بايثون

عن طريق الامثلة

٩٠ مشروع بايثون مع الكود المصدري تم حلها وشرحها

ترجمة واعداد: د. علاء طعيمة

بـــمه تعالى

بايثون: عن طريق الامثلة

90 مشروع بايثون مع الكود المصدري تم حلها وشرحها

ترجمة واعداد:

د. علاء طعيمة

المقدمة

بايثون هي واحدة من أفضل لغات البرمجة. نظرًا لقابليتها للقراءة وطبيعتها الصديقة للمبتدئين، فقد تم قبولها من قبل الصناعاتفي جميع أنحاء العالم. لذلك لإتقان بايثونفي أي مجال، عليك العملفي المشاريع.في هذه الكتاب، سيقدم لك المؤلف أكثر من 90 مشروع مذهلفي بايثون مع حل الكود المصدري وشرحها مجانًا.

إذا كنت مبتدئًافي بايثون حيث تعلمت للتو القوائم (list) والصفوف (tuples) والقواميس (dictionaries) وبعض الوحدات النمطية (modules) لبايثون الأساسية مثل الوحدة العشوائية (random)، فإليك بعض مشاريع بايثون ذات التعليمات البرمجية المصدر للمبتدئين.

لقد حاولت قدر المستطاع ان اترجم المشاريع الأكثر طرحاًفي مجال تعلم بايثون مع الشرح المناسب والكافي، ومع هذا يبقى عملاً بشرياً يحتمل النقص، فاذاكان لديك أي ملاحظات حول هذا الكتاب، فلا تتردد بمراسلتنا عبر بريدنا الالكتروني alaa.taima@qu.edu.iq .

نأمل ان يساعد هذا الكتاب كل من يريد ان يدخلفي مجال لغة برمجة بايثون ومساعدة القارئ العربي على تعلم هذا المجال. اسأل الله التوفيقفي هذا العمل لأثراء المحتوى العربيفي مجال البرمجة ببايثون. ونرجو لك الاستمتاع مع الكتاب ولا تنسونا من صالح الدعاء.

د. علاء طعيمة

كلية علوم الحاسوب وتكنولوجيا المعلومات

جامعة القادسية

العراق

المحتويات

1) لعبة التخمين باستخدام بايثون Number Guessing Game using Python

لعبة التخمين بالأرقام (Number Guessing Game) هي لعبة شائعة بين المبرمجين.في لعبة التخمين بالأرقام، يختار البرنامج رقمًا عشوائيًا بين رقمين، ويخمن المستخدم الرقم الصحيح. إذا كنت تريد معرفة كيفية إنشاء لعبة تخمين باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول إنشاء لعبة تخمين الأرقام باستخدام لغة برمجة بايثون.

لعبة التخمين باستخدام بايثون

لإنشاء لعبة تخمين، نحتاج إلى كتابة برنامج لتحديد رقم عشوائي بين 1 و 10. لإعطاء تلميحات للمستخدم، يمكننا استخدام العبارات الشرطية (conditional statements) لإخبار المستخدم إذا كان الرقم الذي تم تخمينه أصغر أو أكبر من أو يساوي الرقم المختار عشوائيًا.

فيما يلي كيفية كتابة برنامج لإنشاء لعبة تخمين الأرقام باستخدام بايثون:

```
import random
n = random.randrange(1,10)
guess = int(input("Enter any number: "))
while n!= guess:
    if guess < n:
        print("Too low")
        guess = int(input("Enter number again: "))
    elif guess > n:
        print("Too high!")
        guess = int(input("Enter number again: "))
    else:
      break
print("you guessed it right!!")
```

```
Enter any number: 2
Too low
Enter number again: 5
Too low
Enter number again: 8
you guessed it right!!
```

إذا كان الرقم الذي تم تخمينه أقل من الرقم المحدد عشوائيًا، فسيرى المستخدم "منخفض جداً Too low". إذا كان الرقم الذي تم تخمينه أعلى من الرقم المحدد عشوائيًا، فسيظهر للمستخدم "مرتفع جداً Too high " . عندما يخمن المستخدم الرقم الصحيح، "لقد خمنته بشكل صحيح you guessed it right!! " سيتم عرضهافي الإخراج.

الملخص

هذه هي الطريقة التي يمكنك بها كتابة برنامج لإنشاء لعبة تخمين باستخدام بايثون. إنها لعبة شائعة بين المبرمجين.في هذه اللعبة، يختار البرنامج رقمًا عشوائيًا بين رقمين، ويخمن المستخدم الرقم الصحيح. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء لعبة تخمين باستخدام بايثون.

2) المتوسط والوسيط والمنوال باستخدام بايثون Mean Median and Mode using Python

المتوسط (Mean) والوسيط (Median) والمنوال (Mode) هي أساسيات الإحصاء المستخدمةفي كل مجال تقريبًا حيث نتعامل مع الأرقام. بايثون هي واحدة من أفضل لغات البرمجة للحسابات الرقمية. لذلك يجب أن تعرف كيفية حساب المتوسط والوسيط والمنوال باستخدام بايثون دون استخدام أي مكتبة أو وحدة بايثون مضمنة. لذلكفي هذه المقالة، سوف أتعرف على كيفية حساب المتوسط والوسيط والمنوال باستخدام بايثون.

المتوسط والوسيط والمنوال باستخدام بايثون

المتوسط

المتوسط هو متوسط قيمة جميع القيمفي مجموعة البيانات. لحساب القيمة المتوسطة لمجموعة البيانات، نحتاج أولاً إلى إيجاد مجموع كل القيم ثم قسمة مجموع كل القيم على العدد الإجمالي للقيم. إليك كيفية حساب المتوسط باستخدام بايثون:

```
# Mean
list1 = [12, 16, 20, 20, 12, 30, 25, 23, 24, 20]
mean = sum(list1)/len(list1)
print(mean)
```

20.2

الوسيط

الوسيط هو القيمة الوسطى بين جميع القيم بالترتيب. نحتاج هنا إلى حساب القيمة المتوسطة لجميع القيمفي مجموعة البيانات. لكن قبل حساب الوسيط، نحتاج إلى فرز كل القيم بالترتيب. هناك طريقتان مختلفتان لحساب القيمة المتوسطة:

1. عندما يكون العدد الإجمالي للقيم زوجيًا:

$$Median = [(n/2)^{th}\ term + \{(n/2)+1\}^{th}]/2$$

2. عندما يكون العدد الإجمالي للقيم فرديًا

$$Median = \{(n+1)/2\}^{th} term$$

الآن فيما يلي كيف يمكنك حساب الوسيط باستخدام بايثون:

```
# Median
list1 = [12, 16, 20, 20, 12, 30, 25, 23, 24, 20]
list1.sort()

if len(list1) % 2 == 0:
```

```
    m1 = list1[len(list1)//2]
    m2 = list1[len(list1)//2 - 1]
    median = (m1 + m2)/2
else:
    median = list1[len(list1)//2]
print(median)
```

```
20.0
```

المنوال

المنوال هو القيمة الأكثر تكرارا بين جميع القيم. فيما يلي كيفية حساب قيمة المنوال لمجموعة البيانات باستخدام بايثون:

```
# Mode
list1 = [12, 16, 20, 20, 12, 30, 25, 23, 24, 20]
frequency = {}
for i in list1:
    frequency.setdefault(i, 0)
    frequency[i]+=1

frequent = max(frequency.values())
for i, j in frequency.items():
    if j == frequent:
        mode = i
print(mode)
```

```
20
```

الملخص

هذه هي الطريقة التي يمكنك بها حساب المتوسط والوسيط والمنوال باستخدام بايثون دون استخدام أي مكتبة أو أي وحدة بايثون مضمنةفي ثناياه عوامل. بايثون هي واحدة من أفضل لغات البرمجة للحسابات الرقمية. لذلك يجب أن تعرف كيفية حساب المتوسط والوسيط والمنوال باستخدام بايثون دون استخدام أي مكتبة أو وحدة بايثون مضمنة. أتمنى أن تكون قد أحببت هذه المقالة حول حساب المتوسط والوسيط والمنوال باستخدام بايثون.

3) مصادقة كلمة المرور باستخدام بايثون Password Authentication using Python

مصادقة كلمة المرور (Password Authentication) هي عملية التحقق من هوية المستخدم. تتأكد كل منصة على الإنترنت تقريبًا اليوم من أنها تمنح حق الوصول إلى المستخدم الحقيقي فقط والذي لا يمكن أن يكون ممكنًا إلا عن طريق طلب كلمة مرور بينما يريد المستخدم تسجيل الدخول إلى الحساب. لذلكفي هذه المقالة، سوف آخذك خلال مهمة مصادقة كلمة المرور باستخدام بايثون.

ما هو نظام مصادقة كلمة المرور؟

نظام مصادقة كلمة المرور هو نظام يُستخدم لتحديد هوية المستخدم. فكرفي الأمر مثل شاشة تسجيل الدخول التي تراها أثناء تسجيل الدخول إلى حسابك على Facebook. يسألك عن بريدك الإلكتروني أو اسم مستخدم ثم يطلب كلمة المرور الخاصة بك. إذا أدخلت كلمة المرور الصحيحة، فستتحقق من أنك المستخدم الحقيقي.

يعد إنشاء نظام مصادقة كلمات المرور المستند إلى المنطق سؤالًا شائعًا أيضًافي مقابلات البرمجة. لذلك،في القسم أدناه، سوف أطلعك على كيفية إنشاء نظام مصادقة كلمة المرور باستخدام بايثون.

مصادقة كلمة المرور باستخدام بايثون

لإنشاء نظام مصادقة كلمة المرور باستخدام بايثون، عليك اتباع الخطوات المذكورة أدناه:

3. أنشئ قاموسًا لأسماء المستخدمين باستخدام كلمات المرور الخاصة بهم.
4. ثم عليك أن تطلب إدخال المستخدم كاسم مستخدم باستخدام دالة الإدخالفي بايثون.
5. ثم يتعين عليك استخدام وحدة (getpass)في بايثون للمطالبة بإدخال المستخدم ككلمة المرور. نحن هنا نستخدم وحدة (getpass) بدلاً من دالة الإدخال للتأكد من عدم تمكن المستخدم من رؤية ما يكتبهفي حقل كلمة المرور.

لذلك دعونا نتبع الخطوات المذكورة أعلاه لإنشاء نظام مصادقة كلمة المرور باستخدام بايثون:

```
import getpass
database = {"aman.kharwal": "123456", "kharwal.aman":
"654321"}
username = input("Enter Your Username : ")
password = getpass.getpass("Enter Your Password : ")
for i in database.keys():
    if username == i:
        while password != database.get(i):
```

```
            password = getpass.getpass("Enter Your Password
Again : ")
        break
print("Verified")
```

```
Enter Your Username : aman.kharwal
Enter Your Password : ·····
Enter Your Password Again : ·····
Enter Your Password Again : ·····
Verified
```

الملخص

هذه هي الطريقة التي يمكننا بها مصادقة هوية المستخدم باستخدام لغة برمجة بايثون. يمكنك الآن تجربة نفس المنطق مع المزيد من أسماء المستخدمين وهياكل البيانات الأخرى أيضًا. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء نظام مصادقة كلمة المرور باستخدام بايثون.

4) إرسال رسائل بريد إلكتروني تلقائية باستخدام بايثون
Send Automatic Emails using Python

في كل مرة تقوم فيها بالتسجيل في تطبيق جديد، تتلقى تلقائيًا رسالة ترحيب بها اسمك. إذا كنت تريد معرفة كيفية إرسال رسائل البريد الإلكتروني هذه تلقائيًا، فهذه المقالة مناسبة لك. في هذه المقالة، سوف أطلعك على كيفية إرسال رسائل بريد إلكتروني تلقائية باستخدام بايثون.

كيف ترسل رسائل بريد إلكتروني تلقائية باستخدام بايثون؟

لإرسال رسائل بريد إلكتروني تلقائية باستخدام بايثون، يجب أن تفهم أولاً كيفية إرسال بريد إلكتروني باستخدام لغة برمجة بايثون. بمجرد معرفة كيفية إرسال بريد إلكتروني باستخدام بايثون، فإن الشيء التالي الذي تحتاج إلى اكتشافه هو ما تريد إرساله تلقائيًا. على سبيل المثال، ترسل العديد من الشركات OTP أو رسائل ترحيب، بينما ترسل بعض الشركات رسائل إخبارية إلى المستخدمين المسجلين حديثًا.

لذلك في القسم أدناه، سوف أطلعك على كيفية إرسال رسائل البريد الإلكتروني التلقائية باستخدام بايثون. سأرسل تلقائيًا رسالة ترحيب إلى المستخدم المسجل حديثًا. لهذه المهمة، يجب عليك أولاً إنشاء كلمة مرور تطبيقات Google لحساب Gmail الخاص بك.

إرسال رسائل بريد إلكتروني تلقائية باستخدام بايثون

من المهم جدًا إنشاء كلمة مرور تطبيقات Google لحساب Gmail الخاص بك، حيث سترسل رسائل بريد إلكتروني تلقائية باستخدام بايثون من خلال حساب Gmail الخاص بك. بمجرد إنشاء كلمة مرور تطبيق Google، إليك كيفية بدء مهمة إرسال رسائل البريد الإلكتروني باستخدام بايثون:

```
import os
import random
import smtplib


def automatic_email():
    user = input("Enter Your Name >>: ")
    email = input("Enter Your Email >>: ")
    message = (f"Dear {user}, Welcome to Thecleverprogrammer")
    s = smtplib.SMTP('smtp.gmail.com', 587)
    s.starttls()
    s.login("Your Gmail Account", "Your App Password")
    s.sendmail('&&&&&&&&&&', email, message)
    print("Email Sent!")

automatic_email()
```

في الكود أعلاه، حددت دالة بايثون التي سترسل تلقائيًا رسائل بريد إلكتروني إلى مستخدم مسجل حديثًا. ستبدأ هذه الدالة بالسؤال عن اسم المستخدم والبريد الإلكتروني. ثم سيتم تخزين اسم المستخدمفي الرسالة. ثمفي السطر الثاني عشر من الكود أعلاه، تحتاج إلى استبدال المعلمة الأولى ببريدك الإلكتروني والمعلمة الثانية بكلمة مرور تطبيقات Google التي أنشأتها من قبل. بعد ذلك فقط قم بتشغيل الكود أعلاه وسترى الإخراج كما هو موضح أدناه.

```
Enter Your Name >>: Aman Kharwal
Enter Your Email >>: support@thecleverprogrammer.com
Email Sent!
```

بعد ملء التفاصيل لإدخالات المستخدم هذه، تلقيت بريدًا إلكترونيًافي صندوق الوارد الخاص بي كما هو موضح أدناه.

```
Dear Aman Kharwal, Welcome to Thecleverprogrammer
```

الملخص

هذه هي الطريقة التي يمكنك بها إرسال رسائل بريد إلكتروني تلقائية باستخدام بايثون. لإرسال رسائل بريد إلكتروني تلقائية باستخدام بايثون، يجب أن تفهم أولاً كيفية إرسال بريد إلكتروني باستخدام لغة برمجة بايثون. بمجرد معرفة كيفية إرسال بريد إلكتروني باستخدام بايثون، فإن الشيء التالي الذي تحتاج إلى اكتشافه هو ما تريد إرساله تلقائيًا. آمل أن تكون قد أحببت هذه المقالات حول كيفية إرسال رسائل البريد الإلكتروني تلقائيًا باستخدام بايثون.

5) آلة حساب العمر باستخدام بايثون Age Calculator using Python

آلة حاسبة العمر (Age Calculator) هي فكرة مشروع برمجة مذهلة للمبتدئين. إذا كنت جديدًا على أي لغة برمجة، فيجب أن تحاول إنشاء آلة حاسبة للعمر. هو تطبيق يقوم فيه المستخدم بإدخال تاريخ ميلاده كمدخل، ويعطي التطبيق عمره كمخرج. لذلك، إذا كنت تريد معرفة كيفية إنشاء آلة حاسبة للعمر باستخدام لغة برمجة بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم لك برنامجًا تعليميًا حول كيفية إنشاء آلة حاسبة للعمر باستخدام بايثون.

آلة حاسبة العمر باستخدام بايثون

حساب العمر هو تطبيق رائع يمكنك إنشاؤه كمبتدئفي أي لغة برمجة. لإنشاء آلة حاسبة للعمر، تحتاج إلى تاريخين:

1. تاريخ اليوم.
2. تاريخ الولادة.

يمكنك إما أن تطلب من المستخدم كلا التاريخين أو أن تطلب فقط تاريخ الميلاد واستخدام تاريخ اليوم من الكمبيوتر نفسه. يبدو طلب عيد الميلاد خيارًا أكثر سهولةفي الاستخدام. إليك كيفية إنشاء آلة حاسبة للعمر باستخدام بايثون:

```
def ageCalculator(y, m, d):
    import datetime
    today = datetime.datetime.now().date()
    dob = datetime.date(y, m, d)
    age = int((today-dob).days / 365.25)
    print(age)
ageCalculator(1998, 9, 3)
```

في الكود أعلاه:

1. لقد حددت لأول مرة دالة بايثون حيث أطلب ثلاث مدخلات للمستخدم:
   - y: سنة الميلاد.
   - m: شهر الميلاد.
   - d: تاريخ الميلاد.
2. ثم أقوم باستيراد وحدة التاريخ والوقتفي بايثون داخل الدالة.
3. ثمفي السطر التالي، سآخذ تاريخ اليوم باستخدام طريقة datetime.now() لوحدة التاريخ والوقت.

4. ثم أدخلت متغيرًا جديدًافي السطر التالي باسم dob، حيث أستخدم تاريخ الميلاد كمدخل قدمه المستخدم.
5. ثم أقوم بطرح dob بتاريخ اليوم ثم أقسمه على 365.25 والذي يعيد عمر المستخدم.

الملخص

حساب العمر هي فكرة مشروع برمجة مذهلة للمبتدئين. هو تطبيق يقوم فيه المستخدم بإدخال تاريخ ميلاده كمدخل، ويعطي التطبيق عمره كمخرج. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء آلة حاسبة للعمر باستخدام بايثون.

6) مجموعة الجناس الناقصة باستخدام بايثون Group Anagrams using Python

الجناس الناقصة (Anagrams) هي كلمات تتكون من إعادة ترتيب حروف كلمة أخرى، على سبيل المثال، سيارة وقوس، قطة وفعل، إلخ. تجميع الجناس الناقصة هو أحد الأسئلة الشائعةفي مقابلات البرمجة. لذلك إذا كنت تريد معرفة كيفية حل مشكلة تجميع الجناس الناقصة، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول كيفية تجميع الجناس الناقصة باستخدام بايثون.

مجموعة الجناس الناقصة باستخدام بايثون

يعد تجميع الجناس الناقصة أحد الأسئلة الشائعةفي مقابلات البرمجة. هنا سوف تحصل على قائمة بالكلمات، وعليك أن تكتب خوارزمية لتجميع كل الكلمات التي تمثل الجناس الناقص لبعضها البعض. فيما يلي كيفية كتابة دالة بايثون لتجميع الجناس الناقصة:

```
from collections import defaultdict

def group_anagrams(a):
    dfdict = defaultdict(list)
    for i in a:
        sorted_i = " ".join(sorted(i))
        dfdict[sorted_i].append(i)
    return dfdict.values()
```

دعنا الآن نختبر الدالة عن طريق إنشاء قائمة بالكلمات التي تحتوي على الجناس الناقصة وبعض الكلمات الأخرى:

```
words = ["tea", "eat", "bat", "ate", "arc", "car"]
print(group_anagrams(words))
```

```
dict_values([['tea', 'eat', 'ate'], ['bat'], ['arc', 'car']])
```

هذه هي الطريقة التي يمكنك بها تجميع الجناس الناقصة باستخدام لغة برمجة بايثون. من المفيد ممارسة مقابلات البرمجة لتحسين منطق البرمجة ومهارات البرمجة. يمكنك التدرب على المزيد من أسئلة مقابلات البرمجة ومشاريع بايثون من هنا.

