قيمته من المستخدم وذلك بعرض عبارة: Enter Kuwait Currency.
- طباعة العبارة: Info About Kuwait.
- طباعة فاصل أسطر من خلال تكرار (-) 40 مرة.
- طباعة عناصر القاموس my_country.
- تعديل بعض عناصر القاموس gcc_countries:
  - حذف العنصر الأخير من Dictionary ليتضمن دول مجلس التعاون الخليجي فقط.
  - طباعة العبارة: The Gulf Cooperation Council Countries.
  - طباعة فاصل أسطر من خلال تكرار (-) 40 مرة.
  - طباعة عناصر Dictionary بعد الحذف باستخدام التكرار المحدد.
- نهاية البرنامج.

المطلوب:

ارسم خريطة التدفق



- البرنامج.
  - افتح الملف dictionary_Kw.py من مجلد أوراق العمل.
  - ادرس التعليمات البرمجية، ثم استكملها حسب الخوارزمية السابقة.
  - اختبر البرنامج، وتأكد من خلوه من الأخطاء.

```
gcc_countries = {
  "Saudi Arabia": "Riyadh",
  "Kuwait": "Kuwait City",
  "United Arab Emirates": "Abu Dhabi",
  "Qatar": "Doha",
  "Oman": "Muscat",
  "Bahrain":"Manama",
  "Yemen": "Sana'a"
}
my_country = {
  "country": "Kuwait",
  "capital": "Kuwait City",
  "population": 8000000
}
#Update key population value to 5000000

#Enter new value to Currency key

print("Info About Kuwait: ")
print("-" * 40)
for key, value in my_country.items():
    # Print element of my_country dictionary

#Delete an element from GCC_countries

print("The Gulf Cooperation Council Countries: ")
print("-" * 40)
for country, capital in gcc_countries.items():
    # Print element of my_country dictionary

```

Program Output

```
Enter Kuwait Currency: Kuwaiti Dinar
info About Kuwait:
---------------------------------------
country : Kuwait
capital : Kuwait City
population : 5000000
currency : Kuwaiti Dinar
The Gulf Cooperation Council Countries:
---------------------------------------
Saudi Arabia : Riyadh
Kuwait : Kuwait City
United Arab Emirates : Abu Dhabi
Qatar : Doha
Oman : Muscat
Bahrain : Manama
```

ورقة عمل 5:

برنامج سجل الطالب

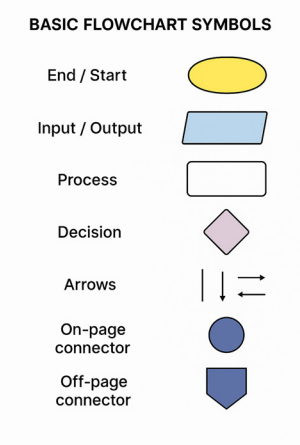
مشكلة البرنامج: التعامل مع عناصر القاموس Dictionary، والقوائم Lists.

الخوارزمية:

- بداية البرنامج.
- إنشاء ثلاثة قواميس: student3 , student2 , student1 يحتوي كل منها على بيانات متعلم واحد: (الاسم name - النتيجة result - مجموع الدرجات total_degrees).
- إنشاء قائمة all_students تتضمن القواميس الثلاثة.
- إدخال بيانات متعلم جديد عن طريق المستخدم:
  - عرض العبارة: Enter new student name، لإدخال الاسم وتخزينه في المتغير name.
  - عرض العبارة: Enter student result (success/failed)، لإدخال النتيجة وتخزينها في المتغير result.
  - عرض العبارة: Enter total degrees لإدخال مجموع الدرجات وتخزينه في المتغير total_degrees.
- إنشاء قاموس new_student للمتعلم الجديد يحتوي على نفس المفاتيح والقيم المدخلة.
- إضافة القاموس new_student إلى القائمة all_student.
- باستخدام التكرارات المتداخلة Nested Loops، يتم طباعة بيانات كل متعلم كالتالي:
  - المرور عبر قائمة قواميس المتعلمين all_students:
    - المرور عبر مفاتيح القاموس وقيمه.
    - طباعة المفتاح والقيمة بتنسيق عرض 20 خانة للمفتاح.
    - طباعة خط فاصل بين بيانات كل متعلم بتكرار (-) 40 مرة.
- نهاية البرنامج.

المطلوب:

ارسم خريطة التدفق



- **البرنامج:**
  - افتح الملف dictionary_students.py من مجلد أوراق العمل.
  - ادرس التعليمات البرمجية، ثم استكملها حسب الخوارزمية السابقة.
  - اختبر البرنامج، وتأكد من خلوه من الأخطاء.

```
#create dictionaries
student1 = {"name": "Ali","result": "success","total_degrees": 650.0}
student2 = {"name": "Khalid","result": "success","total_degrees": 700.0}
student3 = {"name": "Ahmed","result": "failed","total_degrees": 250.0}
#Create a list includes the three dictionaries

# Enter new student data through user input



#Create new Student's Dictionary

#Add Studen's Dictionary to list (all_student)

# Print the data of all students
for student in all_students:
    for key, value in student.items():
        print(f"{key:<20}{value}")
    print( "-" * 40)
```

تطوير البرنامج:

- إنشاء Dictionary باسم student4 (يحتوي بيانات متعلم).
- إضافة Dictionary إلى القائمة all_students باستخدام الدالة المناسبة.

Program Output

```
Enter new student name: Fahed
Enter student result(success/failed): success
Enter total degrees: 500
name            Ali
result          success
total_degrees   650.0
---------------------------------------
name            Khalid
result          success
total_degrees   700.0
---------------------------------------
name            Ahmed
result          failed
total_degrees   250.0
---------------------------------------
name            Fahed
result          success
total_degrees   500.0
```

الدوال Functions
نتاجـات التعـلم

- التعـرف علـى مفهـوم الدالـة فـي Python بوصفهـا وحـدة برمجيـة مسـتقلة تنفـذ مهـام محـددة عنـد اسـتدعائها.
- التمييز بين أنواع الدوال المدمجة Built-in، والدوال المعرفة من المستخدم User-defined.
- توضيح مميزات استخدام الدوال في تنظيم البرامج وتسهيل الصيانة وإعادة الاستخدام.
- كتابة دوال باستخدام الكلمة المفتاحية def مع تحديد الاسم والمعاملات Parameters.
- تحديد الفرق بين المعاملات Parameters والقيم المرسلة Arguments.
- استدعاء الدالة داخل البرنامج بالطريقة الصحيحة وتمرير القيم إليها.
- استخدام تعليمة return لإرجاع نتائج من داخل الدالة عند الحاجة.
- التمييز بين المتغيرات المحلية Local والعامة Global داخل البرنامج.
- اختبار البرنامج والتأكد من خلوه من الأخطاء البرمجية أثناء التنفيذ.

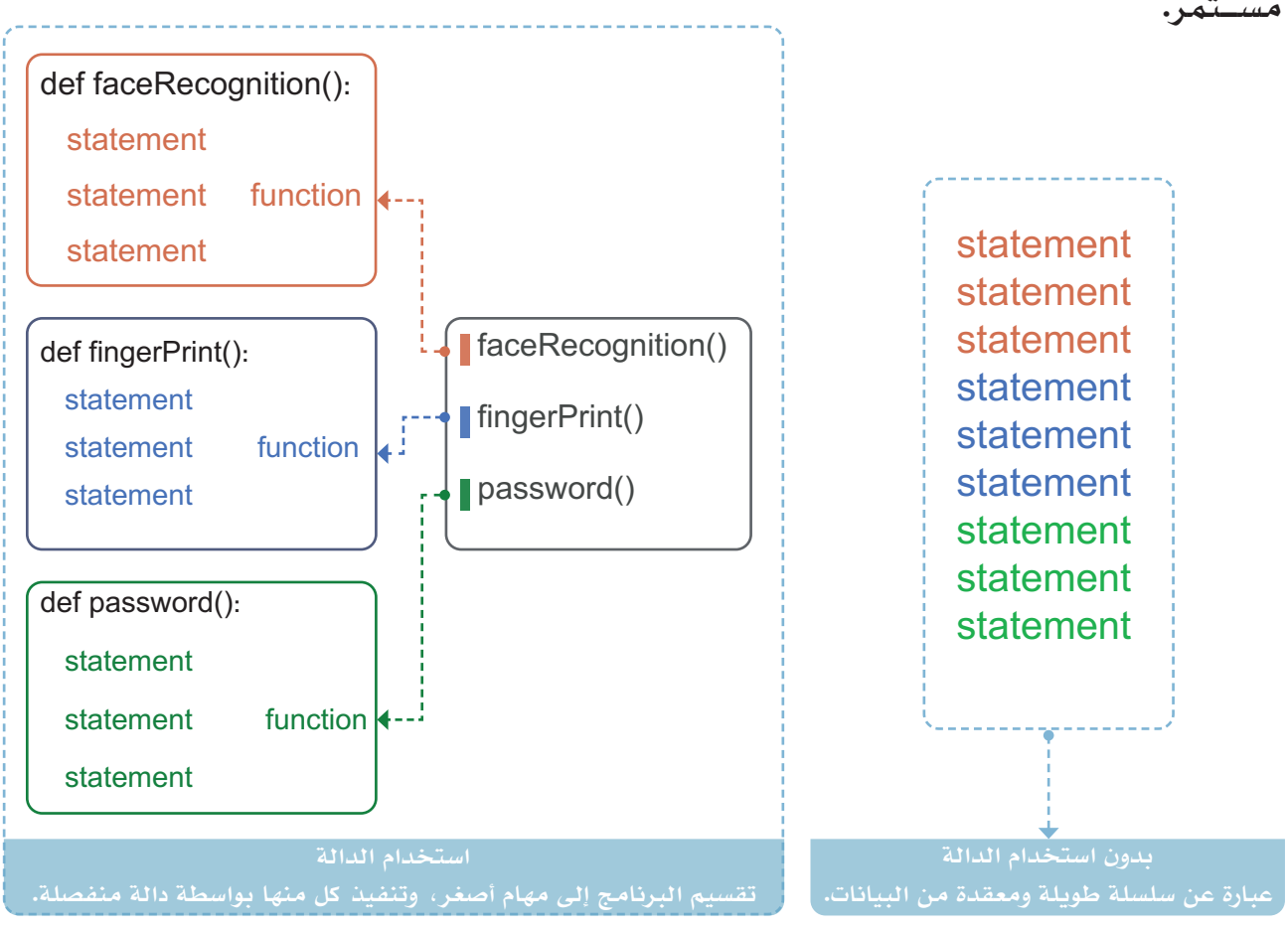
يمثـل رمـز الاسـتجابة السـريعة QR رابـط
لملفـات أوراق العمـل، ومصـادر التعلـم

مدخل إلى الدوال Functions

تعتبـر الـدوال Functions أحـد أدوات البرمجـة الأساسـية فـي Python، التـي تسـاعد علـى كتابـة التعليمـات البرمجيـة بصـورة منظمـة لسـهولة التعامـل معهـا.

