تُستخدم البرمجة القائمة على الأحداث بشكل شائع في تطبيقات واجهة المستخدم الرسومية (GUI) ، حيث تؤدي إجراءات المستخدم مثل نقرات الأزرار أو تحديدات القائمة إلى إجراءات أو تحديثات محددة على الشاشة. يسمح لواجهات المستخدم التفاعلية والمتجاوبة.

تتضمن بعض المكتبات وأطر البرمجة الشائعة التي تعتمد على الأحداث ما يلي:

1. JavaScript مع أحداث DOM (نموذج كائن المستند) لتطوير الويب.

2. Java Swing وJavaFX لتطوير تطبيقات سطح المكتب.

3. C # مع Windows Forms أو WPF لتطوير تطبيقات Windows.

4. Python مع مكتبات مثل Tkinter أو PyQt لتطوير واجهة المستخدم الرسومية.

5. Node.js بهندسته التي تعتمد على الأحداث لتطوير JavaScript من جانب الخادم.

بشكل عام ، تسمح البرمجة القائمة على الأحداث للمطورين بإنشاء تطبيقات يمكنها الاستجابة ديناميكيًا لتفاعلات المستخدم والأحداث الخارجية ، مما يوفر تجربة مستخدم تفاعلية وجذابة

Event-driven programming is a programming paradigm in which the flow of the program is determined by events or user actions rather than being driven by a sequential set of instructions. In event-driven programming, the program responds to events such as button clicks, mouse movements, or keyboard inputs.

The key concept in event-driven programming is the event loop. The program continuously waits for events to occur and then triggers appropriate event handlers or callbacks to respond to those events. Event handlers are functions or methods that are executed when a specific event occurs.

Event-driven programming is commonly used in graphical user interface (GUI) applications, where user actions like button clicks or menu selections trigger specific actions or updates on the screen. It allows for interactive and responsive user interfaces.

Some popular event-driven programming frameworks and libraries include:

1. JavaScript with its DOM (Document Object Model) events for web development.

2. Java Swing and JavaFX for desktop application development.

3. C# with Windows Forms or WPF for Windows application development.

4. Python with libraries like Tkinter or PyQt for GUI development.

5. Node.js with its event-driven architecture for server-side JavaScript development.

Overall, event-driven programming allows developers to create applications that can respond dynamically to user interactions and external events, providing a more interactive and engaging user experience.

Daily quota: 6/10

MAQ:

كود

يحسب محيط الدائرة

ChatGPT DALL-E Bot:

لحساب محيط الدائرة، يمكن استخدام القانون التالي:

محيط الدائرة = 2 × ط × نصف قطر الدائرة

وفقًا لذلك، يمكن كتابة الكود التالي لحساب محيط الدائرة في لغة برمجة Python:

```python

import math

def calculate\_circle\_circumference(radius):

circumference = 2 \* math.pi \* radius

return circumference

# اختبار الكود

radius = float(input("أدخل نصف قطر الدائرة: "))

circumference = calculate\_circle\_circumference(radius)

print("محيط الدائرة هو:", circumference)

```

عند تشغيل هذا الكود، سيُطلب من المستخدم إدخال قيمة نصف قطر الدائرة، ثم سيتم حساب وطباعة محيط الدائرة.

Daily quota: 9/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

كود يحسب مساحة الدائرة

ChatGPT DALL-E Bot:

لحساب مساحة الدائرة، يمكن استخدام القانون التالي:

مساحة الدائرة = نصف قطر الدائرة \* نصف قطر الدائرة \* ط

وهنا هو كود Python لحساب مساحة الدائرة:

```python

import math

def calculate\_area(radius):

area = math.pi \* radius\*\*2

return area

radius = float(input("أدخل قيمة نصف قطر الدائرة: "))

circle\_area = calculate\_area(radius)

print("مساحة الدائرة هي:", circle\_area)

```

عند تشغيل هذا الكود، ستُطلب من المستخدم إدخال قيمة نصف قطر الدائرة. ثم ستقوم البرنامج بحساب مساحة الدائرة وطباعتها على الشاشة.

Daily quota: 8/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

كود يحسب مساحة المثلث

ChatGPT DALL-E Bot:

لحساب مساحة المثلث، يمكن استخدام القانون التالي:

مساحة المثلث = (قاعدة المثلث \* الارتفاع) / 2

وفيما يلي كود Python يستخدم هذا القانون لحساب مساحة المثلث:

```python

base = float(input("أدخل قاعدة المثلث: "))

height = float(input("أدخل الارتفاع: "))

area = (base \* height) / 2

print("مساحة المثلث هي:", area)

```

بعد تشغيل البرنامج، ستظهر رسالة تطلب منك إدخال قيمة قاعدة المثلث والارتفاع. بعدها، ستظهر رسالة تعرض مساحة المثلث.

Daily quota: 7/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

كود يحسب محيط المثلث

ChatGPT DALL-E Bot:

يمكنك استخدام الكود التالي لحساب محيط المثلث:

```python

a = float(input("أدخل طول الضلع الأول: "))

b = float(input("أدخل طول الضلع الثاني: "))

c = float(input("أدخل طول الضلع الثالث: "))

perimeter = a + b + c

print("محيط المثلث هو:", perimeter)

```

في هذا الكود، نقوم بطلب إدخال أطوال أضلاع المثلث من المستخدم باستخدام دالة `input()` ونقوم بتحويلها إلى أعداد عشرية باستخدام `float()`. ثم نقوم بحساب مجموع أطوال الأضلاع ونقوم بطباعة قيمة المحيط.