الملخص

يعد تجميع الجناس الناقصة أحد الأسئلة الشائعةفي مقابلات البرمجة. هنا عليك أن تكتب خوارزمية لتجميع كل الكلمات التي تمثل الجناس الناقص لبعضها البعض. آمل أن تكون قد أحببت هذه المقالة حول تجميع الجناس الناقصة باستخدام بايثون.

7) البحث عن رقم مفقود باستخدام بايثون Finding the missing Number using Python

يعد العثور على الرقم المفقودفي المصفوفة (array) سؤالاً شائعاًفي مقابلة الترميز. وفقًا لـ LeetCode، هذا السؤال شائعفي المقابلات مع شركات مثل Amazon وAdobe وMicrosoft وLinkedIn وغيرها الكثير. إذا كنت تريد معرفة كيفية العثور على الرقم المفقود في المصفوفة، فهذه المقالة مناسبة لك. ستطلعك هذه المقالة على كيفية العثور على الرقم المفقود باستخدام بايثون.

البحث عن رقم مفقود باستخدام بايثون

إيجاد العدد المفقودفي المصفوفة يعني إيجاد الأعداد المفقودة من المصفوفة وفقًا لمدى القيم داخل المصفوفة.في معظم الأحيان، يكون السؤال الذي تحصل عليه بناءً على هذه المشكلة مثل:

> بالنظر إلى مصفوفة تحتوي على نطاق من الأرقام من 0 إلى n مع عدد مفقود، أوجد العدد المفقودفي مصفوفة الإدخال.

للعثور على العدد المفقودفي المصفوفة، نحتاج إلى التكرار على مصفوفة الإدخال وتخزين الأرقام في مصفوفة أخرى لم نجدهافي مصفوفة الإدخال أثناء التكرار عليها. فيما يلي كيفية العثور على الرقم المفقودفي مصفوفة أو قائمة باستخدام لغة برمجة بايثون:

```
def findMissingNumbers(n):
    numbers = set(n)
    length = len(n)
    output = []
    for i in range(1, n[-1]):
        if i not in numbers:
            output.append(i)
    return output

listOfNumbers = [1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 13, 14, 16]
print(findMissingNumbers(listOfNumbers))
```

Output: [4, 12, 15]

هذه هي الطريقة التي يمكنك بها العثور على الأرقام المفقودةفي مصفوفة أو قائمة باستخدام لغة برمجة بايثون. يمكنك العثور على المزيد من أسئلة مقابلة البرمجة ومشاريع بايثون هنا.

الملخص

يعد العثور على الرقم المفقودفي المصفوفة سؤالاً شائعاًفي مقابلة البرمجة. للعثور على العدد المفقودفي المصفوفة، نحتاج إلى التكرار على مصفوفة الإدخال وتخزين الأرقامفي مصفوفة أخرى

لم نجدهافي مصفوفة الإدخال أثناء التكرار عليها. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية العثور على الرقم المفقودفي مصفوفة أو قائمة باستخدام بايثون.

8) تجميع عناصر من نفس المؤشرات باستخدام بايثون
Group Elements of Same Indices using Python

يعني تجميع عناصر نفس المؤشرات (Grouping elements of the same indices) تجميع عناصر هيكلين أو أكثر من البيانات وفقًا لمؤشراتها (indices). إنها مشكلة صعبة للمبتدئين ويمكن طرحهافي أي مقابلة برمجة. إذا كنت تريد معرفة المزيد حول تجميع العناصر من نفس الفهرس، فهذه المقالة مناسبة لك.في هذه المقالة، سأوجهك خلال برنامج تعليمي حول كيفية تجميع عناصر من نفس المؤشرات باستخدام بايثون.

تجميع عناصر من نفس المؤشرات باستخدام بايثون

لتجميع عناصر من نفس الفهرس، سيكون لديكفي البداية قائمتان أو أكثر داخل قائمة مثل [[a, b]،[c, d]] . لتجميع عناصر هذه القوائم، تحتاج إلى إنشاء قائمتين جديدتين حيث ستخزن عناصر كلتا القائمتينفي الفهرس [a, c] 0 والفهرس [b, d] 1. هذا هو معنى تجميع عناصر نفس المؤشرات.

الآن فيما يلي كيفية تجميع عناصر نفس المؤشرات باستخدام لغة برمجة بايثون:

```
inputLists = [[10, 20, 30], [40, 50, 60], [70, 80, 90]]
outputLists = []
index = 0

for i in range(len(inputLists[0])):
    outputLists.append([])
    for j in range(len(inputLists)):
        outputLists[index].append(inputLists[j][ index])
    index = index + 1
a, b, c = outputLists[0], outputLists[1], outputLists[2]
print(a, b, c)
```

```
[10, 40, 70] [20, 50, 80] [30, 60, 90]
```

هذه هي الطريقة التي يمكنك بها تجميع عناصر نفس المؤشرات باستخدام بايثون. اعثر على المزيد من أسئلة ومشاريع البرمجة التي تم حلها وشرحها باستخدام بايثون هنا.

الملخص

لتجميع عناصر من نفس الفهارس، سيكون لديك مبدئيًا قائمتان أو أكثر داخل قائمة مثل [[a, b], [c, d]] ، وتحتاج إلى إنشاء قائمتين جديدتين حيث ستخزن عناصر كلاهما القوائمفي الفهرس [a, c] 0 والفهرس [b, d] 1. أتمنى أن تكون قد أحببت هذه المقالة حول تجميع عناصر نفس المؤشرات باستخدام بايثون.

9) حساب وقت تنفيذ برنامج بايثون Calculating the Execution Time of a Python Program

يشير وقت تنفيذ البرنامج أو تشغيله (The execution or running time of the program) إلى مدى سرعة تسليم المخرجات بناءً على الخوارزمية التي استخدمتها لحل المشكلة. لحساب وقت تنفيذ البرنامج، نحتاج إلى حساب الوقت الذي يستغرقه البرنامج من بدايته إلى النتيجة النهائية. لذلك إذا كنت تريد معرفة كيفية حساب وقت تنفيذ برنامج بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول حساب وقت تنفيذ برنامج بايثون.

وقت تنفيذ برنامج بايثون

من المهم حساب وقت التنفيذ عند العملفي مشروع كبير. عند العملفي مشروع كبير، لدينا عدة طرقفي الاعتبار. يجب أن يكون الأفضل هو الذي يستغرق أقصر وقت تنفيذفي جميع السيناريوهات.

لذلك لحساب وقت تنفيذ برنامج بايثون، نحتاج إلى اتباع الخطوات المذكورة أدناه:

1. أولاً، قم بتخزين وقت بدء البرنامجفي متغير؛
2. اكتب برنامج بايثون؛
3. تخزين وقت انتهاء البرنامجفي متغير؛
4. اطرح وقت بدء البرنامج من وقت انتهاء البرنامج؛

في النهاية، ستحصل على وقت تنفيذ برنامجكفي ثوانٍ.

حساب وقت تنفيذ برنامج بايثون

دعنا الآن نتبع العمليةالموضحةفي القسم أعلاه لحساب الوقت الذي يستغرقه برنامج بايثون. هنا سأكتب برنامجًا بسيطًا لإنشاء الاختصارات:

```
from time import time
start = time()

# Python program to create acronyms
word = "Artificial Intelligence"
text = word.split()
a = " "
for i in text:
    a = a+str(i[0]).upper()
print(a)
```

```
execution_time = end - start
print("Execution Time : ", execution_time)
```

```
AI
Execution Time :  0.0002555847167968750
```

كما تريفي الإخراج أعلاه، تلقينا أولاً نتيجة برنامج بايثون، وفي السطر التالي، يمكننا رؤية وقت تشغيلهفي ثوانٍ. هذه هي الطريقة التي يمكنك بها حساب وقت تنفيذ أي برنامج.

الملخص

هذه هي الطريقة التي يمكنك من خلالها معرفة وقت تنفيذ برنامجك. لحساب وقت تنفيذ البرنامج، نحسب الوقت الذي يستغرقه البرنامج من بدايته حتى الإخراج النهائي. أتمنى أن تكون قد أحببت هذه المقالة حول حساب وقت تشغيل برنامج بايثون.

10) حساب عدد الكلمات في عمود باستخدام بايثون Count Number of Words in a Column using Python

أثناء العملفي مهمة علم البيانات، يتعين علينا أحيانًا التعامل مع البيانات النصية. إحدى المشكلات التي يواجهها المبتدئون أثناء العمل على مجموعة بيانات نصية (textual dataset) هي حساب عدد الكلماتفي جزء من النص. لذلك إذا كنت تريد معرفة كيفية حساب عدد الكلماتفي مجموعة بيانات نصية، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك خلال برنامج تعليمي حول كيفية حساب عدد الكلماتفي عمود باستخدام بايثون.

حساب عدد الكلمات في عمود باستخدام بايثون

يستخدم معظم المتخصصينفي علم البيانات مكتبة (pandas) لمعالجة البيانات وإعدادها. مكتبة (pandas) ليس لديها أي طريقة لحساب عدد الكلماتفي جزء من النص. تتمثل إحدى طرق حل هذه المشكلةفي إيجاد طول النص عن طريق تقسيم النص بالكامل.

فلنستورد مجموعة بيانات نصية (textual dataset) حيث يمكننا حساب عدد الكلماتفي عمود:

```
import pandas as pd
data =
pd.read_csv("https://raw.githubusercontent.com/amankharwal/Website-data/master/articles.csv", encoding = 'latin1')
print(data.head())
```

```
                                     Article  \
0  Data analysis is the process of inspecting and...
1  The performance of a machine learning algorith...
2  You must have seen the news divided into categ...
3  When there are only two classes in a classific...
4  The Multinomial Naive Bayes is one of the vari...

                                      Title
0                Best Books to Learn Data Analysis
1        Assumptions of Machine Learning Algorithms
2         News Classification with Machine Learning
3  Multiclass Classification Algorithms in Machin...
4       Multinomial Naive Bayes in Machine Learning
```

تحتوي مجموعة البيانات على عمودين مقال (Article) وعنوان (Title). دعنا ننشئ عمودًا جديداً بعدد الكلماتفي عمود المقالة:

```
data["Number of Words"] = data["Article"].apply(lambda n:
len(n.split()))
print(data.head())
```

```
                                     Article  \
0  Data analysis is the process of inspecting and...
1  The performance of a machine learning algorith...
2  You must have seen the news divided into categ...
3  When there are only two classes in a classific...
4  The Multinomial Naive Bayes is one of the vari...

                                       Title  Number of Words
0                 Best Books to Learn Data Analysis               76
1         Assumptions of Machine Learning Algorithms               56
2          News Classification with Machine Learning               70
3  Multiclass Classification Algorithms in Machin...               66
4        Multinomial Naive Bayes in Machine Learning               96
```

إذن، تحتوي مجموعة البيانات الآن على ثلاثة أعمدة، المقالة والعنوان وعدد الكلمات التي تحتوي على عدد الكلماتفي عمود المقالة.

الملخص

مكتبة (pandas) ليس لديها أي طريقة لحساب عدد الكلماتفي جزء من النص. تتمثل إحدى طرق حل هذه المشكلةفي إيجاد طول النص عن طريق تقسيم النص بالكامل. إذن، هذه هي الطريقة التي يمكنك بها حساب عدد الكلماتفي أي عمود أثناء العمل على مجموعة بيانات نصية. أتمنى أن تكون قد أحببت هذه المقالة حول حساب عدد الكلماتفي عمود باستخدام بايثون.

11) لعبة حجر ورق مقص باستخدام بايثون Rock Paper Scissors Game using Python

حجرة ورقة مقص (Rock Paper Scissors) هي لعبة يد يتم لعبها عادة بين شخصين.في هذه اللعبة، يمكن للمقص أن يتغلب على الورق، ويمكن للورق أن يتغلب على الحجرة، ويمكن للصخرة أن تتغلب على المقص. إذا كنت تريد أن تتعلم كيفية إنشاء لعبة حجرة ورقة مقص باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول إنشاء لعبة حجرة ورقة مقص باستخدام بايثون.

لعبة حجرة ورقة مقص باستخدام بايثون

لإنشاء ولعب حجرة ورقة مقص، سأستخدم عبارات (if)و (elif)في بايثون. سأقوم بإعداد هذه اللعبة لتلعب بين لاعبين. سيكون اللاعب الاول (Player-1) هو المستخدم، وسيكون اللاعب الثاني (player-2) هو الكمبيوتر. سيختار اللاعب الأول حجرة او ورقة أو المقص يدويًا، بينما سيختار اللاعب الثاني بشكل عشوائي. لذلك سأستخدم أيضًا الوحدة (random)في بايثون لإنشاء هذه اللعبة.

أتمنى أن تكون قد فهمت الآن كل شيء عن لعبة الحجرة والورقة والمقص وكيف سأقوم بإنشائها. الآن، فيما يلي كيفية كتابة سكريبت بايثون لإنشاء وتشغيل حجرة ورقة مقص باستخدام بايثون:

```
import random

player1 = input("Select Rock, Paper, or Scissor :").lower()
player2 = random.choice(["Rock", "Paper", "Scissor"]).lower()
print("Player 2 selected: ", player2)

if player1 == "rock" and player2 == "paper":
    print("Player 2 Won")
elif player1 == "paper" and player2 == "scissor":
    print("Player 2 Won")
elif player1 == "scissor" and player2 == "rock":
    print("Player 2 Won")
elif player1 == player2:
    print("Tie")
else:
    print("Player 1 Won")
```

Select Rock, Paper, or Scissor :Paper
Player 2 selected: scissor
Player 2 Won

هذه هي الطريقة التي يمكنك من خلالها إنشاء لعبة حجرة ورقة مقص بسهولة باستخدام لغة برمجة بايثون كمبتدئ. إذا كنت مبتدئًافي بايثون، فيجب أن تستمرفي العمل على مشاريع البرمجة هذه لتحسين مهاراتكفي البرمجة. يمكنك العثور هنا على بعض مشروعات بايثون التدريبية الرائعة للمبتدئين.

الملخص

هذه هي كيفية إنشاء ولعب حجرة ورقة مقص باستخدام لغة برمجة بايثون. حجرة ورقة مقص هي لعبة يد يتم لعبها عادة بين شخصين.في هذه اللعبة، يمكن للمقص أن يتغلب على الورق، ويمكن للورق أن يتغلب على الحجرة، ويمكن للصخرة أن تتغلب على المقص. آمل أن تكون قد أحببت هذا البرنامج التعليمي حول إنشاء حجرة ورقة مقص باستخدام بايثون.

12) طباعة الايموجيات باستخدام بايثون Printing the Emojis using Python

تُستخدم الرموز التعبيرية (Emojis) للتعبير عن مشاعرنا أثناء كتابة رسالة أو أي جزء من النص. إذا كنت تريد معرفة كيفية عرض الرموز التعبيريةفي الإخراج باستخدام لغة برمجة بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول كيفية طباعة الرموز التعبيرية باستخدام بايثون.

طباعة الايموجيات باستخدام بايثون

الابتسام (Smiling)، الإبهام لأعلى (thumbs up)، والرموز التعبيرية للقلب (heart emoji) هي بعض من الرموز التعبيرية التي نستخدمها غالبًا أثناء إرسال رسائل نصية إلى أصدقائنا أو زملائنا. من الممكن طباعة أي رمز تعبيري باستخدام لغة برمجة بايثون. لطباعة الرموز التعبيرية باستخدام بايثون، تحتاج إلى تثبيت وحدة (emoji)في بيئة بايثون الافتراضية. يمكنك تثبيته بسهولة باستخدام الأمر (pip)في التيرمينال أو موجه الأوامر كما هو مذكور أدناه:

```
pip install emoji
```

تساعدك طريقة emoji.emojize على كتابة وصف لأي رمز تعبيري داخل "::" أثناء كتابة جزء من النص. فيما يلي أمثلة لأوصاف بعض الرموز التعبيرية الشائعة:

1. :thumbs_up:
2. :red_heart:
3. :smiling_face:

يمكنك استخدام وصف أي رمز تعبيري داخل "::" لطباعة الرموز التعبيرية باستخدام بايثون. يمكنك العثور على وصف لجميع الرموز التعبيرية هنا. دعنا الآن نلقي نظرة على مثال لكيفية طباعة الرموز التعبيرية باستخدام بايثون:

```
print(emoji.emojize("I love reading books:books:"))
print(emoji.emojize("Some people have a very sensitive
heart:red_heart:, please be kind with them.:hibiscus:"))
```

```
I love reading books📚
Some people have a very sensitive heart❤️, please be kind with them.🌺
```

هذه هي الطريقة التي يمكنك بها استخدام الرموز التعبيرية وطباعتهافي مخرجاتك باستخدام لغة برمجة بايثون.

الملخص

تُستخدم الرموز التعبيرية للتعبير عن مشاعرنا أثناء كتابة رسالة أو أي جزء من النص. لطباعة أي رمز تعبيري باستخدام بايثون، تحتاج إلى تثبيت وحدة (emoji)في بيئة بايثون الافتراضية. أتمنى أن تكون قد أحببت هذه المقالة حول عرض الرموز التعبيريةفي الإخراج باستخدام لغة برمجة بايثون ووحدة الرموز التعبيرية.

13) تصحيح الهجاء باستخدام بايثون Correcting Spellings using Python

يعد تصحيح الهجاء (Correcting spellings)في جزء من النص إحدى الميزات المفيدة التي يمكن استخدامهافي أي تطبيق حيث يكتب المستخدمون المحتوى. على سبيل المثال، إذا كنت تريد إنشاء مفكرة (notepad)، فيجب أن تحتوي على ميزة للتعرف على الإملاء الخاطئ وتصحيحه. لذا، إذاكنت ترغبفي معرفة كيفية تصحيح الإملاء باستخدام لغة برمجة بايثون، فهذه المقالة مناسبة لك. ستقدم هذه المقالة أداة يدوية لتصحيح الإملاء باستخدام بايثون.

تصحيح الهجاء باستخدام بايثون

تعد وحدة (SpellChecker)في بايثون واحدة من أسهل الأدوات التي يمكن استخدامها لتصحيح الأخطاء الإملائيةفي جزء من النص. إذا لم تكن قد استخدمت وحدة بايثون هذه من قبل ، فيمكنك تثبيتها بسهولةفي بيئة بايثون الافتراضية الخاصة بك عن طريق تشغيل الأمر المذكور أدناهفي موجه الأوامر أو التيرمينال:

```
pip install pyspellchecker
```

الآن فيما يلي كيفية استخدام هذه الوحدة لتصحيح أي كلمة بها أخطاء إملائية باستخدام بايثون:

```
from spellchecker import SpellChecker
corrector = SpellChecker()

word = input("Enter a Word : ")
if word in corrector:
    print("Correct")
else:
    correct_word = corrector.correction(word)
    print("Correct Spelling is ", correct_word)
```

```
Enter a Word : intellignt
Correct Spelling is  intelligent
```

هناك العديد من البدائلفي بايثون لنفس المهمة، لكن وحدة المدقق الإملائي SpellChecker سهلة الاستخدام مقارنة بالبدائل الأخرى.

الملخص

هذه هي الطريقة التي يمكنك بها تصحيح أي كلمة بها أخطاء إملائية باستخدام لغة برمجة بايثون. يعد تصحيح الكلمات ذات الأخطاء الإملائيةفي جزء من النص إحدى الميزات المفيدة التي يمكن استخدامهافي أي تطبيق حيث يكتب المستخدمون المحتوى. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية تصحيح الإملاء باستخدام بايثون.