مفهوم الدالة Function

مجموعـة مـن التعليمـات البرمجيـة لهـا مهـام محـددة تنفـذ عنـد اسـتدعائها، وهنـاك بعـض البرامـج تـؤدي مهامـاً كبيـرة ومعقـدة ممـا يسـتدعي تقسـيمها إلـى مهـام فرعيـة (دوال)، وتكرارهـا فـي عـدة مواضـع فـي البرنامـج؛ ممـا يسـهل تعديلهـا وتطويرهـا بشـكل مسـتمر.



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

أنواع الدوال Functions

من أنواع الدوال التي سيتم تناولها في هذا الكتاب:

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

لمعلوماتك

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

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

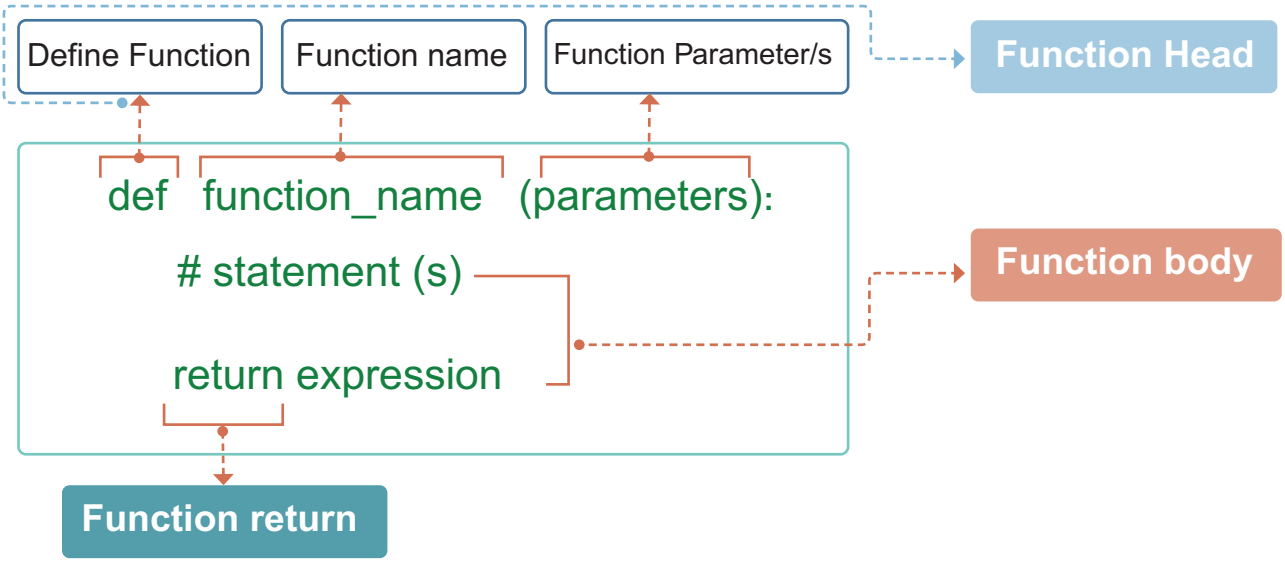
مميزات استخدام الدوال Functions

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.

الصيغة العامة لإنشاء الدالة Function

إنشـاء الـدوال فـي Python يعـد مـن المهـارات الأساسـية لتنظيـم التعليمـات البرمجيـة وإعـادة اسـتخدامها، والتـي يمكـن اسـتدعاؤها فـي أي وقـت عنـد الحاجـة.



شكل (2) الصيغة العامة لإنشاء الدالة

Function Head

- تعريـف الدالـة Define a function: نسـتخدم الكلمـة المفتاحيـة def، وهـي اختصـار لكلمـة definition لتعريـف الدالـة، وهـي مـن الكلمـات المحجـوزة فـي Python.

- اسـم الدالـة Function name: اسـم يُعبـر عنهـا، ويعكـس وظيفتهـا، و يتـم اتبـاع قواعـد تسـمية المتغيـرات عنـد تسـمية الـدوال فـي Python.

- معامـلات الدالـة Function parameter / s: هـي متغيـرات يتـم تعريفهـا فـي الدالة، ويتـم تعييـن قيـم لهـا عنـد اسـتدعائها، والتـي يتـم تمريرهـا للدالـة ويمكـن أن تكـون فارغة.

ويلـي اسـم الدالـة أقـواس دائريـة Parentheses حيـث يُعـرّف معـامِلات الدالـة بداخل الأقـواس، ثم يُتبَـع بالنقطتيـن الرأسـيتين (:) Colon.

Function Body

كتلة من التعليمات البرمجية Statements التي يتم تنفيذها عند استدعاء الدالة، وتميز بمسافة بادئة Indentation تحدد بداية ونهاية الكتلة البرمجية.

Function return

تستخدم لإرجاع قيمة أو أكثر وإنهاء تنفيذ الدالة، وفي حال عدم كتابة تعليمة الإرجاع return ترجع الدالة قيمة فارغة None تلقائياً.

استدعاء الدالة Function

خطوات استدعاء الدالة: يمكن استدعاء الدالة في أي مكان داخل البرنامج لتنفيذ التعليمات البرمجية التي تحتويها.

- كتابة اسم الدالة متبوعاً 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.

بعد تنفيذ كتلة التعليمات البرمجية الموجودة في جسم الدالة ينتقل مرة أخرى إلى السطر البرمجي الذي تم فيه استدعاء الدالة ومن ثم استكمال تنفيذ البرنامج.

إنشاء الدالة Function

مثال 1: إنشاء دالة (بدون معاملات)

إنشاء دالة لطباعة جملة ترحيبية عند استدعائها.

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

Program Output

```
Hello, Kuwait!
```

مثال 2: إنشاء دالة (مع معاملات)

إنشاء دالة لطباعة ناتج جمع عددين، بإدخال قيمة المعاملات Parameters.

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

Program Output

```
40
```

مثال 3: دالة ترجع قيمة

إنشاء دالة لجمع عددين وإرجاع الناتج باستخدام Return Expression.

```
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.

نطاق المتغيرات Variable Scope

مفهوم يحدد مكان وكيفية الوصول إلى المتغيرات داخل البرنامج.

المتغير المحلي Local Variable: المتغيرات التي تعرّف داخل الدالة ولا يمكن الوصول إليها من خارج هذه الدالة.

المتغير العام Global Variable: المتغيرات التي تعرّف خارج الدالة ويمكن الوصول إليها من أي مكان في البرنامج بكتابة اسم المتغير مسبوقاً بالتعليمة global.

مثال 4: مجال المتغير المحلي Local variable في الدالة Functions

عـدم إمكانيـة اسـتخدام المتغيـر المحلـي local_var خـارج نطـاق الدالـة وظهـور رسـالة الخطـأ (المتغيـر غيـر مُعـرف).

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

Program Output

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

مثال 5: مجال المتغير العام في الدالة

تغيير قيمة المتغير العام grade من داخل الدالة.

```
1    grade = 0
2    def change_grade( ):
3      global grade
4      grade = int(input("Enter grade: "))
5    print("The grade before the input is :",grade)
6    change_grade( )
7    print("New grade: ",grade)
```

Program Output

```
The grade before the input is : 0
Enter grade: 18
New grade: 18
```

لاحظ

عنـد حـذف التعليمـة البرمجيـة global grade لا يتـم تغييـر قيمـة المتغيـر grade (القيمـة = 0) عنـد اسـتدعاء الدالـة.

مثال 6: متوالية فيبوناتشي Fibonacci Generator

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

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

- الحصول على العنصر الرابع في السلسلة بجمع الرقمين السابقين له (1+1) فتكون قيمته 2.
- الحصول على العنصر الخامس في السلسلة بجمع الرقمين السابقين له (1+2) فتكون قيمته 3.
- الحصول على العنصر السادس في السلسلة بجمع الرقمين السابقين له (2+3) فتكون قيمته 5.
- . . . . . .

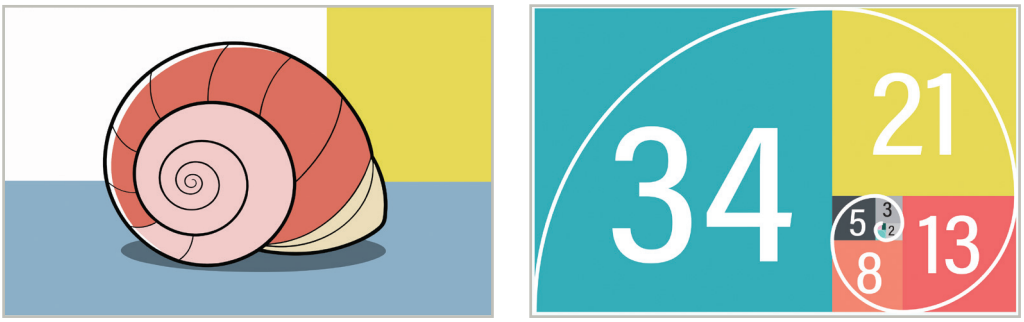
النسبة الذهبية Golden Ratio: قيمة رياضية تساوي تقريبًا 1.618، ويرمز إليها بالحرف اليوناني (Φ)، وترتبط ارتباطًا وثيقًا بمتتالية فيبوناتشي، حيث يقترب حاصل قسمة أي عددين متتاليين في السلسلة من قيمة النسبة الذهبية كلما زادت الأعداد.

يمكن الحصول على النسبة الذهبية في سلسلة فيبوناتشي بقسمة أي عددين متتاليين (العدد الأكبر ÷ العدد الأصغر)، 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 واجهة 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 6: متوالية فيبوناتشي Fibonacci Generator



شكل (3) أمثلة على النسبة الذهبية في الحياة.

البرنامج التالي يوضح كيفية الحصول على سلسلة فيبوناتشي.