Daily quota: 6/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

كود يحسب محيط المستطيل

ChatGPT DALL-E Bot:

يمكنك استخدام الكود التالي لحساب محيط المستطيل:

```python

def calculate\_perimeter(length, width):

perimeter = 2 \* (length + width)

return perimeter

length = float(input("أدخل طول المستطيل: "))

width = float(input("أدخل عرض المستطيل: "))

perimeter = calculate\_perimeter(length, width)

print("محيط المستطيل هو:", perimeter)

```

عند تشغيل البرنامج، ستظهر رسالة تطلب من المستخدم إدخال طول وعرض المستطيل. بعد ذلك، ستقوم الدالة `calculate\_perimeter` بحساب محيط المستطيل باستخدام الصيغة `2 \* (length + width)` وتعيدها كقيمة. أخيرًا، يتم طباعة قيمة محيط المستطيل على الشاشة.

Daily quota: 5/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

كود يحسب مساحة المستطيل

ChatGPT DALL-E Bot:

يمكنك استخدام الكود التالي لحساب مساحة المستطيل:

```python

# استلام طول وعرض المستطيل من المستخدم

length = float(input("أدخل طول المستطيل: "))

width = float(input("أدخل عرض المستطيل: "))

# حساب مساحة المستطيل

area = length \* width

# طباعة نتيجة مساحة المستطيل

print("مساحة المستطيل هي:", area)

```

عند تشغيل البرنامج، ستظهر رسالة تطلب من المستخدم إدخال طول وعرض المستطيل. بعد إدخال القيم، ستقوم البرنامج بحساب مساحة المستطيل وإظهارها في النهاية.

Daily quota: 4/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

كود يحسب محيط المربع

ChatGPT DALL-E Bot:

يمكنك استخدام الكود التالي لحساب محيط المربع:

```python

def calculate\_perimeter(side\_length):

perimeter = 4 \* side\_length

return perimeter

side\_length = float(input("أدخل طول ضلع المربع: "))

perimeter = calculate\_perimeter(side\_length)

print("محيط المربع هو:", perimeter)

```

في هذا الكود، يتم تعريف دالة `calculate\_perimeter` التي تأخذ طول ضلع المربع كمدخل وتقوم بحساب محيط المربع باستخدام الصيغة `4 \* side\_length`. ثم يتم استدعاء الدالة وتخزين ناتجها في متغير `perimeter`، وأخيرًا يتم طباعة قيمة محيط المربع.

Daily quota: 9/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

كود يحسب مساحة المربع

ChatGPT DALL-E Bot:

يمكنك استخدام الكود التالي لحساب مساحة المربع:

```python

# استلام طول ضلع المربع من المستخدم

side = float(input("أدخل طول ضلع المربع: "))

# حساب مساحة المربع

area = side \*\* 2

# طباعة نتيجة مساحة المربع

print("مساحة المربع هي:", area)

```

في هذا الكود، نقوم بطلب من المستخدم إدخال طول ضلع المربع باستخدام دالة `input()` ونقوم بتحويل القيمة إلى عدد عشري باستخدام `float()`. ثم نقوم بحساب مساحة المربع بتربيع قيمة الضلع ونقوم بطباعتها باستخدام دالة `print()`.

Daily quota: 8/10

ENG | ES | عربي | 中文 | فارسی

MAQ:

قانون محيط المربع ومساحة المربع

ChatGPT DALL-E Bot:

قانون محيط المربع:

محيط المربع يساوي ضعف طول أحد أضلاعه.

إذاً، محيط المربع = 4 × طول الضلع

قانون مساحة المربع:

مساحة المربع تُحسب بضرب طول أحد أضلاعه في نفسه.

إذاً، مساحة المربع = طول الضلع × طول الضلع

Daily quota: 7/10

ENG | ES | عربي | 中文 | فارسی

class MathOperations:

@staticmethod

def add(x, y):

return x + y

MathOperations.add(8,4)

الـ Class Method والـ Static Method هما نوعان من الوظائف (أو الأساليب) في البرمجة الشيئية، وهما يُستخدمان في الصفوف (Classes) في اللغات التي تدعم هذا المفهوم مثل Python و Java وغيرها. إليك الفرق بينهما:

1. الـ Class Method (الوظيفة التابعة للصف):

- تُعرف باستخدام مُشتقة (decorator) خاصة تسمى "@classmethod".

- تتعامل مع الصف نفسه وتمتلك إمكانية الوصول إلى الخصائص الساكنة (static properties) والوظائف الأخرى في الصف.

- تأخذ دائمًا المعامل الأول كمرجع إلى الصف نفسه، غالبًا يُسمى "cls".

- يمكن استخدامها لإنشاء وتعديل خصائص تخص الصف بشكل عام.

مثال في Python:

class MyClass:

class\_variable = 0

@classmethod

def increment\_variable(cls):

cls.class\_variable += 1

2. الـ Static Method (الوظيفة الثابتة):

- تُعرف باستخدام مُشتقة (decorator) خاصة تسمى "@staticmethod".

- ليست مُرتبطة بأي معاملات خاصة بالصف أو بالمثيلات (instances) وتتعامل فقط مع المدخلات التي تُمر إليها.

- لا تتيح لها الوصول إلى الخصائص الساكنة أو الوظائف الأخرى في الصف.

مثال في Python:

class MathOperations:

@staticmethod

def add(x, y):

return x + y