14) استخراج ملف تعريف GitHub باستخدام بايثون
Scraping GitHub Profile using Python

يعد تجريف الويب (Web scraping) أحد أهم المهارات التي يجب أن يمتلكها كل مبرمج. إذا كنت تريد معرفة كيفية جمع البيانات من GitHub باستخدام تقنيات تجريف الويب، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك عبر برنامج تعليمي حول تجريف ملف تعريف GitHub باستخدام بايثون.

استخراج ملف تعريف GitHub باستخدام بايثون

عندما نفتح أي حساب على GitHub، نرى صورة الملف الشخصي واسم المستخدم ووصفًا موجزًا للمستخدمفي قسم الملف الشخصي. هنا سوف تتعلم كيفية كشط صورة ملفك الشخصي على GitHub. لهذه المهمة، تحتاج إلى بعض المعرفة بـ HTML ومكتبات(requests) و(BeautifulSoup)في بايثون.

إذا لم تكن قد استخدمت مكتبة (BeautifulSoup)من قبل، فاستخدم الأمر المذكور أدناهفي موجه الأوامر أو التيرمينال لتثبيت هذه المكتبةفي بيئة بايثون الافتراضية الخاصة بك:

```
pip install beautifulsoup4
```

لست بحاجة إلى تثبيت مكتبة (requests) لأنها موجودة بالفعلفي مكتبة بايثون القياسية. الآن فيما يلي كيفية كتابة برنامج بايثون لجرف صورة ملف تعريف من أي ملف تعريف GitHub:

```
import requests
from bs4 import BeautifulSoup as bs

github_profile = "https://github.com/amankharwal"
req = requests.get(github_profile)
scraper = bs(req.content, "html.parser")
profile_picture = scraper.find("img", {"alt":
"Avatar"})["src"]
print(profile_picture)
```

```
Output:
https://avatars.githubusercontent.com/u/57987909?v=4
```

الآن، إذا نقرت على الرابط الذي حصلت عليه كمخرج، فسترى صورة الملف الشخصي لمستخدم GitHub. هذه هي الطريقة التي يمكنك بها تجريف صور الملف الشخصي من أي ملف تعريف GitHub باستخدام بايثون.

ملخص

هذه هي الطريقة التي يمكنك بها تجريف ملف تعريف GitHub باستخدام لغة برمجة بايثون. يعد تجريف الويب أحد أهم المهارات التي يجب أن يمتلكها كل مبرمج. لذلك يجب أن تعرف كيفية تجريف الصور من أي موقع ويب باستخدام تقنيات تجريف الويب. آمل أن تكون قد أحببت هذه المقالة حول إلغاء أي ملف تعريف على GitHub باستخدام بايثون.

15) تصوير العلاقة الخطية باستخدام بايثون Visualizing Linear Relationship using Python

العلاقة الخطية (linear relationship) هي مصطلح إحصائي ليس سوى علاقة بين متغيرين. توضح العلاقة الخطية مدى ارتباط متغيرين x و y ببعضهما البعض. بصفتك محترفًافي علم البيانات، يجب أن تعرف كيفية تصور علاقة خطية لأنها ستظهر العلاقة بين ميزتين عدديتين لمجموعة البيانات. لذلك إذا كنت تريد معرفة كيفية تصور علاقة خطية، فهذه المقالة مناسبة لك. في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول كيفية تصوير علاقة خطية باستخدام بايثون.

تصوير علاقة خطية باستخدام بايثون

عندما تزداد قيمة المتغير أو تنقص مع زيادة أو نقصان قيمة متغير آخر، فهي ليست سوى علاقة خطية. عندما نتخيل علاقة خطية، فإنها توضح ما إذا كانت العلاقة بين السمتين خطية أم لا.

يمكنك استخدام أي مكتبة لرسم البياناتفي بايثون لتصور علاقة خطية. أفضل استخدام (plotly) لأنه يوفر نتائج تفاعلية. ولكن نظرًا لأن العديد من مبرمجي بايثون يستخدمون (matplotlib) لتصوير البيانات (data visualization)، فسوف أوضح لك كيفية تصوير علاقة خطية مع بايثون باستخدام (plotly) و(matplotlib).

تصوير العلاقات الخطية باستخدام بايثون

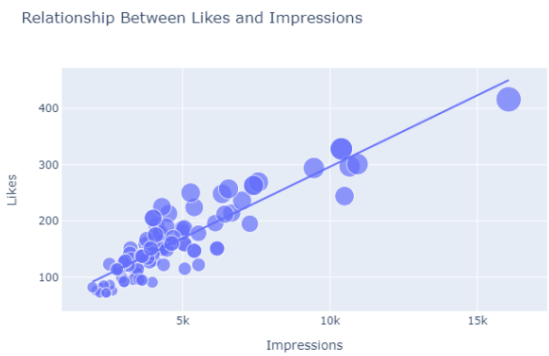
فلنستورد مجموعة بيانات وجميع مكتبات بايثون الضرورية لهذه المهمة:

```
import pandas as pd
import numpy as np
import plotly.express as px
import matplotlib.pyplot as plt
import seaborn as sns

data =
pd.read_csv("https://raw.githubusercontent.com/amankharwal/Web
site-data/master/Instagram.csv", encoding = 'latin1')
data = data.dropna()
print(data.head())
```

```
   Impressions  From Home  From Hashtags  From Explore  From Other  Saves  \
0       3920.0     2586.0         1028.0         619.0        56.0   98.0
1       5394.0     2727.0         1838.0        1174.0        78.0  194.0
2       4021.0     2085.0         1188.0           0.0       533.0   41.0
3       4528.0     2700.0          621.0         932.0        73.0  172.0
4       2518.0     1704.0          255.0         279.0        37.0   96.0

   Comments  Shares  Likes  Profile Visits  Follows  \
0       9.0     5.0  162.0            35.0      2.0
1       7.0    14.0  224.0            48.0     10.0
2      11.0     1.0  131.0            62.0     12.0
3      10.0     7.0  213.0            23.0      8.0
4       5.0     4.0  123.0             8.0      0.0

                                             Caption  \
0  Here are some of the most important data visua...
1  Here are some of the best data science project...
2  Learn how to train a machine learning model an...
3  Here□s how you can write a Python program to d...
4  Plotting annotations while visualizing your da...

                                            Hashtags
0  #finance #money #business #investing #investme...
1  #healthcare #health #covid #data #datascience ...
2  #data #datascience #dataanalysis #dataanalytic...
3  #python #pythonprogramming #pythonprojects #py...
4  #datavisualization #datascience #data #dataana...
```

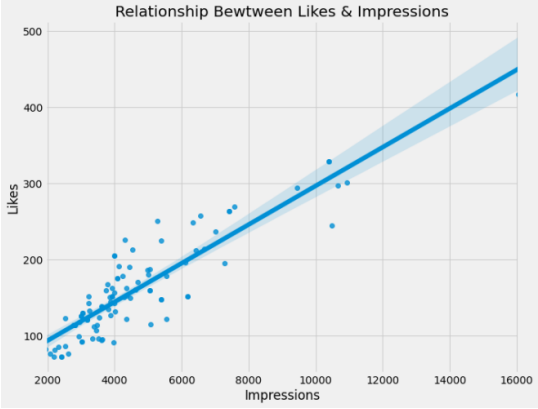
إليك كيفية تصوير العلاقات الخطية باستخدام مكتبة (plotly)في بايثون:

```
figure = px.scatter(data_frame = data ,
                    x="Impressions,"
                    y="Likes ,"
                    size="Likes ,"
                    trendline="ols ,"
                    title = "Relationship Between Likes and
Impressions("
figure.show()
```



لتصوير العلاقات الخطية باستخدام (matplotlib)، عليك استخدام طريقة seaborn.regplot. إليك كيفية رسم العلاقات الخطية باستخدام مكتبة (matplotlib)في بايثون:

```
plt.figure(figsize=(10, 8))
plt.style.use('fivethirtyeight')
plt.title("Relationship Bewtween Likes & Impressions")
sns.regplot(x="Impressions", y="Likes", data=data)
plt.show()
```



الملخص

هذه هي كيفية تصوير العلاقات الخطية باستخدام لغة برمجة بايثون. عندما تزداد قيمة المتغير أو تنقص مع زيادة أو نقصان قيمة متغير آخر، فهي ليست سوى علاقة خطية. آمل أن تكون قد أحببت هذه المقالة حول تصور العلاقات الخطية باستخدام بايثون.

16) إنشاء نص باستخدام بايثون Generating Text using Python

يتضمن إنشاء النص (Text generation) إنشاء نص باستخدام تقنيات التعلم الآلي (machine learning). الغرض من إنشاء النص هو إنشاء نص تلقائيًا لا يمكن تمييزه عن نص مكتوب بواسطة إنسان. إذا كنت تريد معرفة كيفية إنشاء نص باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية استخدام نموذج إنشاء نص GPT-2 الشهير لإنشاء نص باستخدام بايثون.

ما هو نموذج GPT-2؟

يرمز GPT-2 إلى المحول التوليدي مسبق التدريب 2 (Generative Pre-trained Transformer 2). وهو عبارة عن نموذج مفتوح المصدر لمعالجة اللغة الطبيعية (Natural Language Processing) تم إنشاؤه بواسطة OpenAI. يمكنه إنشاء فقرات نصية بأداء حديث على العديد من معايير اللغة. كما أنها تستخدم للترجمة الآلية (machine translation) والإجابة على الأسئلة (question answering) وتلخيص النص (text summarization).

لاستخدام نموذج GPT-2 لإنشاء نص باستخدام بايثون، تحتاج إلى تثبيت مكتبة (Transformers)في بايثون. يمكن تثبيته بسهولة باستخدام الأمر (pip)في موجه الأوامر أو التيرمينال كما هو مذكور أدناه:

```
pip install transformers
```

آمل أن تكون قد فهمت الآن ما هو نموذج GPT-2 وكيف يمكنك تثبيتهفي بيئة بايثون الافتراضية الخاصة بك. يمكنك قراءة المزيد عن هذا النموذج هنا. الآنفي القسم أدناه، سأشرح كيف يمكنك استخدام هذا النموذج لإنشاء نص باستخدام بايثون.

إنشاء نص باستخدام بايثون

دعنا نستورد نموذج GPT-2 من مكتبة (Transformers) ونبدأ بمهمة إنشاء نص باستخدام بايثون:

```
from transformers import pipeline
model = pipeline("text-generation", model = "gpt2")
```

```
Downloading: 100%
665/665 [00:00<00:00, 8.60kB/s]
Downloading: 100%
523M/523M [00:11<00:00, 43.5MB/s]
Downloading: 100%
0.99M/0.99M [00:00<00:00, 1.74MB/s]
Downloading: 100%
446k/446k [00:00<00:00, 1.74MB/s]
Downloading: 100%
1.29M/1.29M [00:00<00:00, 3.44MB/s]
```

إليك كيفية إنشاء نص باستخدام بايثون باستخدام نموذج GPT-2:

```
sentence = model("Hi, My name is John Cena, I am here",
                 do_sample=True, top_k=50,
                 temperature=0.9, max_length=100,
                 num_return_sentences=2)

for i in sentence:
  print(i["generated_text"])
```

```
Hi, My name is John Cena, I am here to see you. I have been here this entire time. I've worked. I've
seen all these things. It's just, man, my life has changed because of you guys. You guys get to see
everything, including my career, things like that. You guys have, you know, the most amazing stuff
about me.

JANUARY 10, 2015:

After the match, the fans were happy.
```

الملخص

الغرض من إنشاء النص هو إنشاء نص تلقائيًا لا يمكن تمييزه عن النص المكتوب بواسطة إنسان. GPT-2 هو نموذج مفتوح المصدر لمعالجة اللغة الطبيعية تم إنشاؤه بواسطة OpenAI. يمكنه إنشاء فقرات نصية بأداء حديث على العديد من معايير اللغة. أتمنى أن تكون قد أحببت هذه المقالة حول إنشاء نص باستخدام بايثون ونموذج GPT-2.

17) استخراج الجدول من موقع ويب باستخدام بايثون
Scraping Table From a Website using Python

يعد تجريف الويب (Web Scraping) إحدى المهارات التي يجب أن يعرفها كل متخصص في علم البيانات. في بعض الأحيان، تكون البيانات التي نحتاجها متاحة على موقع ويب في شكل جدول لا يمكن تنزيله مباشرة من موقع الويب. لاستخدام هذه البيانات في أي مهمة تتعلق بعلم البيانات، نحتاج إلى جمعها من موقع الويب باستخدام تقنيات تجريف الويب. لذلك إذا كنت تريد معرفة كيفية تجريف جدول من موقع ويب، فهذه المقالة مناسبة لك. في هذه المقالة، سوف آخذك خلال برنامج تعليمي حول كيفية تجريف جدول من موقع ويب باستخدام بايثون.

استخراج الجدول من موقع على شبكة الإنترنت باستخدام بايثون

هناك العديد من مكتبات ووحدات بايثون التي يمكنك استخدامها في تجريف الويب. لاستخراج جدول من موقع ويب، سأستخدم وحدة (urllib) في بايثون، وهي متوفرة بالفعل في مكتبة بايثون القياسية. لذلك لا تحتاج إلى تثبيت أي مكتبة خارجية لتجريف البيانات من موقع ويب. فيما يلي كيفية استخدام وحدة (urlib) لتجريف جدول من موقع ويب باستخدام لغة برمجة بايثون:

```
import urllib.request
import pandas as pd
url =
"https://en.wikipedia.org/wiki/Programming_languages_used_in_most_popular_websites"

with urllib.request.urlopen(url) as i:
    html = i.read()

data = pd.read_html(html)[0]
print(data.head())
```

```
  Websites Popularity(unique visitors per month)[1]  Front-end(Client-side)  \
0  Google[2]                                1600000000  JavaScript, TypeScript
1   Facebook                                1120000000              JavaScript
2    YouTube                                1100000000   JavaScript,TypeScript
3      Yahoo                                 750000000              JavaScript
4       Etsy                           516,000,000[15]              JavaScript

                          Back-end(Server-side)  \
0           C, C++, PHP, Go,[3] Java, Python, Node
1  Hack, PHP (HHVM), Python, C++, Java, Erlang, D...
2           C, C++, Python, PHP, Java, [11] Go[12]
3                                              PHP
4                                   PHP[16][17]

                             Database  \
0                 Bigtable,[4] MariaDB[5]
1    MariaDB, MySQL,[9] HBase, Cassandra[10]
2          Vitess, BigTable, MariaDB[5][13]
3  PostgreSQL, HBase, Cassandra, MongoDB,[14]
4                       MySQL, Redis[18]

                                      Notes
0         The most used search engine in the world
1          The most visited social networking site
2  The most popular video sharing site [YouTube i...
3                                              NaN
4                          E-commerce website.
```

في الكود أعلاه، أقوم بجمع البيانات من جدول متاح على صفحة ويب تحتوي على جدول يصف لغات البرمجة المستخدمةفي معظم الشركات الشائعة. يمكنك رؤية البيانات التي تلقيناها بعد تجريف الويب حول لغات البرمجة وقواعد البيانات التي تستخدمها الشركات. هذه هي الطريقة التي يمكنك بها تجريف الجداول من أي موقع ويب باستخدام لغة برمجة بايثون.

إذا كنت تريد حفظ هذه البياناتفي ملفCSV ، فيما يلي كيفية حفظها:

```
data.to_csv("programming.csv")
```

بعد تشغيل الكود أعلاه، سترى ملف CSV محفوظًافي نفس الدليل حيث يوجد ملف بايثون الخاص بك.

ملخص

هذه هي الطريقة التي يمكننا بها تجريف الجداول من موقع ويب باستخدام بايثون. يعد تجريف الويب إحدى المهارات التي يجب أن يعرفها كل متخصصفي علم البيانات. أتمنى أن تكون قد أحببت هذه المقالة عن تجريف جداول من مواقع الويب التي تستخدم بايثون.

18) استخراج النص من ملف PDF باستخدام بايثون
Extracting Text From PDF with Python

يحتاج مطور بايثون أحيانًا إلى جمع بعض المعلومات النصية من ملفات PDF. لذا فإن استخراج نص من ملف PDF يمثل مشكلة يجب أن تعرف كيفية حلها كمطور بايثون. إذا كنت تريد معرفة كيفية استخراج نص من ملف pdf، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول كيفية استخراج النص من ملف pdf باستخدام بايثون.

استخراج نص من ملف PDF باستخدام بايثون

يجب أن تعرف كيفية جمع النص من pdf كمطور بايثون. هذه المهارة مفيدة عند العمل مع السير الذاتية. لا يعد استخراج نص من ملف pdf مهمة صعبة على الإطلاق. لهذه المهمة، تحتاج إلى تثبيت مكتبة بايثون المعروفة باسم (PyPDF2).

يمكنك بسهولة تثبيت مكتبة بايثون هذه باستخدام الأمر (pip)في التيرمينال أو موجه الأوامر كما هو مذكور أدناه:

```
pip install pypdf2
```

بعد تثبيت مكتبة بايثون هذه، نحن جميعًا على استعداد لاستخراج النص من أي ملف pdf. فيما يلي كيفية استخراج نص من أي ملف PDF باستخدام لغة برمجة بايثون:

```
import PyPDF2
pdf = open("Aman.pdf", "rb")
reader = PyPDF2.PdfFileReader(pdf)
page = reader.getPage(0)
print(page.extractText())
```

في السطر الرابع من الكود أعلاه، ستساعدك طريقة getPage()في تحديد رقم الصفحة التي تريد استخراج النص منها.

الملخص

هذه هي الطريقة التي يمكنك بها جمع نص من ملف PDF باستخدام لغة برمجة بايثون. يعد استخراج نص من ملف PDF مشكلة يجب أن تعرف كيفية حلها كمطور بايثون. أتمنى أن تكون قد أحببت هذه المقالة حول استخراج نص من ملفات PDF باستخدام بايثون.

19) عكس سلسلة نصية باستخدام بايثون Reversing a String using Python

السلسلة النصية (string) عبارة عن سلسلة من الأحرف المضمنة بين علامتي اقتباس مفردة أو مزدوجة. يعد عكس السلسلة النصية أحد أكثر المشكلات شيوعًافي علوم الكمبيوتر. هنا نحتاج إلى عكس أحرف سلسلة. لذلك، إذا كنت تريد معرفة كيفية عكس سلسلة، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على برنامج تعليمي حول كيفية عكس سلسلة باستخدام بايثون.

عكس سلسلة باستخدام بايثون

هناك طرق عديدة لعكس سلسلة نصية باستخدام بايثون. يمكنك استخدام أي طريقة تجدها سهلة ما لم يُطلب منك استخدام طريقة معينة.

يجب أن تكون قد سمعت عن مفهوم التقطيع (slicing)في بايثون. سأوضح لك هنا كيفية استخدام تقطيع السلاسل لعكس سلسلة باستخدام بايثون:

```
def reverse_string(string):
    return string[1-::]

a = "lawrahK namA"
print(reverse_string(a))
```

```
Output:
Aman Kharwal
```

يحتوي الحرف الأولفي السلسلة على الفهرس 0، بينما يحتوي الحرف الأخير على الفهرس-n 1، حيث يمثل n طول السلسلة. يقرأ عامل تقطيع السلسلة النصية "::" جميع أحرف السلسلة ، و 1ـ ،في النهاية ، يعكس ترتيب الأحرف. هذه هي الطريقة التي يمكننا بها عكس السلسلة.

ملخص

هذه هي الطريقة التي يمكننا بها استخدام تقطيع السلاسل لعكس ترتيب أحرف السلسلة. يعد انعكاس السلسلة أحد أكثر المشكلات شيوعًافي علوم الكمبيوتر. آمل أن تكون قد أحببت هذه المقالةفي برنامج تعليمي حول عكس سلسلة باستخدام لغة برمجة بايثون.