```
1    def fibonacci(num):
2        fib = [0, 1]
3        for i in range(num - 2):
4            next_number = fib[-1] + fib[-2]
5            fib.append(next_number)
6        return fib
7    num = int(input("Enter how many Fibonacci numbers (greater than 2):"))
8    print("Fibonacci sequence:", fibonacci(num))
```

قدم حلول برمجية بديلة للحصول على سلسلة فيبوناتشي.

أوراق العمل

ورقة عمل6:

برنامج الآلة الحاسبة

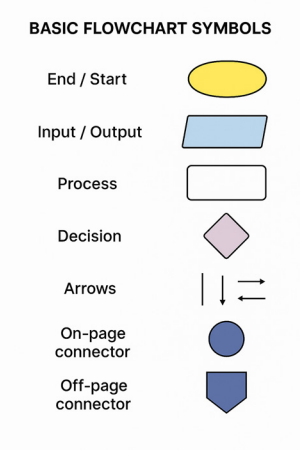
مشكلة البرنامج: توظيف الدوال لإجراء العمليات الحسابية بين عددين مدخلين.

الخوارزمية:

- بداية البرنامج.
- إنشاء أربع دوال:
  - دالة الجمع addition: تستقبل رقمين وتُرجع ناتج جمعهما.
  - دالة الطرح subtraction: تستقبل رقمين وتُرجع ناتج طرحهما.
  - دالة الضرب multiplication: تستقبل رقمين وتُرجع ناتج ضربهما.
  - دالة القسمة division: تستقبل رقمين وتُرجع ناتج القسمة، وتعيد رسالة خطأ Sorry, Can not divide by zero! عندما يكون المقسوم عليه يساوي صفر.
- إنشاء دالة رئيسة simple_calculator:
  - عرض العبارة :Enter first number، ثم تخزين إدخال المستخدم كعدد عشري في المتغير number1.
  - عرض العبارة :Enter second number، ثم تخزين إدخال المستخدم كعدد عشري في المتغير number2.
  - عرض العبارة :Enter your choice: / - * +، ثم تخزين إدخال المستخدم في المتغير user_input.
- التحقق من user_input:
  - إذا كانت +: استدعاء دالة الجمع.
  - إذا كانت -: استدعاء دالة الطرح.
  - إذا كانت *: استدعاء دالة الضرب.
  - إذا كانت /: استدعاء دالة القسمة.
  - إذا كان رمز العملية غير صحيح: طباعة رسالة Invalid input.
  - طباعة الناتج النهائي للعملية.
- استدعاء الدالة الرئيسية simple_calculator.
- نهاية البرنامج.

المطلوب:

ارسم خريطة التدفق



تابع : ورقة عمل 6:

- **البرنامج:**
  - افتح الملف Simple_Calc.py من مجلد أوراق العمل.
  - ادرس التعليمات البرمجية، ثم استكملها حسب الخوارزمية السابقة.
  - اختبر البرنامج، وتأكد من خلوه من الأخطاء.

```
1   def addition(a, b):
2       return a + b
3   def subtraction(a, b):
4       return a - b
5   #Define multiplication operation function
6   
7   
8   def division(a, b):
9       if b == 0:
10          return ("Sorry, Can not divide by zero!!")
11      return a / b
12  def simple_calculator():
13      number1 = float(input("Enter first number : "))
14      number2 = float(input("Enter second number : "))
15      user_input = input("Enter your choice : +, -, *, / : ")
16      if user_input == "+":
17          result = addition(number1, number2)
18      elif user_input == "-":
19          result = subtraction(number1,number2)
20  # call multiplication function
21  
22  
23      elif user_input == "/":
24          result = division(number1, number2)
25      else:
26          print("Invalid input.")
27      print(result)
28  simple_calculator()
```

تابع : ورقة عمل 6:

Program Output

```
Enter first number : 9
Enter second number : 0
Enter your choice : +, *, -, / : /
Sorry, Can not 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.....

15.....

16.....

الوحدة الثانية

المُنتجات الرقمية

Digital Products

الذكاء الاصطناعي
Artificial Intelligence

1 الوحدات والحزم Modules and PacKages

2 مدخل إلى الذكاء الاصطناعي
Introduction to Artificial Intelligence

3 مشروعات الذكاء الاصطناعي
Artificial Intelligence Projects

الوحدات والحزم Modules and PacKages

نتاجـــات التعـلم

- التعرف على مفهوم الوحدات Modules، والحزم Packages في برمجة Python.
- توضيح أهمية استخدام الوحدات والحزم في تنظيم التعليمات البرمجية وإعادة استخدامها.
- التمييـــز بيـــن بعـــض أشـــهر الوحـــدات المدمجـــة مثـــل datetime - random، والحـــزم الخارجيـــة مثـــل NumPy - OpenCV.
- استيراد وحدة أو حزمة باستخدام الأمر import.
- اســـتيراد دالـــة أو عنصـــر محـــدد مـــن وحـــدة أو حزمـــة دون اســـتيرادها بالكامـــل باســـتخدام from .... import ...
- اســـتخدام الحـــزم الجاهـــزة فـــي تنفيـــذ وظائـــف محـــددة مثـــل توليـــد عـــدد عشـــوائي أو التعامـــل مـــع الصـــور والفيديـــو.
- تثبيت الحزم الخارجية باستخدام الأمر pip install من خلال Terminal في PyCharm.
- البحـــث عـــن مكتبـــات مناســـبة لمجـــالات مختلفـــة (مثـــل الـــذكاء الاصطناعـــي أو الأمـــن الســـيبراني) وتثبيتهـــا.
- تنفيذ برامج بسيطة توظف حزم خارجية بعد تثبيتها بنجاح.
- ربـــط اســـتخدام الحـــزم البرمجيـــة بتطبيقـــات حقيقيـــة فـــي مجـــالات متنوعـــة (مثـــل الرؤيـــة الحاســـوبية — تحليـــل البيانـــات — الألعـــاب).

يمثـــل رمـــز الاســـتجابة الســـريعة QR رابـــط لملفـــات أوراق العمـــل، ومصـــادر التعلـــم

مدخل إلى الوحدات والحزم

تُعد الوحدات Modules والحزم Packages من أهم أدوات تنظيم التعليمات البرمجية، حيث تتيح للمبرمجين كتابة برامج قابلة لإعادة الاستخدام، سهلة الصيانة، ومنظمة من خلال تقسيم البرنامج إلى أجزاء بسيطة.

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.

Modules

ملف يتضمن مجموعة من التعليمات البرمجية، قابلة لإعادة الاستخدام، يمكن استدعاؤها واستخدامها في البرامج لتسهيل تنفيذ العديد من المهام.

Packages

مجموعة من الوحدات Modules المرتبطة ببعضها البعض.

أهمية استخدام الوحدات والحزم

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

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

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

أشهر الوحدات Modules والحزم PacKages

- وحدة Datetime : تتعامل مع الدوال الخاصة بالتاريخ والوقت.
- وحدة Random : تتعامل مع الدوال الخاصة لتوليد أعداد عشوائية وتنفيذ عمليات تعتمد على العشوائية.
- وحدة Sqlite3 : الاتصال بقواعد البيانات وإدارتها.
- حزمة OpenCV : تعمل في مجال الرؤية الحاسوبية Computer Vision، وتحليل الصور ومقاطع الفيديو.
- حزمة NumPy : تتعامل مع الدوال الخاصة بالعمليات الحسابية، الجبر الخطي، والمصفوفات.

أقسام الوحدات Modules والحزم PacKages

يمكن تقسيم الوحدات والحزم إلى:

1 الخارجية Third-Party

وحدات وحزم غير مدمجة، تم إنشاؤها من قبل مبرمجين آخرين، يتم تثبيتها عند الحاجة.

2 القياسية Standard

وحدات وحزم مدمجة، يتم تثبيتها تلقائياً مع لغة Python.

3 الخاصة Custom

وحدات وحزم يتم إنشاؤها من قبل المبرمج نفسه ضمن مشروعه البرمجي.

تثبيت الحزم PacKages الخارجية

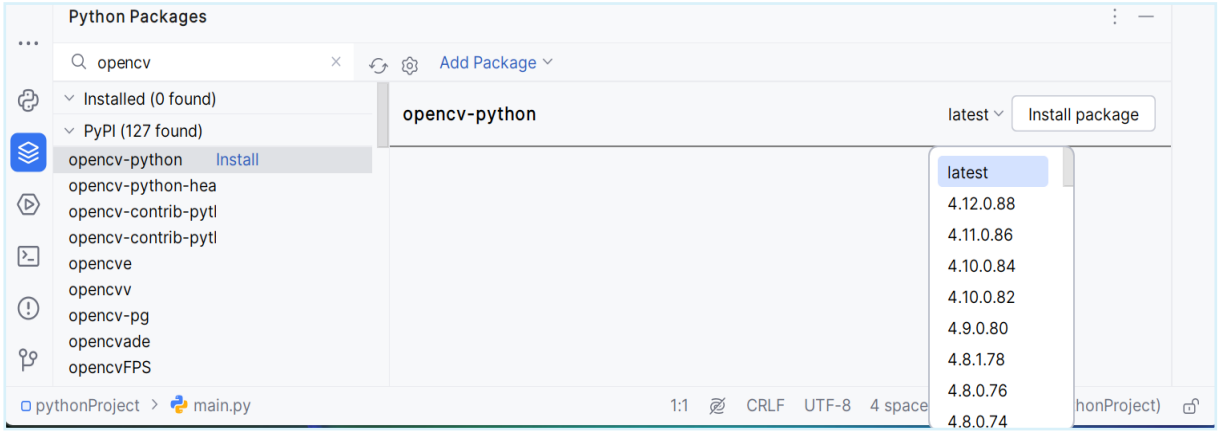
توجد العديد من الطرق لتثبيت الحزم الخارجية، فمثلاً لتثبيت حزمة opencv في Python يمكن استخدام إحدى الطرق التالية:

الطريقة الأولى: استخدام الأمر pip install يليه اسم الحزمة Package name

- من خلال أداة Terminal في PyCharm.
- كتابة الأمر: pip install opencv-python.

الطريقة الثانية: استخدام الأداة Python Packages .

- البحث عن الحزمة المناسبة وتثبيتها من خلال الأمر install واختيار الإصدار المناسب للحزمة.



ملاحظة

يُمكن إلغاء تثبيت حزمة خارجية نستخدم من أداة Terminal الأمر التالي:

```
pip uninstall package-name
```

مثال:

```
pip uninstall opencv-python
```

الصيغة العامة لاستيراد الوحدات Modules والحزم PacKages

يُمكن استخدام الوحدات والحزم باستيرادها أولاً من خلال الصيغة التالية:

```
import module/package-name
```

مثال 1: استيراد وحدة random لتوليد عدد عشوائي صحيح من ١ إلى ١٠

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

Program Output

```
The random integer is = 2
```

يمكن استيراد عنصر محدد من الوحدة أو الحزمة دون الحاجة إلى استيرادها بالكامل.

مثال 2: استيراد الدالة randint فقط من الوحدة random

```
1  from random import randint
2  random_num = randint(1, 10)
3  print(f'The random integer is = {random_num}')
```

Program Output

```
The random integer is = 8
```

مثال 3: استدعاء صورة وعرضها بعد تحويلها إلى اللون الرمادي.

```
1  import cv2
2  image_path = input('Enter the image path: ')
3  colored_image = cv2.imread(image_path)
4  grayed_image = cv2.cvtColor(colored_image, cv2.COLOR_BGR2GRAY)
5  cv2.imshow('Colored Image', colored_image)
6  cv2.waitKey(2000)
7  cv2.destroyAllWindows()
8  cv2.imshow('Grayed Image', grayed_image)
9  cv2.waitKey(0)
10 cv2.destroyAllWindows()
```

Program Output

```
Enter the image path: RGB.jpg
```


التفسير

تحميل الصورة المُلونة المُراد معالجتها من المسار المحدد.