20) مطابقة التسلسلات باستخدام بايثون Match Sequences using Python

SequenceMatcher هو كلاس (class)في بايثون متوفرةفي وحدة (difflib)، والتي توفر دوال لمقارنة التسلسلاتفي جزأين مختلفين من النص. لذلك عندما تريد مقارنة ملفين نصيين، يمكنك استكشاف وحدة (difflib)في بايثون. إذا لم تستخدم فئة SequenceMatcherفي بايثون مطلقًا، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك خلال برنامج تعليمي حول SequenceMatcherفي بايثون.

SequenceMatcher في بايثون

كلاس SequenceMatcher متاحةفي وحدة (difflib)في بايثون، وهي متوفرةفي مكتبة بايثون القياسية. ليس عليك تثبيته قبل استخدامه. هناك العديد من الفئاتفي وحدة (difflib) لمقارنة النصوص. أحد هذه الفئات هو SequenceMatcher الذي يحسب مدى تطابق تسلسل نصين مع بعضهما البعض. بكلمات بسيطة، تجد أوجه التشابهفي تسلسل نصين مختلفين.

دعونا نرى كيفية استخدام هذا الكلاس لإيجاد أوجه التشابهفي تسلسل نصين. سأقوم أولاً بإدخال نصين متشابهين جدًافي هذا الكلاس:

```
from difflib import SequenceMatcher
text1 = "My Name is Aman Kharwal"
text2 = "Hi, My Name is Aman Kharwal"
sequenceScore = SequenceMatcher(None, text1, text2).ratio()
print(f"Both are {sequenceScore * 100} % similar")
```

```
Both are 92.0 % similar
```

لذلك، وفقًا للنتيجة أعلاه، فإنه يوضح أن كلا مدخلات النص لها تسلسلات متشابهة جدًا. لنجربها الآن باستخدام مدخلات نصية مختلفة عن بعضها البعض:

```
text1 = "My Name is Aman Kharwal"
text2 = "I am the founder of thecleverprogrammer.com"
sequenceScore = SequenceMatcher(None, text1, text2).ratio()
print(f"Both are {sequenceScore * 100} % similar")
```

```
Both are 24.242424242424242 % similar
```

لذلك، وفقًا للنتيجة أعلاه، يُظهر أن كلا من مدخلات النص لهما تسلسلات أقل تشابهًا. هذه هي الطريقة التي يمكنك بها استخدام هذا الكلاسفي بايثون المتوفرةفي وحدة (difflib).

ملخص

كلاس SequenceMatcher متاحفي وحدة (difflib)في بايثون، وهي متوفرةفي مكتبة بايثون القياسية. ليس عليك تثبيته قبل استخدامه. أتمنى أن تكون قد أحببت هذه المقالةفي برنامج تعليمي حول SequenceMatcherفي بايثون.

21) رمز QR باستخدام بايثون QR Code using Python

QR هو رمز يمكن مسحه ضوئيًا يستخدم لتخزين المعلومات. تُستخدم رموز QR لإعادة توجيهك إلى صفحة معينة أو لتظهر لك بعض المعلومات. يجب أن تكون قد قمت بمسح رموز QR ضوئيًا مرة واحدةفي حياتك عند إجراء المدفوعات. إذا كنت مهتمًا بمعرفة كيفية إنشاء رمز QR باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم لك برنامجًا تعليميًا حول كيفية إنشاء رمز QR باستخدام بايثون.

رمز QR باستخدام بايثون

رموز QR لها استخدامات متنوعة؛ بدءًا من إنشاء بوابة دفع وحتى عرض قائمة طعام أحد المطاعم، يتم استخدام رمز QR بعدة طرق. على مدار السنوات الخمس الماضية، بدأت العديد من الشركات التي تعتمد فقط على إنشاء رموز QR للأعمال. لذلك إذا كنت تعرف كيفية إنشاء رمز QR، فسيكون ذلك مفيدًا لك من نواح كثيرة.

لذلك، لإنشاء رموز QR باستخدام لغة برمجة بايثون، تحتاج أولاً إلى التأكد من تثبيت وحدتي PyQRCode و pypngفي بيئة بايثون الافتراضية الخاصة بك. يمكنك بسهولة تثبيت كلتا الوحدتين من خلال تنفيذ الأوامر المذكورة أدناهفي موجه الأوامر أو التيرمينال:

```
pip install PyQRCode
pip install pypng
```

بعد تثبيت هذه الوحدات، يمكنك البدءفي كتابة برنامج لإنشاء رمز QR باستخدام بايثون، كما هو موضحفي قسم الكود أدناه:

```
import pyqrcode
import png
link = "https://www.instagram.com/the.clever.programmer"/
qr_code = pyqrcode.create(link)
qr_code.png("instagram.png", scale=5)
```

سيتم حفظ رمز QR الذي تم إنشاؤه بواسطة برنامج بايثون هذافي نفس الدليل حيث يوجد ملف بايثون الخاص بك. سيقوم رمز QR هذا بإعادة توجيهك إلى حساب Instagram الخاص بي.

ملخص

هذه هي الطريقة التي يمكنك بها إنشاء رموز QR باستخدام لغة برمجة بايثون. أكواد QR هي أكواد يمكن مسحها ضوئيًا تُستخدم لتخزين المعلومات. يتم استخدامها لإعادة توجيهك إلى

صفحة معينة أو لتظهر لك بعض المعلومات. آمل أن تكون قد أحببت هذه المقالة حول إنشاء أكواد QR باستخدام بايثون.

22) فك تشفير رمز QR باستخدام بايثون Decoding a QR Code using Python

يعني فك تشفير رمز الاستجابة السريعة العثور على القيمة أو الرقم أو النص أو الرابط الموجود خلف رمز QR. هناك العديد من الطرق لفك تشفير رمز QR الذي تستخدمه كاميرات هاتفك الذكي، مما يساعدك على مسح رمز QR ضوئيًا أثناء إجراء المدفوعات عبر الإنترنت. لذلك إذا كنت تريد معرفة كيفية فك تشفير رمز QR، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول كيفية فك تشفير رمز QR باستخدام بايثون.

فك تشفير رمز QRباستخدام بايثون

لقد قمت مؤخرًا بمشاركة مقال حول إنشاء رمز QR باستخدام بايثون. إذا كنت تريد التعرف على كيفية إنشاء رمز QR، فيمكنك مراجعة المقالة السابقة. لفك شفرة QR، تحتاج إلى صورة لرمز QR. يمكنك استخدام أي صورة لرمز QR لهذا البرنامج التعليمي، أو يمكنك إنشاء رمز QR الخاص بك.

لفك رموز QR باستخدام بايثون، تحتاج إلى تثبيت مكتبتين بايثونفي بيئة بايثون الخاصة بك؛ Pyzbar وpillow . يمكنك تثبيت هاتين المكتبتين عن طريق تنفيذ الأوامر المذكورة أدناهفي موجه الأوامر أو التيرمينال:

```
pip install pyzbar
pip install pillow
```

الآن فيما يلي كيفية كتابة برنامج لفك شفرة QR باستخدام بايثون:

```
from pyzbar.pyzbar import decode
from PIL import Image
decocdeQR = decode(Image.open('instagram.png'))
print(decocdeQR[0].data.decode('ascii'))
```

```
https://www.instagram.com/the.clever.programmer/
```

في الكود أعلاه، أستخدم صورة رمز QR التي تعيد توجيه الأشخاص إلى حسابي على Instagram. إذن هذا البرنامج يعطي رابط حساب Instagram الخاص بيفي الإخراج.

الملخص

هذه هي الطريقة التي يمكنك بها فك رموز QR باستخدام لغة برمجة بايثون. يعني فك رموز QR العثور على القيمة أو الرقم أو النص أو الرابط الموجود خلف رمز QR. آمل أن تكون قد أحببت هذه المقالة حول فك رموز QR باستخدام بايثون.

23) إنشاء بيانات وهمية باستخدام بايثون Creating Dummy Data using Python

إذا كنت تتعلم علم البيانات (Data Science) وتجد صعوبةفي إنشاء مجموعة بيانات للتدرب عليها من البداية، فيمكنك إما تنزيل مجموعة بيانات من Kaggle أو إنشاء بيانات مزيفة. إذا كنت تريد معرفة كيفية إنشاء مجموعة بيانات وهمية (dummy dataset)في بضعة أسطر من التعليمات البرمجية، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية إنشاء بيانات وهمية باستخدام بايثون.

إنشاء بيانات وهمية باستخدام بايثون

لإنشاء بيانات وهمية باستخدام بايثون، يمكننا استخدام مكتبة(faker). تقوم مكتبة faker بإنشاء بيانات مزيفة بشكل عشوائي. إذا لم تكن قد استخدمت هذه المكتبة من قبل، فيمكنك تثبيتها بسهولة باستخدام الأمر (pip)المذكور أدناهفي موجه الأوامر أو التيرمينال:

```
pip install faker
```

دعونا الآن نلقي نظرة على بعض الأمثلة على هذه المكتبة قبل إنشاء مجموعة بيانات وهمية. سيعيد الكود أدناه اسمًا (name) وعنوانًا (address) ونصًا (text) مزيفًا بشكل عشوائي:

```
from faker import Faker
fake = Faker()
print(fake.name())
print(fake.address())
print(fake.text())
```

```
Sean Obrien
2606 Mackenzie Tunnel Apt. 215
East Ericfurt, CO 88091
Building job station sometimes what language money. Able air really it study suffer health. Body why
approach difference case notice choose.
```

في كل مرة تقوم فيها بتشغيل هذا الكود، ستحصل على نتائج مختلفة. دعنا الآن نرى كيفية إنشاء بيانات وهمية لإنشاء مجموعة بيانات وهمية باستخدام بايثون.

تقوم طريقة Faker().profile() بإرجاع بيانات مزيفة حول ملفات تعريف الدوال التي تحتوي على 13 عمودًا. فيما يلي كيفية إنشاء مجموعة بيانات وهمية باستخدام بايثون:

```
from faker import Faker
import pandas as pd
fake = Faker()
data = [fake.profile() for i in range(50)]
data = pd.DataFrame(data)
```

```
print(data.head())
```

```
                                job  ...   birthdate
0  Engineer, control and instrumentation  ...  1949-06-13
1                    Editor, film/video  ...  1959-07-23
2                         Chiropractor  ...  1927-12-12
3                         Nurse, adult  ...  1996-11-02
4                   Personnel officer  ...  1953-08-19

[5 rows x 13 columns]
```

يمكنك معرفة المزيد حول إنشاء بيانات وهمية باستخدام مكتبة faker من هنا.

الملخص

هذه هي الطريقة التي يمكنك بها إنشاء مجموعة بيانات وهمية أو زائفة باستخدام لغة برمجة بايثون. إذا كنت ترغبفي العمل مع مجموعات بيانات أفضل، فإنني أوصي بزيارة Kaggle. آمل أن تكون قد أحببت هذه المقالة حول إنشاء بيانات وهمية باستخدام بايثون.

24) إزالة الكلمات النابية باستخدام بايثون Removing Cuss Words using Python

الكلمات النابية (Cuss Words) هي الكلمات التي تجعل لغتك تبدو غير مهذبة ووقحة ومهينة ثقافيًا. نحتاجفي بعض الأحيان إلى تحديد وإزالة الكلمات النابية من جزء من النص. لذلك إذا كنت تريد أن تتعلم إزالة الكلمات النابية، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية إزالة الكلمات النابية باستخدام بايثون.

إزالة الكلمات النابية باستخدام بايثون

في أحد الأبحاث، وجد أنهفي المتوسط، 80 ـ 90 كلمة يتحدث بها الشخص كل يوم، 50 ـ 70٪ من جميع الكلمات هي كلمات نابية. هذا يعني أن الناس يجدونها طبيعية أثناء تبادل المحادثات مع الكلمات النابية. لكنفي بعض الأحيان، نحتاج إلى إزالة مثل هذه الكلمات من قطعة نصية لإتاحتها للجمهور من كل فئة عمرية.

لذا لإزالة الكلمات النابية، باستخدام لغة برمجة بايثون، نحتاج إلى تثبيت مكتبة (better_profanity)في بيئة بايثون الخاصة بنا. يساعدفي التعرف على الكلمات النابية وإزالتها عن طريق إدخال الرمز *في كل حرف من كلمة لعنة.

لتثبيت مكتبة بايثون هذهفي بيئة بايثون الخاصة بك، تحتاج إلى تنفيذ الأمر المذكور أدناهفي موجه الأوامر أو التيرمينال:

```
pip install better_profanity
```

بعد تثبيت مكتبة بايثون هذه، فيما يلي كيفية كتابة برنامج لإزالة الكلمات النابية باستخدام بايثون:

```
from better_profanity import profanity
text = "Please leave me alone and just piss off"
censored = profanity.censor(text)
print(censored)
```

```
Please leave me alone and just ****
```

لذلك، كما ترونفي الإخراج، تمت إزالة كلمة نابية من النص، ونرى أربع نجوم بدلاً من الكلمة النابية.

ملخص

الكلمات النابية (Cuss Words) هي الكلمات التي تجعل لغتك تبدو غير مهذبة ووقحة ومهينة ثقافيًا.في أحد الأبحاث، وجد أنهفي المتوسط، 80ـ90 كلمة يتحدث بها الشخص كل يوم، 50ـ 70٪ من جميع الكلمات هي كلمات نابية.

25) ايجاد القيم المكررة باستخدام بايثون Finding Duplicate Values using Python

يعد العثور على قيم مكررة (Finding duplicate values) من مصفوفة أو أي هيكل بيانات أخرى أحد أسئلة مقابلة البرمجة الشائعة التي يمكنك الحصول عليهافي أي مقابلة برمجة. توفر لغة برمجة بايثون العديد من الدوال المضمنة للعثور على القيم المكررة، ولكنفي مقابلة البرمجة، يجب عليك استخدام خوارزمية بدلاً من دالة مضمنة. لذلك إذاكنت تريد معرفة كيفية العثور على قيم مكررة، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج للعثور على قيم مكررة باستخدام بايثون.

ايجاد القيم المكررة باستخدام بايثون

لكتابة برنامج للعثور على قيم مكررة باستخدام بايثون، سأحدد دالة بايثون التي ستأخذ قائمة بالقيمفي أي نوع بيانات. فيما يلي دالة بايثون للعثور على قيم مكررةفي قائمة (list):

```
def find_duplicates(x):
    length = len(x)
    duplicates[] =
    for i in range(length):
        n = i + 1
        for a in range(n, length):
            if x[i] == x[a] and x[i] not in duplicates:
                duplicates.append(x[i])
    return duplicates
names = ["Aman", "Akanksha", "Divyansha", "Devyansh ,"
"          Aman", "Diksha", "Akanksha["
print(find_duplicates(names))
```

```
['Aman', 'Akanksha']
```

فيما يلي كيفية عمل الوظيفة المذكورة أعلاه:

1. الدالة المذكورة أعلاه تأخذ قائمة كمدخلات؛
2. ثم تحسب طول القائمة؛
3. ثم يبحث عن نفس القيمةفي القائمة الموجودةفي الفهرس الأول؛
4. إذا عثرت على قيم متعددة، فإنها تلحق تلك القيمةفي قائمة أخرى من القيم المكررة؛
5. تستمر هذه العملية حتى تصل الحلقة إلى الفهرس النهائي للقائمة.في النهاية، تقوم بإرجاع قائمة القيم المكررة.

يمكنك استخدام هذه الدالةفي قائمة بايثون من أي نوع بيانات.

ملخص

هذه هي الطريقة التي يمكنك بها كتابة دالة بايثون للعثور على القيم المكررةفي قائمة من أي نوع بيانات. توفر لغة برمجة بايثون العديد من الدوال المضمنة للعثور على العناصر المكررة، ولكنفي مقابلة البرمجة، يجب عليك استخدام خوارزمية بدلاً من دالة مضمنة. آمل أن تكون قد أحببت هذه المقالة حول العثور على قيم مكررةفي قائمة بايثون.

26) كشف الأسئلة باستخدام بايثون Detecting Questions using Python

الكشف عن الأسئلة (Detecting questions) يعني تحديد ما إذا كانت الجملة استفهام أم لا. يمكننا أيضًا استخدام التعلم الآلي (machine learning) لاكتشاف الأسئلة، ولكن نظرًا لأننا جميعًا نعرف نوع الجمل التي نراهافي جملة الاستفهام، فمن الممكن أيضًا كتابة نص برمجي بايثون لاكتشاف ما إذا كانت الجملة عبارة عن سؤال أم لا. لذلك إذا كنت تريد معرفة كيفية اكتشاف ما إذا كانت الجملة عبارة عن سؤال أم لا، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج لاكتشاف الأسئلة باستخدام بايثون.

كشف الأسئلة باستخدام بايثون

لكتابة برنامج بايثون لاكتشاف ما إذا كانت الجملة عبارة عن سؤال أم لا، نحتاج أولاً إلى إنشاء قائمة (list) بالكلمات التي نراهافي بداية جملة الاستفهام. على سبيل المثال ما اسمك؟ أين تعيش؟،في هذين السؤالين، "ماذا" و "أين" هي أنواع الكلمات التي نحتاج إلى تخزينهافي قائمة بايثون. بعد ذلك، للتحقق مما إذا كانت الجملة عبارة عن سؤال أم لا، نحتاج إلى التحقق من وجود أي كلمة من القائمةفي بداية الجملة. إذا كانت موجودة، فإن الجملة هي سؤال، وإذا لم تكن موجودة، فإن الجملة ليست سؤالًا.

الآن فيما يلي كيفية اكتشاف الأسئلة باستخدام بايثون باتباع المنطق الموضحفي القسم أعلاه:

```
from nltk.tokenize import word_tokenize
question_words = ["what", "why", "when", "where ,"
"              name", "is", "how", "do", "does ,"
"              which", "are", "could", "would ,"
"              should", "has", "have", "whom", "whose",
"don't["

question = input("Input a sentence: ")
question = question.lower()
question = word_tokenize(question)

if any(x in question[0] for x in question_words):
    print("This is a question!")
else:
    print("This is not a question!")
```

```
Input a sentence: Do you have any feelings for me?
This is a question!
```

هذه هي الطريقة التي يمكنك من خلالها اكتشاف ما إذا كانت الجملة عبارة عن سؤال أم لا.

ملخص

آمل أن تكون قد فهمت الآن كيفية تحديد الأسئلة أو جمل الاستفهام باستخدام بايثون. يمكننا أيضًا استخدام التعلم الآلي للكشف عن الأسئلة، ولكن كما نعلم جميعًا نوع الجمل التي نراهافي جملة الاستفهام، لذلك من الممكن أيضًا كتابة نص برمجي بايثون لاكتشاف ما إذا كانت الجملة عبارة عن سؤال أم لا. آمل أن تكون قد أحببت هذه المقالة حول كيفية اكتشاف الأسئلة باستخدام بايثون.

27) مسجل الصوت باستخدام بايثون Voice Recorder using Python

يوجد مسجل صوتفي كل هاتف ذكي وجهاز كمبيوتر اليوم. هو تطبيق يتم استخدامه لتسجيل الصوت وحفظه بتنسيق ملف معين يمكن الاستماع إليه ونقله إلى جهاز آخر. إذا كنت تريد معرفة كيفية إنشاء مسجل صوت باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية إنشاء مسجل صوت باستخدام بايثون.