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

تحويل الصورة من نظام الألوان BRG إلى نظام الألوان Grayscale تدرج الرمادي.

```
grayed_image = cv2.cvtColor(colored_image, cv2.COLOR_BGR2GRAY)
```

عرض الصورة colored_image.

```
cv2.imshow('Colored Image', colored_image)
```

الانتظار ثانيتين قبل استئناف معالجة التعليمات البرمجية.

```
cv2.waitKey(2000)
```

إغلاق جميع النوافذ المفتوحة باستخدام cv2.

```
cv2.destroyAllWindows()
```

عرض الصورة grayed_image.

```
cv2.imshow('Grayed Image', grayed_image)
```

الانتظار حتى الضغط على أي مفتاح من لوحة المفاتيح.

```
cv2.waitKey(0)
```

إغلاق النوافذ المفتوحة.

```
cv2.destroyAllWindows()
```

مثال 4: استدعاء فيديو وإغلاقه عند الانتهاء من التشغيل.

```
import cv2
video = cv2.VideoCapture(input('Enter video file path: '))
while True:
    ret, frame = video.read()
    if ret is False:
        break
    cv2.imshow('Video Frame', frame)
    cv2.waitKey(25)
cv2.destroyAllWindows()
```

Program Output

Enter video file path: video.mp4

التفسير

إنشاء كائن للاتصال بمصدر الفيديو، يمكنه قراءة إطارات الفيديو بالتتابع.

```
video = cv2.VideoCapture(input('Enter the video path: '))
```

لاحظ

يمكن تشغيل الكاميرا باستخدام التعليمة البرمجية التالية:

```
video = cv2.VideoCapture(0)
```

إنشاء تكرار لا نهائي لتنفيذ التالي:

```
while True:
```

قراءة الإطار الحالي من الفيديو وتخزينه في المتغير frame، وتخزين قيمة منطقية Boolean في المتغير ret تشير إلى نجاح أو فشل عملية القراءة.

```
ret, frame = video.read()
```

إيقاف الحلقة التكرارية إذا كانت قراءة الإطار غير ناجحة بسبب انتهاء الفيديو أو حدوث خلل في الإطار.

```
if ret is False:
    break
```

عرض الإطار الحالي داخل نافذة العرض Video Frame، مع الانتظار لمدة 25 millisecond قبل الانتقال إلى الإطار التالي.

```
cv2.imshow('Video Frame', frame)
cv2.waitKey(25)
```

إغلاق جميع نوافذ العرض المفتوحة التي تم إنشاؤها بواسطة OpenCV.

```
cv2.destroyAllWindows()
```

مدخل إلى الذكاء الاصطناعي

Introduction to Artificial Intelligence

نتاجـــات التعـلم

- التعــرّف علــى مفهــوم الــذكاء الاصطناعــي وتطــوّره وأنواعــه الأساســية، وأهميتــه فــي مجــالات الحيــاة المختلفــة.
- التمييــز بيــن وظائــف تعلّــم الآلــة (التصنيــف، التنبــؤ، الانحــدار، التجميــع، كشــف الشــذوذ) وربطهــا بتطبيقاتهــا.
- توضيح مفهوم التعلّم العميق والذكاء التوليدي وأبرز استخداماتهما.
- تحليل العلاقة بين الذكاء الاصطناعي وإنترنت الأشياء IoT في بناء أنظمة ذكية متكاملة.
- توظيــف مهــارات تصميــم الأوامــر Prompt Design للحصــول علــى مخرجــات دقيقــة مــن أدوات الــذكاء التوليــدي.
- تحليل مراحل إعداد البيانات (جمع، تنظيف، تحليل، تحويل، تصنيف) وأثر جودتها على النماذج.
- مناقشة المبادئ الأخلاقية للذكاء الاصطناعي وأثرها على الاستخدام المسؤول للتقنيات.
- استكشاف الأدوات البرمجية الذكية لتطبيقات الذكاء الاصطناعي وإجراء تجارب بسيطة عليها.

يمثــل رمــز الاســتجابة الســريعة QR رابــط
لملفــات أوراق العمــل، ومصــادر التعلــم

مدخل إلى الذكاء الاصطناعي Introduction to Artificial Intelligence

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 عبد الله 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، وهو الرئيس الإقليمي لمكتب المنظمة العالمية ملست آسيا، 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.

المفاهيم الأساسية

الذكاء الاصطناعي (Artificial Intelligence) علم تصميم أنظمة قادرة على محاكاة القدرات العقلية للإنسان مثل الفهم، والتعلّم، وحل المشكلات، واتخاذ القرار، حيث يسعى إلى جعل الحاسوب قادرًا على التفكير والتصرف بذكاء من خلال تحليل البيانات والتكيّف مع المواقف الجديدة.

تعلم الآلة (Machine Learning) أحد فروع الذكاء الاصطناعي، يعتمد على تمكين الأنظمة من التعلم ذاتياً من البيانات السابقة دون برمجة صريحة لكل مهمة، حيث يتعلم النظام من التجارب والأنماط السابقة ليكوّن قرارات أو تنبؤات جديدة.

وظائف تعلم الآلة

يُستخدم تعلم الآلة في أداء عدد من العمليات الذكية التي تساعد الأنظمة على تحليل البيانات واتخاذ قرارات دقيقة، ومن أبرز هذه الوظائف:

الوظيفة	الهدف الأساسي	مثال تطبيقي
التصنيف Classification	تحديد الفئة أو المجموعة التي تنتمي إليها البيانات الجديدة.	تصنيف البريد إلى «عادي» أو «مزعج»، أو تصنيف الصور إلى «قطة» أو «كلب».
التنبؤ Prediction	هي عملية تقدير نتيجة مستقبلية أو حدث قادم اعتمادًا على بيانات سابقة.	التنبؤ بما إذا كان الطالب سينجح في الاختبار القادم بناءً على أدائه السابق.
الانحدار Regression	يُعدّ أحد أنواع التنبؤ، لكنه يركّز تحديدًا على تقدير القيم الرقمية المستمرة.	التنبؤ بسعر منتج، أو بدرجة الحرارة، أو بكمية المبيعات خلال الأسبوع القادم.
التجميع Clustering	تجميع البيانات المتشابهة في مجموعات دون معرفة مسبقة بالفئات.	تقسيم العملاء إلى مجموعات وفق سلوك الشراء أو نمط الاستخدام.
الكشف عن الحالات غير الطبيعية Anomaly Detection	التعرف على القيم أو السلوكيات غير الطبيعية داخل البيانات.	اكتشاف محاولات الاحتيال البنكي أو الأعطال في أجهزة الاستشعار.

الأنواع الرئيسية لتعلم الآلة Machine Learning (ML)

التعلم الموجّه Supervised Learning

يتعلم النموذج العلاقة بين المدخلات والمخرجات الصحيحة، مثل عمليات تصنيف البريد الإلكتروني إلى «عادي» أو «مزعج».

التعلم غير الموجّه Unsupervised Learning

يكتشف الأنماط والعلاقات الخفية في البيانات غير المصنّفة، مثل عمليات تجميع الصور المتشابهة أو تقسيم العملاء إلى مجموعات سلوكية.

التعلم المعزّز Reinforcement Learning

يعتمد على التجربة والخطأ، ويتعلم من المكافآت والعقوبات لاختيار السلوك الأفضل، مثل تدريب الروبوت على المشي أو قيادة السيارة ذاتيًا.

التعلم العميق Deep Learning (DL)

يمثل التعلم العميق أحد أهم فروع الذكاء الاصطناعي المتقدمة، ويعتمد على الشبكات العصبية الاصطناعية Artificial Neural Networks (ANNs) التي تحاكي طريقة عمل الدماغ البشري.

أهم أنواع الشبكات العصبية في الذكاء الاصطناعي:

الشبكات الالتفافية Convolutional Neural Networks (CNNs)
معالجة الصور والفيديوهات، وتستخدم في التعرف على الوجوه، كشف الأورام في الأشعة.

الشبكات المتكررة Recurrent Neural Networks (RNNs)
تحليل البيانات المتسلسلة زمنياً، وتستخدم في الترجمة الآلية، التعرف على الكلام.

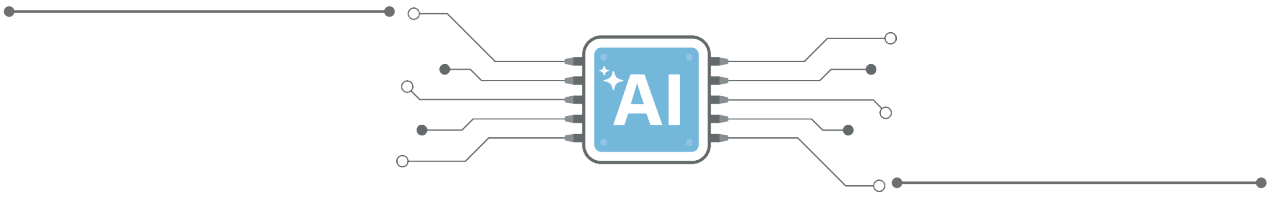
الشبكات التحويلية Transformers
معالجة العلاقات المعقدة في النصوص الطويلة، وتستخدم في الذكاء التوليدي، إنشاء المقالات وتلخيص النصوص.

الذكاء التوليدي Generative Artificial Intelligence

يُعد الذكاء التوليدي أحد أحدث فروع الذكاء الاصطناعي، ويتميّز بقدرته على إنشاء محتوى جديد يشبه ما ينتجه الإنسان، بدلاً من الاكتفاء بتحليل البيانات فقط. ويُستخدم في توليد النصوص، والصور، والموسيقى، ومقاطع الفيديو، وغيرها من أنواع المحتوى الإبداعي، من خلال خوارزميات تعتمد على التعلم العميق Deep Learning والشبكات التحويلية Transformers.

ومن أبرز تطبيقات الذكاء التوليدي ما يُعرف باسم نماذج اللغة الكبيرة Large Language Models (LLMs)، وهي أنظمة ذكية مدرّبة على كميات ضخمة من النصوص لفهم اللغة البشرية وتحليلها وتوليدها بطريقة طبيعية تشبه أسلوب الإنسان.

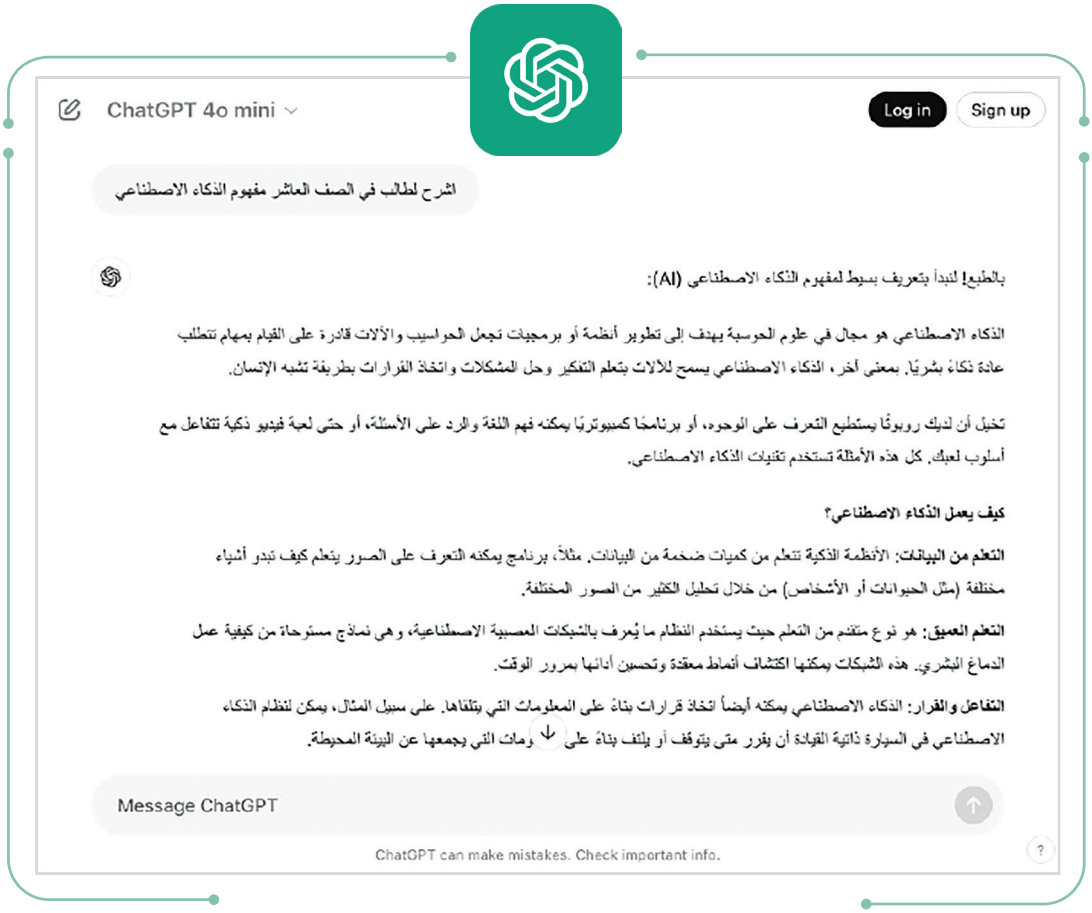
تستطيع هذه النماذج كتابة النصوص، والإجابة عن الأسئلة، وترجمة اللغات، وتلخيص المحتوى بدقة وسرعة عالية، وتُستخدم اليوم على نطاق واسع في التعليم، والبحث العلمي، والإبداع الرقمي، إذ تساعد المتعلمين على توليد الأفكار، وتحسين اللغة، وصياغة المحتوى التعليمي والتفاعلي بذكاء.



أشهر تطبيقات الذكاء التوليدي:

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 وصحيح نحويًا في مختلف اللغات، وفيما يلي مثال على استخدام ChatGPT للإجابة على سؤال تم طرحه على البرنامج.



شكل (1) 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 خيالية تشبه بصرياً صور لأشخاص في العالم الحقيقي، ويمكن معاينة تلك الصور من خلال الموقع الإلكتروني thispersondoesnotexist.com.

شكل (2) يوضح صور غير حقيقية تم إنشاؤها باستخدام تقنية الذكاء الاصطناعي التوليدي

- **توليد الفيديو Video Generation:**

عملية استخدام نماذج الذكاء الاصطناعي التوليدية لإنشاء مقاطع فيديو جديدة انطلاقًا من وصف نصي، وتعتمد النماذج الحديثة "ومنها نموذج Veo من شركة Google" على نماذج الانتشار الكامن Latent Diffusion Models، حيث يتم ضغط البيانات إلى فضاء كامن منخفض الأبعاد ثم تطبيق خطوات الانتشار لإزالة الضوضاء تدريجيًا وبناء الفيديو. وتتيح هذه التقنية إنتاج فيديوهات عالية الدقة مع الحفاظ على التناسق الزمني بين الإطارات، وثبات الألوان والإضاءة والحركة وزاوية الكاميرا، مما يجعلها من أكثر الأساليب دقة وكفاءة في توليد الفيديو في الوقت الحالي.

- **توليد الصوت Voice Generation:**

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

- Microsoft Copilot:

نموذج لغوي يعمل بنظام الذكاء الاصطناعي على إنجاز المهام بسهولة ويسر، تُساعد على كتابة المحادثة نصيًا أو عبر الإدخال الصوتي المباشر حيث أنه يعمل على تجميع المعلومات من الويب وتقديم الدعم وإكمال المهام وغير ذلك الكثير، ويمكن الوصول لهذه الأداة من خلال منصة Microsoft Teams بحساب المستخدم الشخصي.

- Google Gemini:

نموذج لغوي كبير طورته Google قادر على فهم وتوليد النص والرمز والصوت والصورة والفيديو، وهي مصممة لتكون متعددة الوسائط ومرنة، وقادرة على العمل على أجهزة مختلفة، ويعتبر أحد نماذج الذكاء الاصطناعي المتقدمة.

- Internet of Things:

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 الأفراد 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 أو الهاتف الذكي.

تصميم الأوامر Prompt Design / Prompting

يمثل تصميم الأوامر أحد أهم مهارات التعامل مع أدوات الذكاء الاصطناعي التوليدي الحديثة، ويقصد به فن كتابة الطلبات أو التعليمات بشكل دقيق وواضح للحصول على أفضل النتائج من النماذج الذكية مثل ChatGPT أو Gemini، فكلما كان الأمر محددًا وواضحًا وموجّهًا لهدف محدد، كانت الإجابة أكثر دقة وملاءمة.

مستويات الذكاء الاصطناعي Artificial Intelligence Levels

الذكاء الاصطناعي الضيق Artificial Narrow Intelligence (ANI)

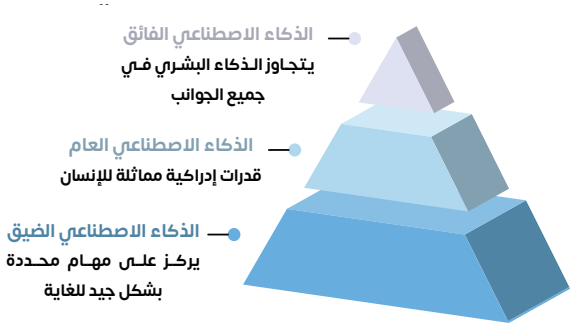
مستوى يتميز في أداء مهمة محددة واحدة أو مجموعة صغيرة من المهام. إنه «ذكي» فقط في مجاله المحدود، ومن أمثلته المساعدون الصوتيون (مثل Siri أو Alexa)، وأنظمة التوصية (مثال YouTube)، وبرامج الشطرنج (التي تتفوق على البشر في اللعبة)، والسيارات ذاتية القيادة.

الذكاء الاصطناعي العام Artificial General Intelligence (AGI)

مستوى يمتلك قدرات شبيهة بالبشر من حيث القدرة على فهم، تعلم، وتطبيق الذكاء البشري لحل أي مشكلة أو مهمة تقريبًا. يمكنه التفكير، التخطيط، التعلم من الخبرة، والتكيف مع بيئات جديدة، ويمثل الجيل القادم من الذكاء الاصطناعي.

الذكاء الذكاء الاصطناعي الفائق Artificial Super Intelligence (ASI)

مستوى أكثر ذكاءً وقدرة يتجاوز قدرات العقول البشرية في كل المجالات تقريبًا، بما في ذلك الإبداع العلمي، حل المشكلات العامة، والمهارات الاجتماعية، وهو افتراضي لو تحقق، فسيحدث تحولاً جذريًا في العالم.



شكل (3) هرم مستويات الذكاء الاصطناعي

مراحل عمل نظام الذكاء الاصطناعي

توضح المراحل التالية كيفية عمل الأنظمة الذكية، حيث تمر بخمس مراحل أساسية:

1 **جمع البيانات** Data Collection
الحصول على بيانات من مصادر مختلفة.

2 **معالجة البيانات** Data Processing
تنظيفها وتنظيمها وتجهيزها للنموذج.

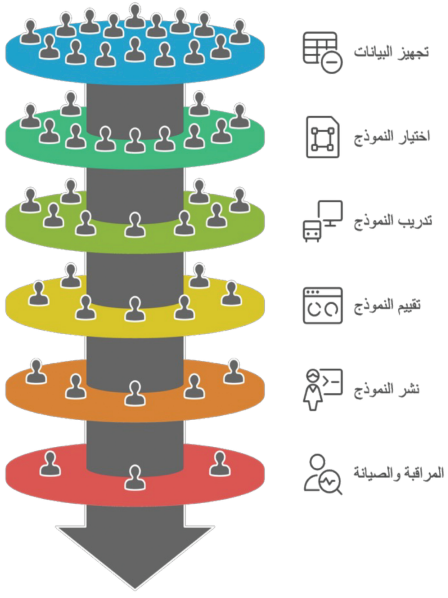
3 **التدريب** Training
تعليم النموذج باستخدام البيانات وتحسين دقته.

4 **التنبؤ** Prediction/Decision
استخدام النموذج لاتخاذ قرارات أو تقديم نتائج جديدة.

5 **التقييم والتحسين** Optimization & Evaluation
مقارنة النتائج بالأهداف وتحسين الأداء المستمر.

6 **نشر النموذج** Model Deployment
بعد تقييم النموذج والتأكد من أدائه الجيد، يمكن نشره للاستخدام في التطبيقات الحقيقية.

7 **المراقبة والصيانة** Monitoring & Maintenance
يجب مراقبته وصيانته بشكل دوري للتأكد من استمراره في العمل بشكل جيد.



شكل (4) مراحل تطوير نظام الذكاء الاصطناعي

العلاقة بين البيانات والذكاء الاصطناعي

الذكاء الاصطناعي لا يعمل دون بيانات ذات جودة عالية، فكلما كانت البيانات أدق وأكثر تنوعًا، كانت نتائج النموذج أذكى وأقرب للحقيقة.

عملية إعداد البيانات:

مرحلة جمع البيانات Data Collection

هي مرحلة تجميع المواد الخام، حيث يتم الحصول على البيانات من مصادر متنوعة (مثل قواعد البيانات، الصور، النصوص، الاستبيانات، أو أجهزة الاستشعار) التي سيتم استخدامها لتدريب النموذج.

مرحلة إعداد ومعالجة البيانات Data Preparation and Prepocessing

هي مرحلة التنظيف العميق. يتم فيها معالجة البيانات الفوضوية عن طريق إزالة التكرارات، تصحيح الأخطاء الواضحة، وملء أو حذف البيانات المفقودة لضمان جودتها.

مرحلة تحليل البيانات Data Analysis

هي مرحلة استخدام الإحصاء والرسوم البيانية لاكتشاف الأنماط الخفية، والعلاقات بين المتغيرات، والقيم غير المعتادة، وفهم توزيع البيانات. هذا يساعد في معرفة ما إذا كانت البيانات متحيزة أم لا.

تحويل البيانات Data Transformation

هي مرحلة توحيد تنسيق البيانات أو تحويل أنواعها لتكون مناسبة لإدخالها في النموذج.