مسجل الصوت باستخدام بايثون

يجب أن تكون قد استخدمت مسجل صوت على هاتفك الذكي أو جهاز الكمبيوتر الخاص بك مرة واحدةفي حياتك. نستخدمه بشكل عام لتسجيل رسالة صوتية، ويستخدمه بعض منشئي الفيديو لتسجيل الصوت لمقاطع الفيديو الخاصة بهم. لإنشاء مسجل صوت باستخدام لغة برمجة بايثون، تحتاج إلى استخدام مكتبة sounddevice في بايثون. إذا لم تكن قد استخدمت هذه المكتبة من قبل، فيمكنك تثبيتها بسهولة باستخدام الأمر (pip)المذكور أدناه:

```
pip install sounddevice
```

ستساعدك مكتبة sounddevice على تسجيل صوتك، ولكن لحفظ صوتك بتنسيق ملف معين، تحتاج إلى استخدام مكتبة (SciPy)في بايثون، والتي يمكن تثبيتها باستخدام أمر pip:

```
pip install SciPy
```

الآن فيما يلي كيفية إنشاء مسجل صوت باستخدام بايثون:

```
import sounddevice
from scipy.io.wavfile import write

def voice_recorder(seconds, file):
    print("Recording Started...")
    recording = sounddevice.rec((seconds * 44100), samplerate=
44100, channels=2)
    sounddevice.wait()
    write(file, 44100, recording)
    print("Recording Finished")

voice_recorder(10, "record.wav")
```

في الكود أعلاه، قمت بتحديد دالة بايثون لتسجيل وحفظ الملفات المسجلة. يتطلب معلمتين:

1. المعلمة الأولى هي الثواني (seconds)، حيث ستدخل عدد الثواني التي تريد تسجيل صوتك فيها.
2. المعلمة الثانية هو الملف (file)، حيث ستدخل الاسم الذي تريد حفظ الملف المسجل به. على سبيل المثال، "voice.wav".

بعد تشغيل الكود أعلاه، سيظهر لك رسالة مفادها أن التسجيل قد بدأ، وبعد عدد الثواني التي قدمتها كمدخل، سيظهر لك أن التسجيل قد اكتمل. بمجرد اكتماله، سيتم حفظه تلقائيًافي نفس الدليل حيث يوجد ملف بايثون الخاص بك.

ملخص

يوجد مسجل صوتفي كل هاتف ذكي وجهاز كمبيوتر اليوم. هو تطبيق يتم استخدامه لتسجيل الصوت وحفظه بتنسيق ملف معين يمكن الاستماع إليه ونقله إلى جهاز آخر. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء مسجل صوت باستخدام لغة برمجة بايثون.

28) قراءة وكتابة ملفات CSV باستخدام بايثون Reading and Writing CSV Files using Python

ملفات CSV (قيم مفصولة بفواصل Comma Separated Values) هي أكثر تنسيقات الملفات استخدامًا لاستيراد وتصدير مجموعات البيانات الكبيرة. السببفي تفضيل ملف CSV على ملف Excel هو أن ملف CSV يستهلك ذاكرة أقل مقارنة بملف Excel. لذلك أثناء تعلم علم البيانات، يجب أن تعرف كيفية قراءة ملفات CSV وكتابتها. إذا كنت تريد معرفة كيفية قراءة ملفات CSV وكتابتها، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول قراءة وكتابة ملفات CSV باستخدام بايثون.

قراءة وكتابة ملفات CSV باستخدام بايثون

يمكنك قراءة وكتابة ملف CSV دون استخدام أي وحدة بايثون أو مكتبة. ولكن نظرًا لأننا نستخدم مكتبة pandas للعمل مع البيانات، ستعلمك هذه المقالة كيفية قراءة وكتابة ملف CSV باستخدام مكتبة pandasفي بايثون.

فلنبدأ بكتابة ملف CSV. سأقوم هنا أولاً بإنشاء نموذج بيانات باستخدام قاموس بايثون حول اسم وعمر الطلاب، وبعد ذلك سأخزن قاموس بايثون هذافي ملف CSV:

```
 #writing a csv file
import csv
import pandas as pd
data = {"Name": ["Aman", "Diksha", "Akanksha", "Sajid",
"Akshit ,["
"         Age{[22 ,23 ,25 ,21 ,23] :"
data = pd.DataFrame(data)
data.to_csv("age_data.csv", index=False)
print(data.head())
```

```
      Name  Age
0     Aman   23
1   Diksha   21
2 Akanksha   25
3    Sajid   23
4   Akshit   22
```

هذه هي الطريقة التي يمكنك بها كتابة ملف CSV باستخدام بايثون. الآن فيما يلي كيفية قراءة ملف CSV هذا باستخدام بايثون:

```
 #reading a csv file
import pandas as pd
data = pd.read_csv("age_data.csv")
print(data.head())
```

```
      Name  Age
0     Aman   23
1   Diksha   21
2 Akanksha   25
3    Sajid   23
4   Akshit   22
```

هذا هو مدى سهولة قراءة وكتابة ملف CSV باستخدام مكتبة (pandas)في بايثون.

الملخص

هذه هي الطريقة التي يمكنك بها قراءة ملفات CSV وكتابتها باستخدام لغة برمجة بايثون. ملفات CSV هي تنسيق الملفات الأكثر استخدامًا لاستيراد وتصدير مجموعات البيانات. لذلك أثناء تعلم علم البيانات، يجب أن تعرف كيفية قراءة ملفات CSV وكتابتها. أتمنى أن تكون قد أحببت هذه المقالةفي برنامج تعليمي حول قراءة وكتابة ملفات CSV باستخدام بايثون.

29) المخطط الصندوقي باستخدام بايثون Box Plot using Python

المخطط الصندوقي (Box Plot) هو تقنية تصور البيانات الإحصائية لتحليل توزيع وأنماط نقاط البيانات الرقمية لمجموعة البيانات. وهو يمثل الربع 1 (quartile 1) والربع 3 (quartile 3) والوسيط (median) والحد الأقصى والحد الأدنى من نقاط البيانات لميزة مما يساعد على فهم توزيع القيم العددية لمجموعة البيانات. إذا كنت تريد معرفة كيفية تصور المخطط الصندوقي، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية تصوير المخطط الصندوقي باستخدام لغة برمجة بايثون.

المربع الصندوقي

يحتوي جزء الصندوقفي المخطط على ثلاثة أسطر:

1. يمثل السطر الأولفي الجزء العلوي الربع الثالث من نقاط البيانات، مما يعني أن 75٪ من البيانات تقع أسفل هذه النقطة؛
2. يمثل السطر الثانيفي المنتصف القيمة المتوسطة لنقاط البيانات، مما يعني أن 50٪ من البيانات تقع أسفل هذه النقطة؛
3. يمثل السطر الثالثفي مخطط الصندوق الربع الأول من نقاط البيانات، مما يعني أن 25٪ من البيانات تقع أسفل هذه النقطة؛
4. يُعرف الخطان الأفقيان الموجودان أسفل الصندوق وفوقه بخطوط الشعيرات (whisker lines)، ويمثل الخط الطولي أعلاه القيمة القصوى، ويمثل الطولي السفلي الحد الأدنى للقيمة.

أتمنى أن تكون قد فهمت الآن ما يُظهره لك المخطط الصندوقي حول السمة العددية لمجموعة البيانات. الآنفي القسم أدناه، سوف أطلعك على كيفية تصوير مخطط مربع باستخدام بايثون.

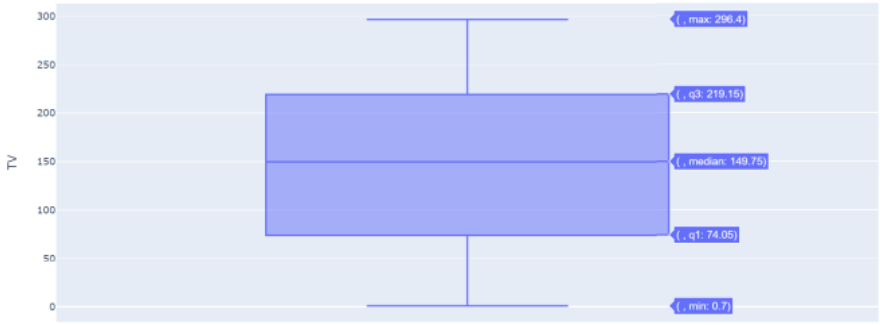
المخطط الصندوقي باستخدام بايثون

```
import pandas as pd
data =
pd.read_csv("https://raw.githubusercontent.com/amankharwal/Web
site-data/master/Advertising.csv")
print(data.head())
```

	Unnamed: 0	TV	Radio	Newspaper	Sales
0	1	230.1	37.8	69.2	22.1
1	2	44.5	39.3	45.1	10.4
2	3	17.2	45.9	69.3	9.3
3	4	151.5	41.3	58.5	18.5
4	5	180.8	10.8	58.4	12.9

الآن فيما يلي كيف يمكنك تصور المخطط الصندوقي باستخدام لغة برمجة بايثون:

```
import plotly.express as px
fig = px.box(data, y="TV")
fig.show()
```



هذه هي الطريقة التي يمكنك من خلالها تصوير المخططات الصندوقية بسهولة باستخدام بايثون.

الملخص

يمثل المخطط الصندوقي الربع 1 والربع 3 والوسيط والحد الأقصى والحد الأدنى لنقاط البيانات لميزة مما يساعد على فهم توزيع الميزات العددية لمجموعة البيانات. آمل أن تكون قد أحببت هذه المقالة حول تصوير المخطط الصندوقي باستخدام بايثون.

30) أرسال رسائل Instagram باستخدام بايثون Sending Instagram Messages using Python

باستخدام تطبيق جهة خارجية أو واجهة برمجة تطبيقات لإدارة وظائف أحد التطبيقات، فأنت تقوم بأتمتة التطبيق. إذا كنت ترسل رسائل أو تنشر صورًا أو مقاطع فيديو أو تتابع شخصًا ما دون فتح Instagram الخاص بك مباشرةً، فهذا يعني أنك تقوم بأتمتة Instagram. لذلك، إذا كنت تريد معرفة كيفية إرسال رسائل Instagram تلقائيًا باستخدام بايثون، فهذه المقالة مناسبة لك. في هذه المقالة، سأقدم برنامجًا تعليميًا حول كيفية إرسال رسائل Instagram باستخدام بايثون.

أرسال رسائل Instagram باستخدام بايثون

لإرسال رسالة Instagram باستخدام بايثون يجب أن يكون لديك حساب Instagram ومكتبة (instabot) مثبتةفي بيئة بايثون الافتراضية الخاصة بك. Instabot عبارة عن مكتبة بايثون يمكنك استخدامها لأتمتة ميزات حساب Instagram الخاص بك باستخدام بايثون ، مثل إرسال الرسائل دون فتح التطبيق الخاص بك. يمكنك تثبيت مكتبة بايثون هذه باستخدام الأمر pip المذكور أدناه:

```
pip install instabot
```

آمل أن تكون قد قمت الآن بتثبيت مكتبة instabotفي بايثون، والآن فيما يلي كيفية إرسال رسائل Instagram باستخدام بايثون:

```
from instabot import Bot
bot = Bot()

bot.login(username="Your Username", password="Your Password")
bot.send_message("Hi Brother", ["Receiver's Username"])
```

في الكود أعلاه، أستخدم دالة تسجيل الدخول لمكتبة instabot لتسجيل الدخول إلى حساب Instagram الخاص بي باستخدام بايثون. هنا يجب عليك استخدام معلمة اسم المستخدم (username) لإدخال اسم المستخدم ومعلمة كلمة المرور (password) لإدخال كلمة المرور الخاصة بك. بعد ذلك،في السطر التالي، أستخدم دالة send_message لإرسال الرسالة حيث يكون المعلمة الأولى هي الرسالة نفسها والمعلمة الثانية هي اسم مستخدم حساب Instagram الذي تريد إرسال رسالة إليه.

الملخص

هذه هي الطريقة التي يمكنك بها إرسال رسالة Instagram باستخدام بايثون. إذا كنت ترسل رسائل أو تنشر صورًا أو مقاطع فيديو أو تتابع شخصًا ما دون فتح Instagram الخاص بك

مباشرةً، فهذا يعني أنك تقوم بأتمتة Instagram. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إرسال رسالة Instagram باستخدام بايثون.

31) ايجاد المضاعف المشترك الأصغر في بايثون Finding LCM using Python

يرمز LCM إلى المضاعف المشترك الأصغر (Least Common Multiple)، مما يعني العثور على أصغر رقم يكون مضاعفًا لرقمين أو أكثر. إذا كنت تريد معرفة كيفية العثور على LCM لرقمين باستخدام لغة برمجة بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم برنامجًا تعليميًا حول كيفية العثور على LCM باستخدام بايثون.

المضاعف المشترك الأصغر باستخدام بايثون

إيجاد المضاعف المشترك الأصغر لرقمين يعني إيجاد أصغر رقم يكون مضاعفًا لكلا العددين. تحتوي بايثون على العديد من الدوال المضمنة التي يمكنك استخدامهافي العمليات الحسابية، ولكن لسوء الحظ، لا تحتوي على أي دالة لحساب المضاعف المشترك الأصغر لرقمين أو أكثر. لذلك لحساب المضاعف المشترك الأصغر لرقمين باستخدام بايثون، عليك تحديد دالة بايثون الخاصة بك. فيما يلي كيفية العثور على المضاعف المشترك الأصغر لرقمين باستخدام بايثون:

```
def least_common_multiple(a, b):
    if a > b:
        greater = a
    elif b > a:
        greater = b
    while(True):
        if ((greater % a == 0) and (greater % b == 0)):
            lcm = greater
            break
        greater = greater + 1
    return lcm

print(least_common_multiple(10, 12))
```

```
60
```

في الكود أعلاه، قمت بتعريف دالة بايثون، حيث استخدمت معلمتين كـ a و b. ثم أجد الرقم الكبير بين الرقمين وأقسم الرقم الأكبر على كلا الرقمينفي حلقة while حيث ستزداد قيمة الرقم الأكبر بمقدار 1 حتى نحصل على 0 كبقية. هذه هي الطريقة التي يمكنك بها إيجاد المضاعف المشترك الأصغر لرقمين باستخدام لغة برمجة بايثون.

الملخص

هذه هي الطريقة التي يمكنك بها إيجاد المضاعف المشترك الأصغر لرقمين باستخدام بايثون. إيجاد المضاعف المشترك الأصغر لرقمين يعني إيجاد أصغر رقم يكون مضاعفًا لكلا الرقمين.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية العثور على المضاعف المشترك الأصغر لرقمين باستخدام بايثون.

32) مرونة السعر للطلب باستخدام بايثون Price Elasticity of Demand using Python

السعر (Price) هو أحد أهم العوامل التي تؤثر على الطلب على المنتج. تشير المرونة (Elasticity) إلى درجة الاستجابة، والمرونة السعرية للطلب تشير إلى درجة استجابة الطلب على المنتج بسبب التغيرفي سعره. ندرس مرونة الطلب السعريةفي الاقتصاد، ولكن إذا كنت تريد معرفة كيفية حساب مرونة الطلب السعرية باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم لك برنامجًا تعليميًا حول حساب مرونة الطلب السعرية (price elasticity of demand) باستخدام بايثون.

مرونة سعر الطلب

تشير مرونة الطلب السعرية إلى درجة استجابة الطلب على المنتج للتغيرفي السعر. ببساطة، يعني ذلك الدرجة التي يتغير بها الطلب على المنتج مع زيادة أو نقصان سعره. على سبيل المثال، يزداد الطلب على منتج ما بنسبة 20٪ بسبب انخفاض سعره بنسبة 10٪. هذا ما يعنيه التغييرفي الطلب مع التغيرفي سعر المنتج. وعندما تحسب الدرجة التي يتغير فيها الطلب، فإنها تسمى مرونة الطلب السعرية.

لحساب مرونة الطلب السعرية، عليك استخدام الصيغة المذكورة أدناه:

```
Percentage Change in Quantity Demanded / Percentage Change in
the Price
```

أتمنى أن تكون قد فهمت الآن معنى مرونة السعر وكيفية حسابها. الآنفي القسم أدناه، سوف آخذك من خلال برنامج تعليمي حول كيفية حساب مرونة الطلب السعرية باستخدام بايثون.

مرونة سعر الطلب باستخدام بايثون

سأبدأ مهمة حساب مرونة الطلب السعرية باستخدام بايثون من خلال إنشاء مجموعة بيانات صغيرة يجب أن تحتوي على بيانات حول التغييرفي السعر والطلب على المنتج:

```
import pandas as pd
data = pd.DataFrame({"Demand": [20, 30, 31, 33, 30, 33, 35 ,[
"                    Price ,1800 ,1700 ,1850 ,1800 ,2000] :"
({[1600 ,1700
print(data)
```

```
  Demand  Price
0      20   2000
2      31   1850
3      33   1700
4      30   1800
5      33   1700
6      35   1600
```

تحتوي الصفوف الأولى من مجموعة البيانات هذه على الطلب والسعر المبدئيين للمنتج (initial demand and price)، وتحتوي الصفوف اللاحقة على التغييرفي الطلب والتغيرفي سعر المنتج. الآن خطوتنا التالية هي إضافة عمودين آخرين كتغير النسبة المئويةفي الطلب (Percentage change in Demand) وتغير النسبة المئويةفي السعر (Percentage change in Price) عن طريق حسابها:

```
data["% Change in Demand"] = data["Demand"].pct_change()
data["% Change in Price"] = data["Price"].pct_change()
print(data)
```

```
   Demand  Price  % Change in Demand  % Change in Price
0      20   2000                 NaN                NaN
1      30   1800            0.500000          -0.100000
2      31   1850            0.033333           0.027778
3      33   1700            0.064516          -0.081081
4      30   1800           -0.090909           0.058824
5      33   1700            0.100000          -0.055556
6      35   1600            0.060606          -0.058824
```

الآن الخطوة الأخيرة هي حساب مرونة الطلب السعرية (النسبة المئوية للتغيرفي الطلب / النسبة المئوية للتغيرفي السعر) عن طريق إضافة عمود جديد إلى هذه البيانات. فيما يلي كيف يمكنك حسابه باستخدام بايثون:

```
data["Price Elasticity"] = data["% Change in Demand"] /
data["% Change in Price"]
print(data)
```

```
   Demand  Price  % Change in Demand  % Change in Price  Price Elasticity
0      20   2000                 NaN                NaN               NaN
1      30   1800            0.500000          -0.100000         -5.000000
2      31   1850            0.033333           0.027778          1.200000
3      33   1700            0.064516          -0.081081         -0.795699
4      30   1800           -0.090909           0.058824         -1.545455
5      33   1700            0.100000          -0.055556         -1.800000
6      35   1600            0.060606          -0.058824         -1.030303
```

الملخص

تشير مرونة الطلب السعرية إلى درجة استجابة الطلب على المنتج للتغيرفي السعر. ببساطة، يعني ذلك الدرجة التي يتغير بها الطلب على المنتج مع زيادة أو نقصان سعره. أتمنى أن تكون قد أحببت هذا المقال حول كيفية حساب مرونة الطلب السعرية باستخدام بايثون.

33) ايجاد الكلمات الأكثر تكراراً في ملف Finding the Most Frequent Words in a File

يعد حساب عدد الكلمات المحددة (specific words)في الملف أمرًا تحتاج إلى معرفته كمبرمج. يعد حساب الكلمات الأكثر تكراراً في الملف أحد أسئلة البرمجة التي يمكنك حلهافي أي مقابلة برمجة. لذلك، إذا كنت تريد معرفة كيفية العثور على الكلمات الأكثر تكراراً في ملف ما، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج بايثون لحساب الكلمات الأكثر تكراراً في الملف.