تصنيف البيانات Data Labeling

هي مرحلة وضع البطاقات التعريفية. يتم فيها إضافة إجابات أو تسميات للبيانات الخام، حتى يعرف النموذج ما الذي يجب أن يتعلمه.

أدوات الذكاء الاصطناعي التوليدي، ومجالات استخدامها

تناولت تطبيقات الذكاء الاصطناعي عدة اتجاهات وهي:

- **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.

تمرين:

كيف يمكننا التعامل مع هذه الأدوات بكفاءة والاستفادة منها؟

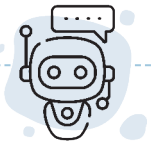
ما هي أدوات الذكاء الاصطناعي؟

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 الروبوت:** تستخدم للتحكم في الروبوت وتوجيهه، وكذلك 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 وتوليد الصور الفنية.
3. Nano Banana: نموذج الذكاء الاصطناعي من Google المعروف باسم Gemini 2.5 Flash Image، والمختص بتوليد الصور وتحريرها.
4. Google Cloud AI: مجموعة من أدوات الذكاء الاصطناعي المقدمة من Google.
5. Amazon Sage Maker: منصة لبناء وتدريب ونشر نماذج التعلم الآلي.

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.

تمرين تطبيقي:

من خلال الدخول الى موقع copilot.microsoft.com.
ابحث عن معلومات تخص إحدى الموضوعات التالية:

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

أرسل النص إلى معلمك عبر فريق الفصل في منصة Teams.

المبادئ الأخلاقية الأساسية في الذكاء الاصطناعي

تُعد الأخلاقيات ركيزة أساسية في تطوير أنظمة الذكاء الاصطناعي، فهي تضمن أن تُستخدم التقنيات الحديثة بطريقة مسؤولة تحترم حقوق الإنسان وتخدم الصالح العام، وترتكز هذه الأخلاقيات على خمسة مبادئ رئيسة يتفرّع عنها عدد من الجوانب المساندة التي تحدد معايير النزاهة والشفافية والأمان في الأنظمة الذكية.

الخصوصية

حماية بيانات الأفراد والمعلومات الشخصية من الوصول غير المصرّح به.

أهم الجوانب المرتبطة به:

- حماية البيانات.
- الموافقة المستنيرة: إعلام المستخدمين بكيفية استخدام بياناتهم والحصول على موافقتهم الصريحة.

العدالة وعدم التحيز

ضمان المساواة وعدم التحيّز في القرارات والنتائج.

أهم الجوانب المرتبطة به:

- عدم التحيّز: تجنّب التفرقة أو التمييز.
- المساواة: تكافؤ الفرص للجميع دون استثناء.

الشفافية

وضوح آلية عمل الأنظمة وقدرتها على التفسير والفهم.

أهم الجوانب المرتبطة به:

- إمكانية التفسير: توضيح منطق القرارات.
- الوضوح: شرح أهداف النظام للمستخدمين.

المُساءلة

تحمّل الأفراد أو المؤسسات مسؤولية نتائج قرارات الأنظمة الذكية.

أهم الجوانب المرتبطة به:

- المسؤولية: التزام الجهات المطوّرة بالممارسات الأخلاقية.
- المحاسبة: القدرة على تبرير نتائج النظام ومراجعتها عند الحاجة.

الأمان

ضمان موثوقية النظام وسلامته من الأخطاء أو الأضرار المحتملة.

أهم الجوانب المرتبطة به:

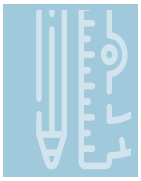
- الموثوقية: أداء مستقر ومتّسق في مختلف الظروف.
- السلامة: عدم التسبب في أي ضرر جسدي أو معنوي للأفراد.

إن الالتزام بهذه المبادئ يعزز الثقة بين الإنسان والتقنية، ويجعل الذكاء الاصطناعي أداة للخير والتطوير لا مصدرًا للخطر أو التمييز، فالذكاء الأخلاقي هو الوجه الإنساني للذكاء الاصطناعي، ويمثل المعيار الذي يوجّه الابتكار نحو خدمة الإنسان والمجتمع.

شكل (5) المبادئ الأخلاقية الأساسية في الذكاء الاصطناعي

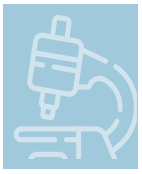
تطبيقات الذكاء الاصطناعي في الحياة اليومية

أصبح الذكاء الاصطناعي جزءًا أساسيًا من أنظمة الحياة الحديثة، حيث تُسهم تقنياته في تسهيل المهام اليومية، وتحسين الخدمات، ورفع كفاءة الأداء في مختلف القطاعات. وتتنوّع تطبيقاته لتشمل مجالات متعددة مثل:



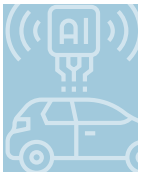
مجال التعليم:

تستخدم في العديد من التطبيقات مثل أنظمة التعلم الذكي، المساعد الافتراضي للطلاب.



مجال الطب:

تستخدم في العديد من التطبيقات مثل تحليل صور الأشعة والتشخيص المبكر للأمراض.

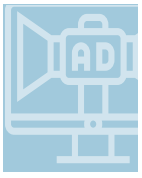


مجال النقل والمواصلات:

تستخدم في العديد من التطبيقات مثل السيارات ذاتية القيادة وإدارة المرور الذكي.

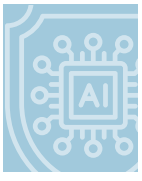
مجال الزراعة:

تستخدم في العديد من التطبيقات مثل تحليل صور المحاصيل وتقدير الإنتاج.



مجال الإعلان:

تستخدم في العديد من التطبيقات مثل تلخيص الأخبار وإنشاء محتوى رقمي ذكي.



مجال الأمن:

تستخدم في العديد من التطبيقات مثل أنظمة المراقبة الذكية وكشف الاحتيال.

العلاقة بين الذكاء الاصطناعي وفروع علوم الحاسوب

تُعد علوم الحاسوب المجال العلمي الأشمل الذي يضم جميع مجالات الحوسبة وتطبيقاتها، فهي توفّر الأسس النظرية والعملية التي يقوم عليها الذكاء الاصطناعي، وتشمل مجالات مثل:

1 البرمجة

تطوير البرامج والتطبيقات.

2 الخوارزميات

وضع خطوات الحل المنطقي للمشكلات.

3 هياكل البيانات

تنظيم البيانات لتسهيل معالجتها.

4 قواعد البيانات

تخزين المعلومات واسترجاعها بكفاءة.

5 الشبكات وأنظمة التشغيل

إدارة الاتصال بين الأجهزة وتشغيل البرمجيات.

ويُعد الذكاء الاصطناعي محورًا مشتركًا يجمع بين العديد من فروع علوم الحاسوب، إذ يُسهم في تطوير قدراتها ويُزوّدها بخاصية «التفكير الذكي» التي تمكّن الأنظمة من التحليل والتعلّم واتخاذ القرار بفاعلية في مجالات متعددة مثل:

الروبوت Robotics

استخدام الذكاء الاصطناعي لاتخاذ قرارات آلية والتحكم الذكي.

الأمن السيبراني Cybersecurity

كشف الهجمات وتحليل التهديدات إلكترونياً.

تصميم الألعاب Game Design

برمجة سلوك ذكي للشخصيات الافتراضية.

علوم البيانات Data Science

دراسة الأنماط واتخاذ قرارات مبنية على المعرفة.

مشروعات الذكاء الاصطناعي

Artificial Intelligence Projects

نتاجـــات التعـلم

- التعرّف على مفهوم الرؤية الحاسوبية ودور مكتبة OpenCV في الذكاء الاصطناعي.
- تمييز الفرق بين الكشف عن الوجوه والكشف عن المشاة من حيث المفهوم والخوارزمية.
- توضيح خطوات استخدام مكتبة OpenCV في تحليل الصور واكتشاف الكائنات.
- تنفيذ مشروع برمجي لاكتشاف الوجوه باستخدام خوارزمية Haar Cascade.
- تعديل البرنامج البرمجي ليتفاعل مع كاميرا الحاسوب في الوقت الحقيقي.
- تصميم برنامج برمجي لاكتشاف المشاة باستخدام خوارزمية HOG، والمصنّف SVM.
- تحليل معايير ضبط الكواشف مثل scaleFactor وminNeighbors وتأثيرها على نتائج الكشف.
- استكشاف تطبيقات الذكاء الاصطناعي في الحياة الواقعية باستخدام الرؤية الحاسوبية (مثل الأمن، القيادة الذاتية، التفاعل الذكي).

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

مدخل إلى مشروعات الذكاء الاصطناعي

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.

وتُعد حزمة OpenCV من أهم الأدوات المستخدمة في تطبيقات الذكاء الاصطناعي المتعلقة بالرؤية الحاسوبية.

فهي تمكّن الحاسوب من رؤية الصور ومقاطع الفيديو وتحليلها لاكتشاف الوجوه والأجسام وحركتها، والتفاعل مع البيئة بذكاء.

حزمة OpenCV في الذكاء الاصطناعي

OpenCV (Open-Source Computer Vision Package)

حزمة برمجية مفتوحة المصدر تُستخدم في الرؤية الحاسوبية Computer Vision ومعالجة الصور ومقاطع الفيديو، وتُعد من أهم الأدوات التي تُستخدم في مشاريع الذكاء الاصطناعي والتعلم الآلي، وتُستخدم في العديد من التطبيقات مثل:

تعمل حزمة OpenCV مع لغات برمجة مختلفة مثل Python - C++ - Java، وتُستخدم غالبًا مع وحدات وحزم مثل NumPy لتحليل البيانات وTensorFlow للتعلم العميق.

بعد التعرف على دور حزمة OpenCV في الذكاء الاصطناعي، سنطبّق عمليًا بعض استخداماتها في مجال الرؤية الحاسوبية، حيث سنتناول مشروعين تطبيقيين هما: اكتشاف الوجوه Face Detection، واكتشاف المشاة Pedestrian Detection، لتوضيح كيف يستطيع الحاسوب تحليل الصور ومقاطع الفيديو والتعرّف على ما بداخلها بذكاء.

المشروع الأول: اكتشاف الوجوه Face Detection

يهدف المشروع إلى كشف الوجوه في الصور أو مقاطع الفيديو باستخدام خوارزمية Haar Cascade الموجودة داخل حزمة OpenCV.

المفهوم العلمي

تُعد عملية كشف الوجوه Face Detection من تطبيقات الرؤية الحاسوبية Computer Vision، وتعتمد الخوارزمية على تعلم الآلة Machine Learning لتحديد الأنماط البصرية في الوجه (مثل العينين والفم والأنف)، مع ملاحظة أنها لا تُميز بين الأشخاص، بل تكتشف أماكن الوجوه فقط.

خطوات تنفيذ المشروع:

تهيئة البرنامج

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

```
pip install opencv-python
```

كتابة التعليمات البرمجية:

استيراد حزمة cv2:

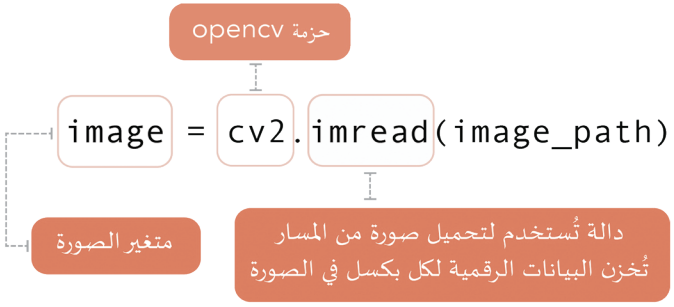
```
import cv2
```

تحميل نموذج كشف الوجوه

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

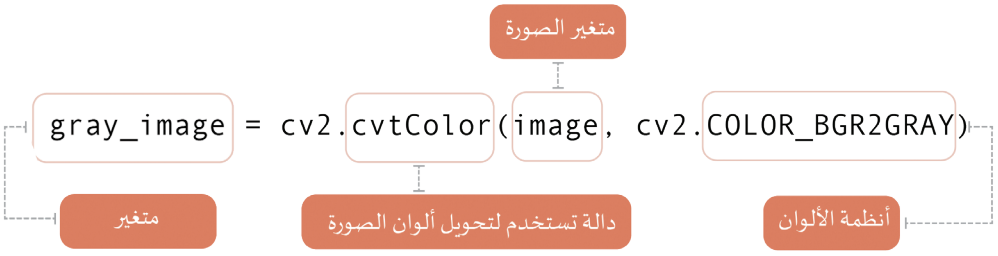
تحميل الصورة المراد معالجتها من المسار المُدخل من المستخدم

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



تحويل الصورة من نظام الألوان BGR إلى نظام الألوان Grayscale تدرج الرمادي

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



Grayscale Image

RGB Image

شكل (1) يوضح الفرق بين الصورة الملونة RGB والصورة ذات الدرجات الرمادية Grayscale.

لاحظ

تحويل الصورة إلى نظام تدرج الرمادي لزيادة كفاءة وسرعة المعالجة، حيث تصبح القيم بين اللونين الأسود والأبيض (0 إلى 255) بدلاً 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)
```

دالة تكشف عن الوجوه في الصورة تقوم بتحديد الموقع والحجم لكل وجه

معاملات تستخدم لتحديد الوجوه بدقة

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

نموذج جاهز للتعرف على الوجوه

scaleFactor=1.1

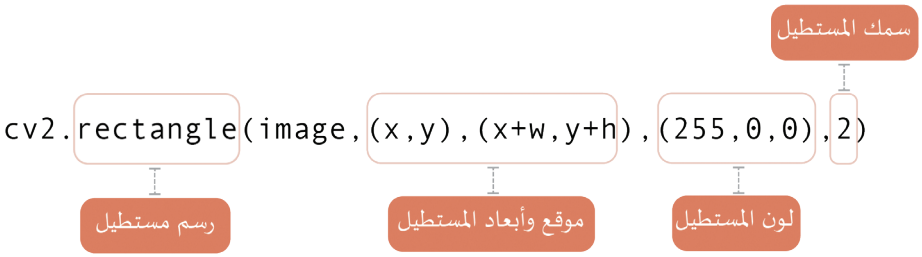
- تحدد نسبة تصغير الصورة في كل خطوة من عملية الكشف.
- قيمة 1. 1 تعني أنه في كل خطوة يتم تصغير الصورة بنسبة 10%.
- كلما كانت القيمة أقل كانت دقة الكشف أعلى، ولكن ذلك يتطلب وقت أطول للمعالجة.

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.

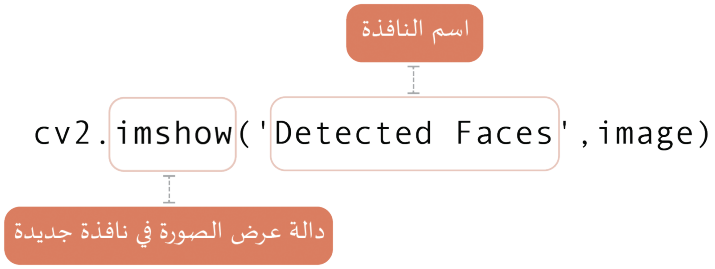
رسم مستطيلات حول الوجوه التي تم اكتشافها في الصورة

```
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)
```



الانتظار إلى أن يضغط المستخدم أي مفتاح على لوحة المفاتيح قبل إغلاق نافذة الصورة ومتابعة تنفيذ البرنامج

```
cv2.waitKey(0)
```

إغلاق جميع النوافذ المفتوحة باستخدام cv2:

```
cv2.destroyAllWindows()
```

التعليمات البرمجية

```
import cv2
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'harrcascade_frontalface_default.xml')
image_path = input('Enter the path to the image: ')
image = cv2.imread(image_path)
grey_image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(grey_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.destroyWindow()
```

الصورة قبل المعالجة　　الصورة بعد المعالجة

شكل (2) يوضح الفرق بين الصورة قبل المعالجة وبعد المعالجة.

تطوير المشروع: الكشف عن الوجوه عبر الكاميرا

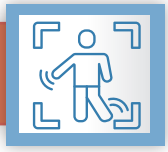
يمكن تطوير المشروع السابق (الكشف عن الوجوه من صورة ثابتة) ليعمل بشكل تفاعلي في الوقت الحقيقي باستخدام كاميرا الحاسوب، ويتم ذلك من خلال استبدال قراءة الصورة الثابتة بأوامر تشغيل الكاميرا وإعادة تحليل الإطارات بشكل متتابع.

التعديل الأساسي للتعليمات البرمجية:

العنصر	Face Detection from Image	Face Detection from Camera
قراءة الصورة	image = cv2.imread(image_path)	cap = cv2.VideoCapture(0)
الحلقة التكرارية لقراءة الصور	لا يوجد	while True: لقراءة الإطارات المستمرة من الكاميرا
عرض النتائج	cv2.imshow('Detected Faces', image)	cv2.imshow('Face Detection', frame) داخل الحلقة

التعليمات البرمجية

```
import cv2

face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5)
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)
    cv2.imshow('Face Detection', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```



المشروع الثاني: اكتشاف المشاة Pedestrian Detection

يهدف المشروع إلى تصميم نظام ذكي يستطيع الكشف عن المشاة في مقاطع الفيديو أو في البث المباشر من الكاميرا، وذلك باستخدام خوارزمية HOG (Histogram of Oriented Gradients) مع المصنّف SVM (Support Vector Machine) داخل حزمة OpenCV، وتُستخدم هذه التقنية في أنظمة المراقبة والأمان والقيادة الذاتية، لما تتميز به من دقة وسرعة في تحليل الصور.

المفهوم العلمي

تعتمد عملية اكتشاف المشاة Pedestrian Detection على تحليل الصورة لاستخراج الخصائص البصرية للأجسام البشرية، مثل شكل الجسم واتجاهاته وحوافه، ويتم ذلك عبر طريقتين:

1. تحليل التدرجات HOG: لاستخراج أنماط الانحناءات والحواف في الصورة.
2. المصنّف SVM: لتمييز ما إذا كانت هذه الأنماط تمثل إنسانًا أم لا.

خطوات تنفيذ المشروع:

تحميل حزمة OpenCV

```
import cv2
```

إنشاء كائن HOG لاستخراج مميزات Features الصور للتعرف على الإنحناءات والحواف

```
hog = cv2.HOGDescriptor()
```

استخدام خوارزمية التعلم الآلي SVM لاكتشاف الأشخاص

```
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
```

فتح الفيديو أو الكاميرا

```
cap = cv2.VideoCapture('video.mp4')
```

حلقة تكرارية لا نهائية، تُستخدم لقراءة كل الإطارات من الفيديو أو الكاميرا

```
while True:
```

قراءة الإطارات، وتُعيد قيمتين: نجاح عملية القراءة، والإطار الحالي

```
    ret, frame = cap.read()
```

إيقاف الحلقة التكرارية إذا كانت قراءة الإطار غير ناجحة بسبب انتهاء الفيديو أو حدوث خلل في الإطار

```
    if not ret:
        break
```

تحليل الإطار باستخدام خوارزمية HOG و SVM لاكتشاف الأشخاص

```
    boxes, weights = hog.detectMultiScale(frame, winStride=(8, 8),
    padding=(8, 8), scale=1.05)
```

- detectMultiScale: تستخدم لكشف الأشخاص داخل الصورة.

- frame: الإطار الحالي الذي يتم تحليله من الفيديو.

- winStride: حجم الخطوة التي تتحرك بها نافذة الفحص أثناء مسح الصورة، كلما كانت صغيرة أصبح الكشف أدق ولكن أبطأ، والعكس صحيح.

- padding: يضيف مساحة حول منطقة الفحص لتحسين جودة الكشف.

- scale: يحدد نسبة تصغير الصورة في كل مستوى من مستويات الفحص، لاكتشاف الأشخاص بمختلف الأحجام

- boxes: تحتوي على مواقع الأشخاص المكتشفين x,y,w,h.

- weights: تحتوي على نسبة الثقة في التعرف على كل شخص تم اكتشافه.

رسم المستطيلات حول المشاة المكتشفين

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

عرض الإطار بعد اكتشاف المشاة داخل نافذة جديدة

```
cv2.imshow('Detected Pedestrians', frame)
```

إيقاف تشغيل البرنامج عند ضغط المستخدم على المفتاح (q)

```
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
```