ايجاد الكلمات الأكثر تكراراً في ملف باستخدام بايثون

تعد كتابة برنامج لحساب الكلمات الأكثر تكراراً في الملف سؤالًا مهمًافي مقابلة البرمجة يمكنك الحصول عليهفي أي مقابلة برمجة. يمكنك الحصول على أسئلة بناءً على هذا المنطق بعدة طرق. هنا سيتم إعطاؤك ملفًا، وسيُطلب منك العثور على الكلمات الأكثر تكراراً في هذا الملف بالإضافة إلى عدد مرات وجودها. إليك كيفية كتابة برنامج بايثون لحساب الكلمات الأكثر تكراراً في الملف:

```
words = []
with open("aman.txt", "r") as f:
    for line in f:
        words.extend(line.split())

from collections import Counter
counts = Counter(words)
top5 = counts.most_common(5)
print(top5)
```

```
[('the', 5), ('you', 5), ('Python', 4), ('is', 4), ('of', 3)]
```

في الكود أعلاه، أقرأ أولاً ملفًا نصيًا من جهاز الكمبيوتر الخاص بي، ثم أقوم بتقسيم جميع الكلمات وتخزينهافي قائمة بايثون. ثم أحسب تكرار جميع الكلماتفي القائمة باستخدام طريقة (Counter) لوحدة التجميعفي بايثون.في النهاية، أقوم بطباعة الكلمات الخمس الأكثر تكراراً في الملف.

الملخص

هذه هي الطريقة التي يمكنك بهاكتابة برنامج لحساب الكلمات الأكثر تكراراً من أي ملف. تعد كتابة برنامج لحساب الكلمات الأكثر تكراراً في الملف سؤالًا مهمًافي مقابلة البرمجة يمكنك الحصول عليهفي أي مقابلة برمجة. يمكنك الحصول على أسئلة بناءً على هذا المنطق بعدة طرق.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية كتابة برنامج بايثون لحساب الكلمات الأكثر تكراراً في الملف.

34) ايجاد عدد الأحرف الكبيرة في ملف Finding the Number of Capital Letters in a File

يعد حساب عدد الأحرف المحددة من ملف أمرًا يجب أن يعرفه كل مبرمج. من أكثر أسئلة مقابلة البرمجة شيوعًا بناءً على هذا المنطق هو حساب عدد الأحرف الكبيرة من الملف. لذلك إذا كنت تريد معرفة كيفية حساب عدد الأحرف الكبيرة (count the number of capital letters) من خلال قراءة أي ملف، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك خلال برنامج تعليمي حول كيفية كتابة برنامج بايثون لحساب الأحرف الكبيرةفي ملف.

ايجاد عدد الأحرف الكبيرة في ملف

تعد كتابة برنامج لحساب عدد الأحرف الكبيرةفي ملف سؤال برمجة مهمًا يمكنك الحصول عليه في أي مقابلة برمجة. يمكنك الحصول على أسئلة بناءً على هذا المنطق بعدة طرق. سيتم إعطاؤك ملفًا نصيًا، وسيُطلب منك قراءة الملف دون استخدام مكتبة بايثون وطباعة عدد الأحرف الكبيرة أو الصغيرةفي الملف النصي. إليك كيفية كتابة برنامج بايثون لحساب الأحرف الكبيرةفي ملف نصي:

```
with open("text.txt") as file:
    count = 0
    text = file.read()
    for i in text:
        if i.isupper():
            count += 1
    print(count)
```

```
21979
```

في الكود أعلاه:

1. فتحت لأول مرة ملفًا نصيًا تم حفظه بالفعل على جهاز الكمبيوتر الخاص بي.
2. ثم قدمت متغيرًا كعدد، والذي يستخدم هنا لتخزين عدد الأحرف الكبيرة.
3. في البداية، أعلنت أن قيمتها تساوي 0، وفي السطر التالي، أقرأ الملف النصي.
4. ثم أستخدم حلقة for فوق محتوى الملف النصي وأستخدم عبارة if داخل الحلقة for والتي ستستمرفي إضافة 1 إلى متغير العد حتى يستمرفي العثور على الأحرف الكبيرةفي الملف باستخدام دالة ()isupperفي بايثون.

تمامًا مثل الطريقة أعلاه، يمكنك كتابة برنامج بايثون لحساب الأحرف الصغيرة أيضًا عن طريق استبدال دالة ()isupper بـ ()islower كما هو موضحفي الكود أدناه:

```
with open("text.txt") as file:
    count = 0
    text = file.read()
    for i in text:
        if i.islower():
            count += 1
    print(count)
```

```
652265
```

الملخص

هذه هي الطريقة التي يمكنك بها العثور على جميع الأحرف الكبيرةفي ملف باستخدام لغة برمجة بايثون. إنه سؤال برمجة مهم يمكنك الحصول عليهفي أي مقابلة برمجة. يمكنك الحصول على أسئلة بناءً على هذا المنطق بعدة طرق. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية كتابة برنامج بايثون لحساب عدد الأحرف الكبيرةفي ملف.

35) فهرس القيمة الكبرى في قائمة بايثون Index of Maximum Value in a Python List

يعد العثور على فهرس القيمة الكبرىفي قائمة أو مصفوفة أحد أكثر أسئلة مقابلة البرمجة شيوعًا التي يمكنك الحصول عليهافي أي مقابلة برمجة. هنا عليك أن تجد فهرس القيمة الكبرى بدلاً من القيمة الكبرى نفسها. لذلك إذا كنت تريد معرفة كيفية العثور على فهرس القيمة الكبرىفي قائمة بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية العثور على فهرس القيمة الكبرىفي قائمة بايثون.

فهرس القيمة الكبرى في قائمة بايثون

تحتوي لغة بايثون على دالة مضمنة للعثور على القيمة الكبرىفي القائمة. يمكنك أيضًا تحديد دالة بايثون الخاصة بك لنفس المهمة. لكن السؤال هنا هو إيجاد فهرس القيمة الكبرى بدلاً من القيمة نفسها. إذن إليك كيفية تحديد دالة بايثون للعثور على فهرس القيمة الكبرىفي قائمة بايثون:

```
def maximum(x):
    maximum_index = 0
    current_index = 1
    while current_index < len(x):
        if x[current_index] > x[maximum_index]:
            maximum_index = current_index
        current_index = current_index + 1
    return maximum_index
a = [23, 76, 45, 20, 70, 65, 15, 54]
print(maximum(a))
```

```
Output: 1
```

تقوم الدالة المذكورة أعلاه بتنفيذ خوارزمية للعثور على فهرس القيمة الكبرى. يفترض أن القائمة ليست فارغة وأن العناصر الموجودةفي القائمة بترتيب عشوائي. يبدأ بمعاملة العنصر الأولفي القائمة باعتباره العنصر الاكبر، ثم يبحث عن أكبر عنصر على اليمين، إذا وجد عنصرًا أكبر من الأول، فإنه يعيد تعيين موضع العنصر الاكبر. عندما تصل إلى العنصر الأخيرفي القائمة باتباع نفس العملية، فإنها تُرجع فهرس القيمة الكبرى.

الملخص

هذه هي الطريقة التي يمكنك بها العثور على فهرس الحد الأقصى للعنصرفي قائمة بايثون. إنه أحد أسئلة مقابلة البرمجة الشائعة التي قد تحصل عليهافي أي مقابلة برمجة. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية العثور على فهرس القيمة القصوى باستخدام لغة برمجة بايثون.

36) فهرس القيمة الصغرى في قائمة بايثون Index of Minimum Value in a Python List

يعد العثور على مؤشر القيمة الصغرىفي قائمة أو مصفوفة أحد أسئلة مقابلة البرمجة الشائعة التي قد تحصل عليهافي أي مقابلة برمجة. هنا عليك أن تجد فهرس القيمة الصغرى بدلاً من القيمة الصغرى. لذلك إذا كنت تريد معرفة كيفية العثور على فهرس القيمة الصغرىفي قائمة بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية العثور على فهرس القيمة الصغرىفي قائمة بايثون.

فهرس القيمة الصغرى في قائمة بايثون

تحتوي لغة بايثون على دالة مضمنة للعثور على القيمة الصغرىفي القائمة. يمكنك أيضًا تحديد دالة بايثون لنفسها، ولكن السؤال هنا هو العثور على فهرس القيمة الصغرى ، حيث يكون فهرس القيمة الأولى هو 0. لذا إليك كيفية تحديد دالة بايثون للعثور على فهرس القيمة الصغرىفي قائمة بايثون:

```
def minimum(x):
    minimum_index = 0
    current_index = 1
    while current_index < len(x):
        if x[current_index] < x[minimum_index]:
            minimum_index = current_index
        current_index = current_index + 1
    return minimum_index
a = [23, 76, 45, 20, 70, 65, 15, 54]
print(minimum(a))
```

```
Output: 6
```

تقوم الدالة المذكورة أعلاه بتنفيذ خوارزمية للعثور على فهرس القيمة الصغرى. يفترض أن القائمة ليست فارغة وأن العناصر الموجودةفي القائمة بترتيب عشوائي. يبدأ بمعاملة العنصر الأول من القائمة باعتباره العنصر الأدنى، ثم يبحث عن العنصر الأصغر إلى اليمين، ثم إذا عثر على عنصر أصغر مقارنة بالعنصر الأول، فإنه يعيد تعيين موضع العنصر الأدنى. عندما تصل إلى العنصر الأخيرفي القائمة، باتباع نفس العملية، فإنها تُرجع فهرس الحد الأدنى للعنصر.

الملخص

هذه هي الطريقة التي يمكنك بها العثور على فهرس القيمة الصغرىفي قائمة بايثون. إنه أحد أسئلة مقابلة البرمجة الشائعة التي قد تحصل عليهافي أي مقابلة برمجة. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية العثور على فهرس القيمة الصغرى باستخدام لغة برمجة بايثون.

37) المخطط المبعثر المتحرك باستخدام بايثون Animated Scatter Plot using Python

يعد المخطط المبعثر (Scatter Plot) أحد أكثر الطرق المفيدة لتحليل العلاقة بين ميزتين. يجب أن تكون قد استخدمت مخطط مبعثر من قبل إذا كنت تتعلم علم البيانات ولكن هل سبق لك أن حاولت إنشاء مخطط مبعثر متحرك باستخدام بايثون؟ إذا كنت تريد معرفة كيفية تصوير مخطط مبعثر متحرك (animated scatter plot)، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول تصور مخطط مبعثر متحرك باستخدام بايثون.

المخطط المبعثر المتحرك باستخدام بايثون

عندما تريد تحليل كيفية تأثر ميزة واحدةفي مجموعة بيانات بميزة أخرى، فربما تحتاج إلى استخدام مخطط مبعثر. يمكنك معرفة المزيد حول المخططات المبعثرة وكيفية تصويرها باستخدام بايثون من هنا. الآن بالعودة إلى المخططات المبعثرة المتحركة، يمكن استخدام مكتبة بايثون الرسومية لإنشاء رسوم بيانية متحركة، حتى نتمكن من استخدام مكتبة (Plotly) لتصوير مخطط مبعثر باستخدام بايثون.

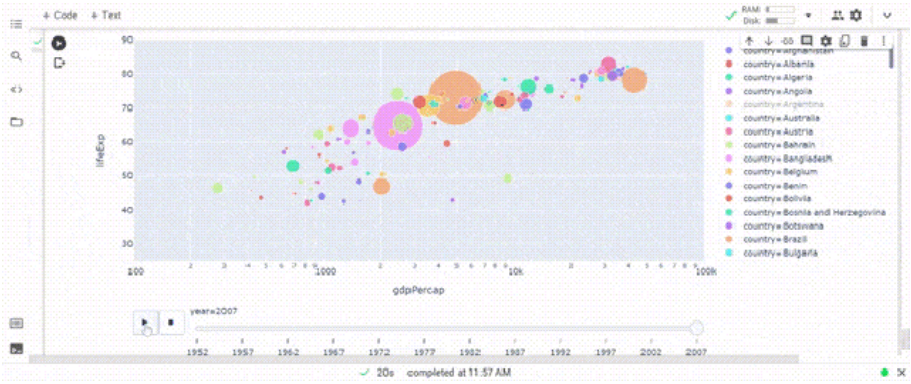
الآن فيما يلي كيف يمكنك تصوير مخطط مبعثر باستخدام مكتبة الرسمفي بايثون:

```
import plotly.express as px
data = px.data.gapminder()
print(data.head())
```

```
       country continent  year  ...   gdpPercap  iso_alpha  iso_num
0  Afghanistan      Asia  1952  ...  779.445314        AFG        4
1  Afghanistan      Asia  1957  ...  820.853030        AFG        4
2  Afghanistan      Asia  1962  ...  853.100710        AFG        4
3  Afghanistan      Asia  1967  ...  836.197138        AFG        4
4  Afghanistan      Asia  1972  ...  739.981106        AFG        4

[5 rows x 8 columns]
```

```
px.scatter(data, x="gdpPercap", y="lifeExp",
animation_frame="year", animation_group="country,"
           size="pop", color="country", hover_name="country,"
           log_x=True, size_max=55, range_x=[100,100000],
range_y=[25,90([
```

في الكود أعلاه، أستخدم مجموعة بيانات مقدمة من مكتبة (Plotly) نفسها. تغطي مجموعة البيانات نصيب الفرد من الناتج المحلي الإجمالي لجميع البلدان. أنا هنا أتخيل نصيب الفرد من الناتج المحلي الإجمالي للبلدان على المحور السيني ومتوسط العمر المتوقع على المحور الصادي. بمجرد تشغيل هذا الكود، سترى الإخراج المتحرك كما هو موضح أدناه.



الملخص

هذه هي الطريقة التي يمكنك من خلالها تصوير الأشكال المتحركة باستخدام مكتبة (Plotly) في بايثون. تعد مكتبة (Plotly)في بايثون أداة مفيدة للغاية يمكن استخدامها لإنشاء رسوم بيانية متحركة باستخدام بايثون. آمل أن تكون قد أحببت هذه المقالة حول كيفية تصوير مخططات مبعثرة متحركة باستخدام بايثون.

38) إنشاء خط فني باستخدام بايثون Creating Font Art using Python

الخط الذي يظهرفي إخراج برنامج بايثون هو الخط الافتراضي لنظام التشغيل الخاص بك. قد لا يكون تغيير خط الإخراج ممكنًا بدون استخدام مكتبة خارجية. هذا هو المكان الذي يمكن أن تكون فيه مكتبة (PyFiglet)في بايثون مفيدة. لذلك إذا كنت تريد معرفة كيفية طباعة مخرجاتك بخطوط مذهلة، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية إنشاء فن الخط باستخدام بايثون.

إنشاء خط فنيباستخدام بايثون

يمكن استخدام مكتبة (PyFiglet)في بايثون لتصوير إخراج برنامج بايثون الخاص بك بنمط خط مذهل. إذا لم تكن قد استخدمت هذه المكتبة من قبل، فيمكنك تثبيتها بسهولة باستخدام الأمر pip:

```
pip install pyfiglet
```

الآن دعونا نرى كيفية كتابة برنامج بايثون لإنشاء خطوط مذهلة. سأقوم هنا بطباعة اسمي كمخرج في font artمذهل باستخدام بايثون:

```
import pyfiglet
font = pyfiglet.figlet_format('Aman Kharwal')
print(font)
```

```
    _                         _  __ _                            _ 
   / \   _ __ ___   __ _ _ __ | |/ /| |__   __ _ _ ____      ____ _| |
  / _ \ | '_ ` _ \ / _` | '_ \| ' / | '_ \ / _` | '__\ \ /\ / / _` | |
 / ___ \| | | | | | (_| | | | | . \ | | | | (_| | |   \ V  V / (_| | |
/_/   \_\_| |_| |_|\__,_|_| |_|_|\_\|_| |_|\__,_|_|    \_/\_/ \__,_|_|
```

الخط أعلاه هو نمط الخط الافتراضي الذي توفره مكتبة (PyFiglet)في بايثون. إذا كنت تريد تغيير خط الإخراج، فيمكنك تغييره باستخدام معلمة الخط ،) (pyfiglet.figlet_format('Aman Kharwal', **font**="font name")). يمكنك العثور على العديد من أنماط الخطوط التي تدعمها هذه المكتبة هنا.

الملخص

هذه هي الطريقة التي يمكنك بها تغيير نمط الخط لمخرجات برنامج بايثون الخاص بك عن طريق إنشاء خطوط فنية باستخدام مكتبة PyFigletفي بايثون. قد لا يكون تغيير خط الإخراج ممكنًا بدون استخدام مكتبة خارجية. هذا هو المكان الذي يمكن أن تكون فيه مكتبة PyFiglet

في بايثون مفيدة. آمل أن تكون قد أحببت هذه المقالة حول كيفية إنشاء خط فني مذهل باستخدام بايثون.

39) مجمع الصور باستخدام بايثون Collage Maker using Python

مجمع الصور (Collage Maker) هو أداة تستخدم لدمج الصور المختلفةفي صورة واحدة. يسمح لك بحفظ ومشاركة الصور كمجموعة من الذكريات. يجب أن تكون قد رأيت خيارًا لإنشاء صورة مجمعة على هاتفك الذكي نفسه، ولكن إذا كنت تريد معرفة كيفية إنشاء صورة مجمعة باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية إنشاء مجمع الصور باستخدام بايثون.

مجمع الصور باستخدام بايثون

لإنشاء مجمع صور باستخدام لغة برمجة بايثون، يجب أن تتعلم أولاً كيفية قراءة وتحويل الصور إلى مصفوفات باستخدام بايثون. لإنشاء صورة مجمعة، تحتاج أولاً إلى قراءة الصور وتحويلها إلى مصفوفات قبل دمجهافي صورة مجمعة. يمكنك تعلم كل شيء عن تحويل الصور إلى مصفوفات باستخدام بايثون من هنا. الآن إليك كيفية كتابة برنامج بايثون لإنشاء مجمع الصور:

```
from PIL import Image
import numpy as np
def collage_maker(image1, image2, name):
    i1 = np.array(image1)
    i2 = np.array(image2)
    collage = np.vstack([i1, i2])
    image = Image.fromarray(collage)
    image.save(name)

# To Run The Above Function
collage_maker("image1.jpg", "image2.jpg", "new.jpg")
```

في الكود أعلاه، قمت أولاً باستيراد فئة الصور من مكتبة (Pillow)في بايثون، والمعروفة أيضًا باسم مكتبة صور بايثون ((Python Image Library (PIL). ثمفي السطر التالي، قمت باستيراد مكتبة (NumPy)التي سأستخدمها لتحويل الصور إلى مصفوفات. ثمفي السطر التالي، قمت بتعريف دالة بايثون لالتقاط صورتين كمعلمات ويتم استخدام المعلمة الثالثة لأخذ الاسم الذي تريد حفظ الصورة المجمعة من الصورتين.

لتشغيل هذا الكود، عليك أولاً إدخال اسم الصورتين اللتين تريد استخدامهما لإنشاء صورة مجمعةفي الدالة، ثم يتعين عليك إدخال الاسم الذي تريد حفظ الصورة به كمعامل ثالث. بمجرد تنفيذ الكود، سيحفظ مجمعة من صورفي نفس الدليل حيث يوجد ملف بايثون الخاص بك.

الملخص

هذه هي الطريقة التي يمكنك بها إنشاء مجمع الصور الخاص بك باستخدام لغة برمجة بايثون. يمكنك تعديله بعدة طرق، على سبيل المثال، إنشاء تطبيق واجهة المستخدم الرسومية (GUI) يقبل صورتين أو أكثر من جهاز كمبيوتر ويحفظ صورة مجمعةفي موقعك المفضل. آمل أن تكون قد أحببت هذه المقالة حول كيفية إنشاء مجمع صور باستخدام بايثون.

40) تفاصيل رقم الهاتف باستخدام بايثون Phone Number Details using Python

هناك الكثير من المهام حيث نريد استخراج تفاصيل رقم هاتف الشخص. يمكن أن يساعد صناعة الاتصالات وكذلك الشركات الأخرى التي تتعامل مع مشاكل الناس مع الشبكات. إذا كنت تريد معرفة كيفية الحصول على تفاصيل رقم هاتف باستخدام لغة برمجة بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سأعرض لك برنامجًا تعليميًا حول كيفية الحصول على تفاصيل رقم الهاتف باستخدام بايثون.