إنهاء البرنامج وتنظيف الموارد

```
cap.release()
cv2.destroyAllWindows()
```

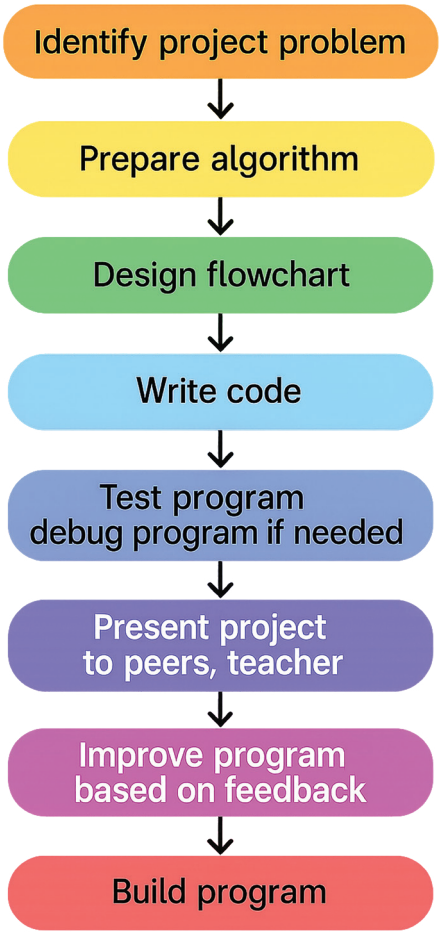
التعليمات البرمجية

```
import cv2
hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
cap = cv2.VideoCapture('video.mp4')          # use (0) to open camera

while True:
    ret, frame = cap.read()
    if not ret:
        break
    boxes, weights = hog.detectMultiScale(frame, winStride=(8, 8), padding=(8, 8), scale=1.05)
    for (x, y, w, h) in boxes:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)
    cv2.imshow('Detected Pedestrians', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

خطوات إعداد المشروعات

إن الهدف الأساسي من إنتاج المشروع هو الاستفادة من المهارات التي تم دراستها وتطبيقها من خلال أوراق العمل بالإضافة إلى تنمية مهارات العمل الجماعي التعاوني، والقدرة على تجميع الوسائط اللازمة لإنتاج المشروع وفق أُسس تحليل النظم والبرمجة والقدرة على تنمية الابتكار.

من خلال دراستك لوحدة البرمجة بلغة Python، والاطلاع على تطبيقاتها المتعددة، والمساعدة المتوافرة في رمز الاستجابة السريع QR في بداية وحدة المُنتجات الرقمية، صمم المشروع الخاص بك، متبعًا الخطوات التالية:

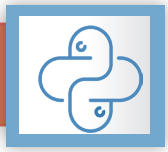


- تحديد مشكلة المشروع.
- إعداد الخوارزمية.
- تصميم خريطة التدفق.
- كتابة التعليمات البرمجية.
- اختبار البرنامج، تصحيح البرنامج إن وجدت.
- عرض المشروع على زملائك، ومعلمك.
- تحسين البرنامج وفق التغذية الراجعة.
- بناء البرنامج.

ضوابط وإرشادات تنفيذ المشروع

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

مخطط إعداد مشروع في PyCharm



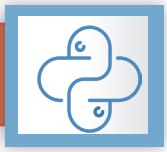
مهام فريق العمل

يتكون الفريق من 4 إلى 5 أعضاء، تُقسم المهام بينهم على النحو التالي:

- قائد الفريق: المسؤول عن إدارة الفريق وتجميع الملفات والمستندات المطلوبة وتقديمها للمعلم.
- مُعد الخوارزمية: المسؤول عن إعداد الخطوات الأساسية لعمل المشروع.
- مصمم خريطة التدفق: المسؤول عن ترجمة الخوارزمية إلى خريطة التدفق.
- المبرمج: المسؤول عن ترجمة خريطة التدفق لتعليمات برمجية واختبارها والتأكد من سلامتها.
- مُعد العرض التقديمي: المسؤول عن إعداد وتصميم العرض التقديمي للمناقشة.

تُقسم المهام بين أعضاء الفريق على النحو التالي:

اسم المتعلم	المهمة	الوظيفة
	قائد الفريق	المسؤول عن إدارة عمل الفريق وتجميع الملفات والمستندات المطلوبة وتقديمها للمعلم



مهام فريق العمل

المرحلة		الناتج المُتوقع
الأولى	التخطيط والتحليل	اختيار فكرة المشروع (مرتبط بتطبيقات الذكاء الاصطناعي).
		تحديد المشكلة والأهداف.
		إعداد وثيقة المتطلبات الأولية.
الثانية	خطة المشروع والتوزيع	توزيع المهام على أعضاء الفريق.
		إعداد خطة تنفيذ المشروع أسبوعياً.
الثالثة	التصميم والتوثيق الأولي	تصميم الواجهة.
		إعداد خطة العمل بالتفصيل.
الرابعة	البرمجة والتنفيذ الأولي	البدء في تنفيذ المشروع عمليًا.
		تجربة الأدوات البرمجية والتقنية.
الخامسة	الاختبار والتحسين	مراجعة الأخطاء البرمجية.
		اختبار الأداء.
		إجراء التعديلات والتحسينات النهائية.
السادسة	التوثيق النهائي والعرض	كتابة التقرير النهائي.
		إعداد العرض التقديمي.
		التدريب على العرض وتقديمه أمام الصف.

ملاحظة: يختلف عدد أسابيع المشروع والمراحل المختلفة وفق خطة توزيع المنهج.

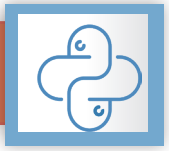
فكرة المشروع

تصميم مشروع تقني تعليمي يعتمد على تقنيات الذكاء الاصطناعي. لحل مشكلة أو تنفيذ فكرة مفيدة بطريقة إبداعية على أن تكون واضحة وقابلة للتطبيق.

يناقش أعضاء الفريق ويحدد موضوع يثير اهتمامهم، سواءً كان تطبيقاً أو لعبة أو أداة تحليل البيانات وغيرها، على أن تكون مُطابقة للشروط التالية:

- أن تكون الفكرة واضحة وقابلة للتطبيق مع شرح المشكلة والحلول.
- تحديد الفئة المستفادة.
- أن تكون الفكرة قابلة لتصميم نموذج برمجي.
- أن تحتوي على عدة مصادر خارجية يسهل الرجوع إليها.
- أن تعتمد على أية تقنية من تقنيات الذكاء الاصطناعي.

سجل فكرة المشروع بعد مناقشة أعضاء الفريق:



أهداف المشروع

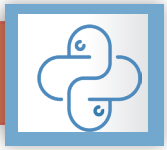
يهدف المشـــــروع إلى إكســـــاب المتعلم مهـــــارات تقنية وإبداعية عبـــــر تصميم وتنفيذ فكـــــرة تعليمية بطريقة منظمـــــة وتعاونية.

سجل أهداف مشروعك:

التوظيف الواقعي في الحياة العملية

يساهم المشروع في ربط التقنية بواقع الحياة من خلال توظيف البرمجة والذكاء الاصطناعي في حل مشكلات حقيقية أو تحسين ممارسات يومية بطريقة ابتكارية.

سجل كيف ستوظف المشروع الخاص بك في الحياة العملية:



مهارات المشروع

يهدف المشـــروع إلى تنمية مهارات المتعلمين التقنية مثل التصميـــم والبرمجة والذكاء الاصطناعي، إلى جانب المهارات الشـــخصية كالتعـــاون وحل المشـــكلات، وذلك من خلال تنفيـــذ فكرة عملية لإكســـابهم خبرات واقعية تعزز جاهزيتهم للمســـتقبل، وتســـاعدهم على اكتســـاب كفاءات تقنية تعزز قدرتهم على الابتـــكار والتطبيق العملي.

حدد المهارات المستخدمة بالمشروع:

	المجال	المهارات الفرعية المستهدفة
	المهارات الرقمية والتقنية	القدرة على كتابة أو تعديل أكواد برمجية بلغة Python.
		تقنيات الذكاء الاصطناعي وتعلم الآلة.
	التفكير الناقد وحل المشكلات	تحليل المشكلة وتحديد عناصرها.
		تجريب الحلول واختيار الأنسب.
		التحقق من دقة النتائج وتعديل الأخطاء.
	مهارات التوثيق والبحث العلمي	تصوير مراحل تنفيذ المشروع وتسجيلها.
		جمع معلومات من مصادر موثوقة.
		كتابة المراجع العلمية.
	مهارات العمل الجماعي	التعاون وتقسيم المهام.
		تحمل المسؤولية والالتزام بالوقت.
		اتخاذ قرارات جماعية.
	مهارات العرض والتواصل	إعداد عرض بصري فعّال (شرائح، فيديو، أدلة، تقارير).
		شرح الفكرة بلغة واضحة ومقنعة.
		التفاعل مع الجمهور والرد على الأسئلة.

خطوات تنفيذ المشروع

1. إعداد بيئة العمل

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

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

و أي مستلزمات أخرى تخدم المشروع.

2. إعداد هيكل المشروع

يُفضل الاستغلال الأمثل للمهارات البرمجية التالية:

- استخدام لغة Python.
- استخدام أنواع متعددة من المتغيرات.
- استخدام التعليمات البرمجية للشروط Conditions.
- استخدام التعليمات البرمجية للتكرار Loops.
- معالجة الأخطاء باستخدام الاستثناءات try/ except.
- توظيف هياكل البيانات غير الأولية Non-Primitive Data Structures، مثل Lists، Dictionaries، Tubles.
- توظيف الحزم البرمجية مثل حزم الذكاء الاصطناعي لتطوير البرنامج.

3. كتابة التعليمات البرمجية

يكتب المبرمج التعليمات البرمجية استناداً لما تم تخطيطه سابقاً.

4. اختبار المشروع

يختبر المبرمج البرنامج، ويتأكد من خلوه من الأخطاء البرمجية، اللغوية، والعلمية.

5. تطوير المشروع

يُبسط المبرمج ويرتب التعليمات البرمجية إذا أصبحت صعبة القراءة، وإضافة التعليقات والبيانات الإرشادية التي توضح مهمة التعليمات البرمجية.

6. توثيق المشروع

يُنشئ مصمم العرض التقديمي عرضًا شاملاً للجوانب التالية:

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

و تسليم مجلد مجمع لجميع ملفات المشروع.

7. مشاركة المشروع

عرض الفريق للمشروع، ومناقشته مع المعلم والمتعلمين.

8. الاستمرار في التعلم

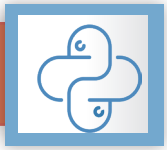
التوسع في الاطلاع على مستجدات الذكاء الاصطناعي وتنمية قدرات المتعلم البرمجية من خلال الدورات التدريبية والمنتديات الحاسوبية.

التوثيق العلمي للمشروع

- المراجع.
  - الالتزام بطريقة علمية لتوثيق المراجع على سبيل المثال نظام APA[1].
- عرض الفريق للمشروع ومناقشته مع المعلم والمتعلمين، مع مشاركته على منصة Teams.
- الاستمرار في التعلم.

مشروعات إثرائية

1 اختصار لعبارة American Psychological Association
ويُعد أحد أشهر أنظمة التوثيق المرجعي المستخدمة عالميًا، خاصةً في الأبحاث العلمية، والعلوم الاجتماعية، والنفسية، والتربوية، والبحث الأكاديمي عمومًا.



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.

- Basu, S. (2021, May 20). Learn SQLite with Python in 24 Hours: Simple, Concise & Easy Guide to Using Database with Python. Independently published.
- Kurniawan, A. (2021, January 30). Python and SQLite Development. PE Press.
- Siahaan, V. S., & Sianipar, R. H. (2025). Learn SQLite with Python: Building Database-Driven DesKtop Projects. Independently published.
- Hayes, W. (2025, 1 فبراير). Master Computer Vision and AI with OpenCV and Python: UnlocK Cutting-Edge Tools, Techniques, and Real-World Applications for Image Processing and Machine Learning. Independently published.
- Howse, J., & Minichino, J. (2020, 20 فبراير). Learning OpenCV 4 Computer Vision with Python 3 (3rd ed.). PacKt Publishing.
- Mugesh, S. (2023). Hands-on ML Projects with OpenCV: Master computer vision and Machine Learning using OpenCV and Python. Independently published.

10