تفاصيل رقم الهاتف باستخدام بايثون

للحصول على تفاصيل أي رقم، يمكننا استخدام وحدة بايثون المذهلة المعروفة باسم (phonenumbers). تم إنشاء هذه الوحدة بواسطة David Drysdale ويمكنك استخدامها للحصول على تفاصيل أي رقم هاتف من أي مكانفي العالم.

لتثبيت وحدة بايثون هذه على نظامك، يمكنك استخدام الأمر pip على التيرمينال أو موجه الأوامر المذكور أدناه:

```
pip install phonenumbers
```

هناك الكثير من التفاصيل التي يمكنك العثور عليها حول رقم باستخدام وحدة بايثون هذه. إليك كيفية العثور على بعض التفاصيل الأساسية حول رقم هاتف باستخدام بايثون:

```
import phonenumbers as ph
from phonenumbers import carrier
from phonenumbers import geocoder
from phonenumbers import timezone

number = "+9185XXXXXXXX"
number = ph.parse(number)
print(timezone.time_zones_for_number(number))
print(carrier.name_for_number(number, "en"))
print(geocoder.description_for_number(number, "en"))
```

هذه هي الطريقة التي يمكنك استخدامها للعثور على بعض التفاصيل الأساسية لرقم الهاتف. يمكنك قراءة المزيد حول هذه الوحدة من هنا للعثور على مزيد من التفاصيل المعقدة حول رقم الهاتف.

الملخص

هذه هي الطريقة التي يمكنك بها استخدام وحدة (phonenumbers)في بايثون للعثور على تفاصيل رقم الهاتف لأي رقم باستخدام لغة برمجة بايثون. يمكن أن يساعد صناعة الاتصالات

وكذلك الشركات الأخرى التي تتعامل مع مشاكل الناس مع الشبكات. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية الحصول على تفاصيل أي رقم اتصالفي العالم باستخدام بايثون.

41) طباعة التقويم باستخدام بايثون Printing a Calendar using Python

توفر وحدة (calendar)في بايثون الوصول إلى التقويم الخاص بأي شهر من أي عام. إذا لم تكن قد استخدمت هذه الوحدة من قبل، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم برنامجًا تعليميًا حول كيفية كتابة برنامج لطباعة تقويم باستخدام بايثون.

طباعة التقويم باستخدام بايثون

هناك العديد من الوحدات النمطية المفيدة المضمنة (built-in modules)في بايثون والتي يمكن استخدامها لتحقيق هدفكفي بضعة أسطر من التعليمات البرمجية. إحدى هذه الوحدات النمطيةفي بايثون هي وحدة (calendar)التي توفر الوصول إلى التقويم الخاص بأي شهر من أي عام.في أي وقت تريد عرض تقويم للمستخدم، ستكون وحدة بايثون هذه مفيدة. الآن إليك كيف يمكنك بسهولة كتابة برنامج لطباعة تقويم باستخدام بايثون:

```
import calendar
print(calendar.month(2021, 9))
```

```
Mo Tu We Th Fr Sa Su
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30
```

في الكود أعلاه، قمت أولاً باستيراد وحدة التقويمفي بايثون. ثم استخدمت دالة print لطباعة تقويم سبتمبر 2021 باستخدام دالة (month)في وحدة التقويم. يمكنك أيضًا أن تأخذ مدخلات المستخدم كسنة وشهر، ثم يمكنك استخدام مدخلات المستخدم كمعلمات لدالة الشهر لوحدة (calendar).

الملخص

هذه هي الطريقة التي يمكنك بها كتابة برنامج لطباعة التقويم لأي شهر من أي عام باستخدام بايثونفي بضعة أسطر من التعليمات البرمجية.في أي وقت تريد عرض التقويم للمستخدم، ستكون وحدة (calendar) مفيدة. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إظهار التقويم باستخدام لغة برمجة بايثون.

43) اختبار سرعة الإنترنت باستخدام بايثون Internet Speed Test using Python

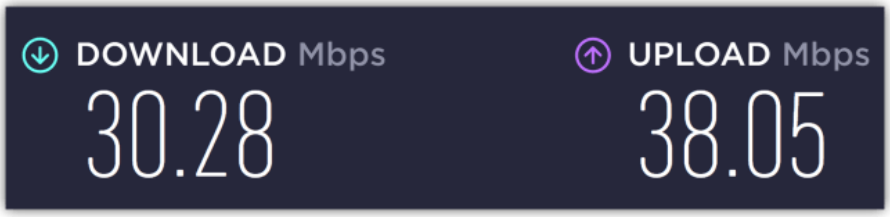
يجب أن تكون قد تحققت من سرعة اتصالك بالإنترنت مرة واحدةفي حياتك باستخدام منصات مثل speedtest.net. هل فكرت يومًافي فعل الشيء نفسه مع بايثون؟ حسنًا، إذا كنت تريد معرفة كيفية إجراء اختبار سرعة الإنترنت باستخدام بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم برنامجًا تعليميًا حول كيفية إجراء اختبار سرعة الإنترنت باستخدام بايثون.

اختبار سرعة الإنترنت

عند التحقق من سرعة الاتصال بالإنترنت، يتم عرض النتائج كسرعة التنزيل (Download speed) وسرعة التحميل (Upload speed). تشير سرعة التنزيل إلى السرعة التي يقوم بها اتصال الإنترنت بتنزيل البيانات من الإنترنت وتشير سرعة التحميل إلى السرعة التي يقوم بها اتصال الإنترنت الخاص بك بتحميل البيانات إلى الإنترنت. لذلك، فإن حساب سرعة التنزيل وسرعة التحميل للاتصال بالإنترنت يلخص اختبار سرعة الإنترنت.

على سبيل المثال، ألق نظرة على النتائج أدناه من اختبار سرعة اتصال الإنترنت الخاص بي وفقًا لاختبار السرعة بواسطة Ookla:

آمل أن تكون قد فهمت الآن اختبار سرعة الإنترنت.في القسم أدناه، سوف أطلعك على كيفية إجراء اختبار سرعة الإنترنت باستخدام بايثون.



اختبار سرعة الإنترنت باستخدام بايثون

لحساب سرعة اتصالك بالإنترنت باستخدام بايثون، عليك تثبيت مكتبة بايثون المعروفة باسم speedtest. إذا لم تستخدمه من قبل، فيمكنك تثبيته بسهولة على نظامك باستخدام أمر pip:

```
pip install speedtest-cli
```

الآن فيما يلي كيفية إجراء اختبار سرعة الإنترنت باستخدام بايثون:

```
import speedtest
wifi  = speedtest.Speedtest()
```

```
print("Wifi Download Speed is ", wifi.download())
print("Wifi Upload Speed is ", wifi.upload())
```

```
Wifi Download Speed is  33260529.44019052
Wifi Upload Speed is  28289053.5236785
```

الملخص

يلخص حساب سرعة التنزيل وسرعة التحميل للاتصال بالإنترنت اختبار سرعة الإنترنت. تشير سرعة التنزيل إلى السرعة التي يقوم بها اتصال الإنترنت بتنزيل البيانات من الإنترنت وتشير سرعة التحميل إلى السرعة التي يقوم بها اتصال الإنترنت الخاص بك بتحميل البيانات إلى الإنترنت. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية حساب سرعة اتصالك بالإنترنت باستخدام بايثون.

44) تحويل النص إلى خط اليد باستخدام بايثون Converting Text to Handwriting using Python

هناك العديد من الميزات الرائعةفي الهاتف الذكي للكتابة اليوم، مثل تحويل خط اليد للمستخدم إلى نص، ولكن ماذا لو كنت تريد تحويل نص إلى نص مكتوب بخط اليد؟ إذا كنت تريد معرفة كيفية تحويل النص إلى نص مكتوب بخط اليد، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على مهمة النص إلى خط اليد باستخدام بايثون.

تحويل النص إلى خط اليد باستخدام بايثون

نظرًا لأن بايثون هي لغة برمجة شائعة ومفتوحة المصدر، فهي توفر مكتبات لأي مهمة تقريبًا يمكنك التفكير فيها. لتحويل النص إلى خط اليد، توجد مكتبة تُعرف باسم PyWhatKit في بايثون. يوفر الكثير من الميزات المفيدة، ولكن الميزة التي أهتم بها أكثر هي تحويل نص إدخال المستخدم إلى نص مكتوب بخط اليد. إذا لم تستخدمه من قبل، فيمكنك تثبيته بسهولة على نظامك باستخدام الأمر pip:

```
pip install pywhatkit
```

الآن إليك كيفية تحويل النص إلى كتابة بخط اليد باستخدام بايثون:

```
import pywhatkit as kit
import cv2

kit.text_to_handwriting("Hope you are doing well",
save_to="handwriting.png")
img = cv2.imread("handwriting.png")
cv2.imshow("Text to Handwriting", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

في الكود أعلاه، قمت أولاً باستيراد مكتبات pywhatkit و OpenCVفي بايثون. هنا يتم استخدام pywhatkit لتحويل النص إلى نص مكتوب بخط اليد ويتم استخدام OpenCV لتصوير الصورة التي نكتب فيها نصًا مكتوبًا بخط اليد.

الملخص

هذه هي الطريقة التي يمكنك بها تحويل نص إدخال المستخدم إلى نص مكتوب بخط اليد باستخدام لغة برمجة بايثون. آمل أن تكون قد أحببت هذه المقالة حول مهمة تحويل النص إلى نص مكتوب بخط اليد باستخدام بايثون. يمكنك العثور على بعض برامج ومشاريع بايثون المذهلة التي تم حلها وشرحها بكود المصدر من هنا.

45) اغلاق الكمبيوتر باستخدام بايثون Shutdown Computer using Python

تُستخدم وحدة OSفي بايثونفي مهام مختلفة تعتمد على نظام التشغيل. يجب أن تكون قد استخدمتها من قبل عند كتابة أو قراءة الملفات من جهاز الكمبيوتر الخاص بك إلى برنامج بايثون الخاص بك. يمكن استخدامه أيضًا لإغلاق جهاز الكمبيوتر الخاص بكفي بضعة أسطر من التعليمات البرمجية. إذا كنت تريد معرفة المزيد حول إيقاف تشغيل نظامك باستخدام وحدة OS، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم برنامجًا تعليميًا حول كيفية إيقاف تشغيل جهاز الكمبيوتر الخاص بك باستخدام بايثون.

اغلاق الكمبيوتر باستخدام بايثون

لإغلاق نظامك باستخدام لغة برمجة بايثون، يجب أن يكون لديك بعض المعرفة بوحدة OSفي بايثون. يأتي مثبتًا مسبقًافي مكتبة بايثون القياسية، لذلك لا تحتاج إلى كتابة أمر pip لتثبيتهفي بيئة بايثون الخاصة بك. من قراءة أو كتابة ملف لإغلاق نظامك باستخدام بايثون، يمكن استخدام وحدة OSفي أي مهمة تعتمد على نظام تشغيل نظامك.

لإغلاق جهاز الكمبيوتر الخاص بك باستخدام بايثون، تأكد من حفظ وإغلاق جميع الملفات قيد التشغيل. الآن إليك كيفية إغلاق جهاز الكمبيوتر الخاص بك باستخدام بايثون:

```
import os
def shutdown_PC():
    os.system("shutdown /s /t 1")
shutdown_PC()
```

كما هو مذكور أعلاه، تأكد من حفظ جميع الملفات وإغلاقها باستثناء محرر الكود الخاص بك حيث كتبت برنامج بايثون لإغلاق نظامك. بمجرد بدء تشغيل البرنامج، سيتم إيقاف تشغيل نظامكفي الثواني القليلة القادمة.

الملخص

هذه هي الطريقة التي يمكنك بها إيقاف تشغيل نظامك بسهولة باستخدام لغة برمجة بايثون. تعد وحدة OSفي بايثون أداة مفيدة للغاية حيث يمكن استخدامهافي العديد من المهام التي تعتمد بشكل كامل على نظام تشغيل نظامك. آمل أن تكون قد أحببت هذه المقالة حول كيفية إيقاف تشغيل جهاز الكمبيوتر الخاص بك باستخدام بايثون.

46) تشويه عنوان IP باستخدام بايثون Defang IP Address using Python

يتم تشويه عنوان IP الخاص بالمستخدم لمنع المستخدم من النقر فوق ارتباط ضار. تعد مشكلة تشويه عناوين IP لـ أحد أسئلة مقابلة البرمجة الشائعة لشخص يخطط لعلم البيانات.في هذه المقالة، سأخبرك بكيفية تشويه عنوان IP باستخدام بايثون.

حل مشكلة تغيير عنوان IP مفيد لشخص مبتدئ لممارسة مفهوم التلاعب بالسلسلة النصية. من السهل جدًا فهمه لأنه يعتمد فقط على مفاهيم الاستبدال والانضمام. هناك العديد من الطرق الفريدة لحل هذه المشكلة، ولهذا السبب يعد هذا أحد الأسئلة المفضلة لمقابلة البرمجة.

بيان مشكلة تشويه عنوان IP

لتحويل عنوان IP إلى عنوان IP مشوه، نحتاج إلى استبدال "." مع "[.]". أثناء مقابلات البرمجة، تتمثل إحدى المشكلات القياسية لتغيير عنوان IPفي أنك تتلقى عنوان IP صالحًا، ويجب عليك إرجاع نسخة مشوهة (defanged version) من عنوان IP هذا.

هذا بشكل عام سؤال تحضيري لمقابلات البرمجة. سيعطي حل هذا السؤال بسرعة انطباعًا جيدًا بأنك تعرف كيفية فهم بيان المشكلة بسرعة لأنه لا يوجد الكثير مما تحتاجه لحل هذه المشكلة. تحتاج فقط إلى استبدال كل "." مع "[.]".

تشويه عنوان IP باستخدام بايثون

الآن دعونا نرى كيفية كتابة برنامج لتعطيل عنوان IP باستخدام بايثون. هنا تحتاج ببساطة إلى معالجة "." كفاصل وتقسيم السلسلة. ثم يتعين عليك إعادة ضم سلسلة فارغة وتحديد "[.]" كفاصل جديد:

```
def ip_address(address):
    new_address = ""
    split_address = address.split(".")
    separator = "[.]"
    new_address = separator.join(split_address)
    return new_address
ipaddress = ip_address("1.1.2.3")
print(ipaddress)
```

هذا هو مدى سهولة تشويه عنوان IP باستخدام بايثون. إذا كنت تخطط لاستخدام C++في مقابلات البرمجة الخاصة بك، فإليك كيفية تعريف عنوان IP باستخدام لغة البرمجة C++:

```
#include<iostream>
```

```
#include <bits/stdc++.h>
using namespace std;

string DefangIP(string str)
{
       string defangIP = "";

       for (char c : str)
              (c == '.') ? defangIP += "[.]" :
                                   defangIP += c;
       return defangIP;
}

int main()
{
       string str;
    cin>>str;
       cout <<DefangIP(str);
       return 0;
}
```

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إلغاء تشكيل عنوان IP باستخدام لغات البرمجة Python و C ++.

47) تجريف الويب لإنشاء مجموعة بيانات باستخدام بايثون Web Scraping to Create a Dataset using Python

يتم إنشاء مجموعات البيانات (datasets) التي تجدها على الإنترنت من مصادر بيانات مختلفة إما من قبل الشركات والمؤسسات أو يتم جمعها من مواقع الويب. يجب أن تكون قد قمت بتجريف البيانات من صفحات الويب باستخدام مكتبات بايثون، ولكن ربما تكون قد توقفت أثناء تحضير البيانات التي تم نسخها لإنشاء مجموعة بيانات. لذلكفي هذه المقالة، سأوجهك خلال برنامج تعليمي حول تجريف الويب لإنشاء مجموعة بيانات باستخدام بايثون.

كيف يتم إنشاء مجموعات البيانات عن طريق تجريف الويب؟

هناك العديد من المكتبات والأطر والأدوات المستخدمةفي مهمة تجريف الويب. بعض المكتبات والوحدات النمطية الأكثر شيوعًافي بايثون المستخدمةفي تجريف الويب هي:

1. Scrapy
2. Selenium
3. BeautifulSoup
4. Urlib.request

تعد جميع مكتبات ووحدات بايثون المذكورة أعلاه رائعة لتجريف البيانات من مواقع الويب. بعد تجريف البيانات، يتم تحضير البيانات بحيث يمكن تخزينهافي ملف CSV لإنشاء مجموعة بيانات.

دعنا الآن نرى كيفية إنشاء مجموعة بيانات عن طريق تجريف الويب باستخدام بايثون. لهذه المهمة، سأستخدم مكتبة BeautifulSoup في بايثون. سأقوم هنا بالبحث عن مصطلح عشوائي على Google وبعد ذلك سأجمع البيانات من الصفحة الأولى التي يعرضها لي Google.

لذلك، بحثت عن "comparison of programming languages" على Google وحصلت على هذه المقالة كنتيجة أولى. دعونا نرى كيف يمكننا تجريف البيانات من صفحة الويب هذه لإنشاء مجموعة بيانات. فيما يلي كيفية استخدام مكتبة BeautifulSoupفي بايثون لمهمة تجريف الويب لإنشاء مجموعة بيانات:

```
import csv
from urllib.request import urlopen
from bs4 import BeautifulSoup
```

```
html =
urlopen("https://en.wikipedia.org/wiki/Comparison_of_programmi
ng_languages")
soup = BeautifulSoup(html, "html.parser")
table = soup.findAll("table", {"class":"wikitable"})[0]
rows = table.findAll("tr")

with open("language.csv", "wt+", newline="") as f:
    writer = csv.writer(f)
    for i in rows:
        row = []
        for cell in i.findAll(["td", "th"]):
            row.append(cell.get_text())
        writer.writerow(row)


import pandas as pd
a = pd.read_csv("language.csv")
a.head()
```

	Language\n	Intended use\n	Imperative\n	Object-oriented\n	Functional\n	Procedural\n	Generic\n	Reflective\n	Event-driven\n	Other paradigm(s)\n
0	1C:Enterprise scripting language\n	Application, RAD, business, general, web, mob...	Yes\n	\n	Yes\n	Yes\n	Yes\n	Yes\n	Yes\n	Object-based,\nPrototype-based programming\n\n
1	ActionScript 3.0\n	Application, client-side, web\n	Yes\n	Yes\n	Yes\n	\n	\n	\n	Yes\n	\n
2	Ada\n	Application, embedded, realtime, system\n	Yes\n	Yes[2]\n	\n	Yes[3]\n	Yes[4]\n	\n	\n	concurrent,[5] distributed,[6]\n
3	Aldor\n	Highly domain-specific, symbolic computing\n	Yes\n	Yes\n	Yes\n	\n	\n	\n	\n	\n
4	ALGOL 58\n	Application\n	Yes\n	\n	\n	\n	\n	\n	\n	\n

الملخص

يمكن تنزيل مجموعة البيانات التي أنشأناها عن طريق تجريف الويب من هنا. يبدو أنه نفس مجموعات البيانات التي نراهافي مصادر البيانات المختلفة على الإنترنت. آمل أن تكون قد أحببت هذه المقالةفي برنامج تعليمي حول تجريف الويب لإنشاء مجموعة بيانات باستخدام بايثون.

48) ماسح السيرة الذاتية باستخدام بايثون Resume Scanner using Python

ماسح السيرة الذاتية (Resume Scanner) هو تطبيق يقوم بمسح جميع الكلمات الرئيسيةفي السيرة الذاتية لمطابقة المهارات والمؤهلات المطلوبة لوظيفة معينة. إذا كنت تريد معرفة كيفية مسح سيرة ذاتية ضوئيًا باستخدام لغة برمجة بايثون، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية إنشاء ماسح ضوئي للسيرة الذاتية باستخدام بايثون.

ما هو ماسح السيرة الذاتية؟

تستخدم معظم أقسام الموارد البشريةفي الشركات اليوم برنامجًا يُعرف باسم نظام تتبع المتقدمين الذي يستخدم لاختيار المرشحين المناسبين لوظيفة معينة. يقوم هذا البرنامج بمسح مستند لاستخراج الكلمات الرئيسية ومطابقتهافي السيرة الذاتية بالمهارات والمؤهلات اللازمة لأداء وظيفة معينة. هذا ليس أكثر من استئناف المسح والتطبيق المستخدم لأداء هذه المهمة يُعرف باسم ماسح السيرة الذاتية.

تستخدم جميع الشركات الكبيرة تقريبًا برنامج استئناف المسح الضوئي للحصول على أفضل المرشحين للوظيفة المنشورةفي غضون وقت.في القسم أدناه، سوف أطلعك على كيفية إنشاء ماسح ضوئي للسيرة الذاتية باستخدام لغة برمجة بايثون.

ماسح السيرة الذاتية باستخدام بايثون

لإنشاء ماسح للسيرة الذاتية باستخدام بايثون، عليك أولاً تثبيت وحدة بايثون المعروفة باسم resume-parser. إذا لم تكن قد استخدمت هذه الوحدة من قبل، فيمكنك تثبيتها بسهولة باستخدام الأمر pip:

```
pip install resume-parser
```

أتمنى أن تكون قد قمت الآن بتثبيت هذه الوحدةفي نظامك، فلنرى الآن كيفية مسح السيرة الذاتية باستخدام بايثون:

```
def scan_resume(resume):
    from resume_parser import resumeparse
    data = resumeparse.read_file(resume)
    for i, j in data.items():
        print(f"{i}:>>{j}")

scan_resume("Aman Kharwal.docx")
```

```
Output:
email:>>support@thecleverprogrammer.com
phone:>>8587******
total_exp:>>2
university:>>['aligarh muslim university']
designition:>>['data scientist', 'scientist', 'quality assurance']
degree:>>['BCom']
skills:>>['prediction', 'organization', 'public', 'word', 'reviews', 'product reviews', 'spotify',
'interpersonal skills', 'data science', 'facebook', 'email', 'sentiment analysis', 'tutorials',
'pressure', 'shopping', 'it', 'budget management', 'online', 'research', 'finance', 'linkedin',
'mask', 'amazon']
```

الملخص

هذه هي الطريقة التي يمكننا بها مسح السيرة الذاتية بسهولة باستخدام لغة برمجة بايثون. تستخدم جميع الشركات الكبيرة تقريبًا برنامج ماسح السيرة الذاتية للحصول على أفضل المرشحين للوظيفة المنشورةفي غضون وقت. آمل أن تكون قد أحببت هذه المقالة حول كيفية مسح السيرة الذاتية باستخدام بايثون.

49) خوارزمية فرز الدمج باستخدام بايثون Merge Sort Algorithm using Python

لفرز مصفوفة نحتاج إلى ترتيب عناصر المصفوفة بمقارنة كل عنصر بكفاءة. تستخدم خوارزمية دمج الترتيب (merge sort) طريقة فرق تسد (divide and conque) لفرز مصفوفة عن طريق إجراء أقل عدد من المقارنات بين عناصر المصفوفة.في هذه المقالة، سوف آخذك خلال تنفيذ Merge Sort باستخدام بايثون.

خوارزمية فرز الدمج

تعد القدرة على فرز عناصر المصفوفة واحدة من أهم المهاراتفي علوم الكمبيوتر ولهذا السبب لدينا مجموعة من خوارزميات الفرز (sorting algorithms)في علوم الكمبيوتر. يستخدم مفهوم الفرز على نطاق واسعفي تطبيقات مثل منصات التسوق عبر الإنترنت والبنوك والأنظمة المالية وما إلى ذلك. تعد خوارزمية فرز الدمج خوارزمية فرز تُستخدم لترتيب عناصر المصفوفة بترتيب تصاعدي أو تنازلي.

هناك مجموعة من خوارزميات الفرزفي علوم الكمبيوتر مثل فرز الفقاعات (Bubble sort) وفرز التحديد (Selection sort) وفرز الإدراج (Insertion Sort) والفرز السريع (quick sort) وفرز الدمج (Merge sort). كل هذه الخوارزميات تقوم بنفس العمل ولكن بنهج مختلف. تعد خوارزمية فرز الدمج حاليًا الطريقة الأكثر فاعلية لفرز عناصر المصفوفة.

خوارزمية دمج الفرز هي خوارزمية تقسيم وقهر تأخذ مصفوفة كمدخلات ثم تقسم المصفوفة الكاملة إلى مصفوفات فرعية من عناصر مفردة. نتيجة لذلك، يتبقى لنا العديد من المصفوفات المرتبة حيث يتم فرز العنصر الفردي دائمًا. ثم نقوم بدمج كل المصفوفات بأخذ مصفوفتينفي وقت واحد حتى نحصل على مصفوفة مرتبة نهائية.

خوارزمية فرز الدمج باستخدام بايثون

أتمنى أن تكون قد فهمت الآن ما هو دمج خوارزمية الفرزفي علوم الكمبيوتر. لتنفيذ خوارزمية فرز الدمج باستخدام بايثون، نحتاج أولاً إلى تقسيم مصفوفة إلى مصفوفات متعددة من عناصر مفردة ومن ثم يمكننا دمجها بسهولةفي مصفوفة مرتبة نهائيًا. دعنا الآن نرى كيفية تنفيذ فرز الدمج باستخدام بايثون:

```
def merge(listA, listB):
    newlist = list()
    a = 0
    b = 0
    while a < len(listA) and b < len(listB):
```

```
        if listA[a] < listB[b]:
            newlist.append(listA[a])
            a += 1
        else:
            newlist.append(listB[b])
            b += 1
    while a < len(listA):
        newlist.append(listA[a])
        a += 1
    while b < len(listB):
        newlist.append(listB[b])
        b += 1
    return newlist

def merge_sort(input_list):
    if len(input_list) <= 1:
        return input_list
    else:
        mid = len(input_list) // 2
        left = merge_sort(input_list[:mid])
        right = merge_sort(input_list[mid:])
        newlist = merge(left, right)
        return newlist

a = [56, 89, 45, 34, 90, 32, 20, 67, 43]
print(merge_sort(a))
```

```
Output:
[20, 32, 34, 43, 45, 56, 67, 89, 90]
```

الملخص

هذه هي الطريقة التي يمكنك بها تنفيذ خوارزمية فرز الدمج باستخدام بايثون. يعد حاليًا الأسلوب الأكثر فاعلية لفرز المصفوفات بين جميع خوارزميات الفرزفي علوم الكمبيوتر. أتمنى أن تكون قد أحببت هذه المقالة حول تنفيذ فرز الدمج باستخدام بايثون.

50) اختيار بطاقة عشوائية باستخدام بايثون Picking a Random Card using Python

هناك العديد من ألعاب الورق اليوم حيث تختار بطاقة عشوائيًا من مجموعة بطاقات لإنشاء حدث. هذه هي ميزة كل لعبة بطاقة اليوم لأنه يتعين عليك اختيار بطاقة بشكل عشوائي وبمجرد اختيار بطاقة تصبح حدثًا. لذلكفي هذه المقالة، سوف أطلعك على برنامج تعليمي حول كيفية اختيار بطاقة عشوائية باستخدام بايثون.

اختيار بطاقة عشوائية باستخدام بايثون

لاختيار بطاقة عشوائية باستخدام بايثون، عليك أولاً تخزين جميع البطاقاتفي هيكل بيانات (data structure). لذا قبل تخزين البطاقةفي هيكل بيانات، دعونا نفهم أنواع البطاقات الموجودةفي مجموعة البطاقات. يوجد أدناه جدول يوضح نوع البطاقات الموجودةفي مجموعة أوراق اللعب:

Spade	*Club*	*Diamond*	*Heart*
1 King	1 King	1 King	1 King
1 Queen	1 Queen	1 Queen	1 Queen
1 Jack	1 Jack	1 Jack	1 Jack
1 Ace	1 Ace	1 Ace	1 Ace
2-10 Cards	2-10 Cards	2-10 Cards	2-10 Cards
Total = 13	***Total = 13***	***Total = 13***	***Total = 13***

وفقًا للجدول أعلاه، تحتوي مجموعة البطاقات على أربع مجموعات من البطاقات بمافي ذلك القلوب (hearts) والنوادي (clubs) والبستوني (clubs) والماس (diamonds). تُعرف هذه المجموعات من البطاقات بالأجنحة (suites)، كل جناح يحتوي على ثلاثة عشر بطاقة تبدأ من 2 إلى 10 ثم تستمر مع جاك (Jack) وكوين (Queen) وكينغ (king) وآيس (king)في كل جناح.

لذا لاختيار بطاقة عشوائية من مجموعة بطاقات، سوف أقوم بإنشاء قائمتين من بايثون:

- واحد لتخزين الاجنحة.

- آخر لتخزين مراتب البطاقات.

فيما يلي كيفية كتابة برنامج بايثون لاختيار بطاقة عشوائية:

```
import random
cards = ["Diamonds", "Spades", "Hearts", "Clubs"]
ranks = [2, 3, 4, 5, 6, 7, 8, 9, 10, "Jack", "Queen", "King",
"Ace"]

def pick_a_card():
    card = random.choices(cards)
    rank = random.choices(ranks)
    return(f"The {rank} of {card}")

print(pick_a_card())
```

```
Output:
The ['Jack'] of ['Diamonds']
```

الملخص

هذه هي الطريقة التي يمكننا بها كتابة برنامج بايثون لسحب بطاقة عشوائية من مجموعة أوراق اللعب. يمكنك أيضًا تنفيذ نفس الإستراتيجية لإنشاء لعبة ورق كاملة.

51) الانحراف الربيعي باستخدام بايثون Quartile Deviation using Python

الانحراف الربيعي (Quartile deviation) يعني المقياس المطلق للتشتت (dispersion). إنه حاصل ضرب نصف الفرق بين الربيعين العلوي والسفلي (الانحراف الرباعي = (Q3 - Q1) / 2).في هذه المقالة، سوف أطلعك على كيفية حساب الانحراف الربعي باستخدام بايثون.

ما هو الانحراف الربيعي؟

الانحراف الربيعي هو المقياس المطلق للتشتت، حيث يكون التشتت هو المدى الذي تختلف فيه قيم التوزيع عن القيمة المتوسطة للتوزيع.في حالة وجود قيمة واحدة عالية أو منخفضة جداًفي البيانات، فلا يزال بإمكانها تقليل فائدة النطاق كمقياس للتشتت.

لذا، لحساب الانحراف الربيعي، نحتاج إلى تقسيم البيانات إلى أربعة أجزاء حيث يحتوي كل جزء على 25٪ من القيم. هنا، حاصل ضرب نصف الفرق بين القمة (75٪) والقاع (25٪) سيعطينا الانحراف الربعي للبيانات.

الانحراف الربيعي باستخدام بايثون

أتمنى أن تكون قد فهمت الآن ما هو الانحراف الربعي، يمكنك معرفة المزيد عنه من هنا. دعنا الآن نرى كيفية حساب الانحراف الربيعي لمجموعة البيانات باستخدام بايثون. لحسابها باستخدام لغة برمجة بايثون، سأقوم أولاً بإنشاء مجموعة بيانات ثم سأجد الرُبيع 1 والرُبيع 2 والرُبيع 3 من البيانات ثم سأقوم بإنشاء دالة تعطينا حاصل ضرب نصف الفرق بين الرُبيع 3 والرُبيع 1 . فيما يلي كيف يمكننا حسابها باستخدام بايثون:

```
import numpy as np
data = list(range(20, 100, 5))
print(data)

Q1 = np.quantile(data, 0.25)
Q2 = np.quantile(data, 0.50)
Q3 = np.quantile(data, 0.75)

print("Quartile 1 : ", Q1)
print("Quartile 2 : ", Q2)
print("Quartile 3 : ", Q3)

def QuartileDeviation(a, b):
    return (a - b)/2
print(QuartileDeviation(Q3, Q1))
```

```
Quartile 1 :  38.75
Quartile 2 :  57.5
Quartile 3 :  76.25
18.75
```

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية حساب الانحراف الربيعي لمجموعة البيانات باستخدام لغة برمجة بايثون.

52) عد تكرارات الأحرف باستخدام بايثون Counting Character Occurrences using Python

عد تكرارات (Counting occurrences) حرففي سلسلة يعني حساب جميع السلاسل الفرعية للحرف من سلسلة الإدخال. هذا أحد الأسئلة المهمة التي تطرح عليكفي مقابلات البرمجة.في هذه المقالة، سوف أطلعك على كيفية حساب تكرارات الاحرف باستخدام بايثون.

عد تكرارات الأحرف باستخدام بايثون

لحساب تكرارات الحرف، نحتاج إلى كتابة خوارزمية تُرجع عدد المرات التي يظهر فيها كل حرف في سلسلة الإدخال. يجب أن تقوم الخوارزمية بالتكرار خلال كل حرف من البداية لحساب عدد المرات التي يظهر فيها كل حرففي السلسلة النصية. إليك كيفية كتابة خوارزمية لحساب تكرارات الأحرف باستخدام بايثون:

```
def count_characters(s):
    count = {}
    for i in s:
        if i in count:
            count[i] += 1
        else:
            count[i] = 1
    print(count)
print(count_characters("Thecleverprogrammer"))
```

```
Output:
{'T': 1, 'h': 1, 'e': 4, 'c': 1, 'l': 1, 'v': 1, 'r': 4, 'p': 1, 'o': 1, 'g': 1, 'a': 1, 'm': 2}
```

في الكود أعلاه، نتخطى كل حرففي سلسلة الإدخال. إذا كان الحرف موجودًا بالفعلفي السلسلة، فإننا ببساطة نزيد قيمة الحرف بمقدار 1 وما إلى ذلك. بخلاف ذلك، نقوم فقط بإضافة الحرف في القاموس وضبطه على 1.في النهاية، نحصل على قاموس يحتوي على الأحرف كمفاتيح وقيمها بعدد مرات ظهورهافي السلسلة.

الملخص

يعد حساب تكرارات الاحرف سؤالًا مهمًا لمقابلة البرمجة. هنا يتعين علينا كتابة خوارزمية لحساب عدد المرات التي يظهر فيها كل حرففي سلسلة نصية. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية حساب تكرارات الاحرف باستخدام بايثون.

53) انشاء النمط الهرمي باستخدام بايثون Creating Pyramid Pattern using Python

إنشاء الأنماط (patterns) باستخدام لغة البرمجة لا يقل عن تصميم خوارزمية للمبتدئين. من أسهل الأنماط التي يمكنك إنشاؤها هو الهرم الذي يشبه المثلث. حتى تحصل على برمجة الأنماط، فلنبدأ بإنشاء نمط هرمي باستخدام بايثون.

النمط الهرمي باستخدام بايثون

ستساعدك برامج أنماط الكتابة على تحسين مهاراتكفي البرمجة والمشكلات المستندة إلى برامج الأنماط التي تُطرح أيضًا بشكل شائعفي مقابلات البرمجة. لذا فإن الوقت الذي تقضيهفي كتابة برامج الأنماط سيكون دائمًا مفيدًا. يشبه نمط الهرم مثلثًا غير فارغ، لذلك لإنشاء مثل هذا النمط باستخدام بايثون، تحتاج إلى استخدام حلقات for لتصميم هيكل هرمي باستخدام النجوم أو الأرقام أو الحروف الهجائية أو أي رمز آخر. فيما يلي كيفية كتابة برنامج بايثون لطباعة هرم باستخدام بايثون:

```
def pyramid_pattern(n):
    a = 2 * n-2
    for i in range(0, n):
        for j in range(0, a):
            print(end=" ")
        a = a-1
        for j in range(0, i+1):
            print("*", end=" ")
        print("\r")
print(pyramid_pattern(10))
```

```
          *
         * *
        * * *
       * * * *
      * * * * *
     * * * * * *
    * * * * * * *
   * * * * * * * *
  * * * * * * * * *
 * * * * * * * * * *
```

في الكود أعلاه، قمت بتعريف دالة بايثون التي تقبل وسيطة بطول هرمك. لذا أثناء تشغيل هذه الدالة، عليك إعطاء معلمة واحدة فقط.

الملخص

لقد أنشأت هذا الهيكل للهرم باستخدام النجوم، يمكنك أيضًا استخدام الحروف الهجائية أو الأرقام أو أي رمز عن طريق استبدالها بـ "*". هذه هي الطريقة التي يمكنك بها إنشاء نمط هرمي بسهولة باستخدام لغة برمجة بايثون.

54) البحث التسلسلي باستخدام بايثون Sequential Search using Python

خوارزمية البحث التسلسلي (Sequential search) هي خوارزمية بحث. لتنفيذ هذه الخوارزمية، نبدأ بالبحث عن القيمة الهدف من بداية المصفوفة ونستمر حتى نجد القيمة المستهدفة.في هذه المقالة، سوف آخذك خلال تنفيذ البحث التسلسلي باستخدام بايثون.

خوارزمية البحث التسلسلي

خوارزميات البحث هي الخوارزميات المستخدمة للبحث عن قيمة معينةفي هيكل بيانات (data struture) مثل القوائم (lists)في بايثون. البحث المتسلسل عبارة عن خوارزمية بحث تتحقق من كل عنصرفي هيكل بيانات من البداية للعثور على القيمة المستهدفة.

على سبيل المثال، تخيل أنك تحاول العثور على بطاقة معينة من مجموعة أوراق اللعب. سوف تمر عبر كل بطاقةفي المجموعة واحدة تلو الأخرى حتى تجد البطاقة التي تبحث عنها. بمجرد حصولك على البطاقة التي كنت تبحث عنها ستتوقف. هذه هي الطريقة التي تعمل بها خوارزمية البحث التسلسلي. الآن،في القسم أدناه، سوف آخذك خلال تنفيذ البحث التسلسلي باستخدام لغة برمجة بايثون.

البحث التسلسلي باستخدام بايثون

أتمنى أن تعرف الآن ما هي خوارزمية البحث التسلسلي وكيف تعمل. لتنفيذ هذه الخوارزمية، نحتاج إلى التحقق من كل عنصر من البداية حتى نجد القيمة التي نبحث عنها. دعنا الآن نرى كيفية تنفيذ البحث التسلسلي باستخدام بايثون:

```
def sequential_search(list_, n):
    found = False
    for i in list_:
        if i == n:
            found = True
            break
    return found
```

في كود بايثون أعلاه، قمت بتعريف دالة بايثون مع معلمتين (list_ , n) تشير "list_" إلى قائمة Python و "n" تشير إلى العنصر الذي نريد البحثفي قائمة. ثم أنا ببساطة أستخدم حلقة for للبحث عن nفي القائمة. الآن دعنا نرى كيفية استخدام الدالة أعلاه لتنفيذ خوارزمية البحث التسلسلي:

```
numbers = list(range(0, 20))
print(sequential_search(numbers, 3))
```

الملخص

البحث التسلسلي هو أحد الخوارزميات الشائعةفي كل مقابلة برمجة. لتنفيذه، عليك البحث عن القيمة المستهدفة من بداية هيكل البيانات حتى تجد القيمة المستهدفة. أتمنى أن تكون قد أحببت هذه المقالة حول تنفيذ البحث التسلسلي باستخدام بايثون.

55) تبديل المتغيرات باستخدام بايثون Swapping Variables using Python

تبديل المتغيرات (Swapping variables) تعني تخصيص قيمة المتغير a للمتغير b والعكس صحيح. لدينا العديد من الخوارزمياتفي علوم الكمبيوتر لمبادلة قيم متغيرين مع بعضهما البعض، لذلكفي هذه المقالة، سأوجهك عبر كيفية تبديل المتغيرات باستخدام بايثون.

تبديل المتغيرات باستخدام بايثون

لدينا العديد من الخوارزمياتفي علوم الكمبيوتر لتبديل المتغيرات ولكن أسهلها:

1. تبديل متغيرين بإدخال متغير آخر.
2. تبديل المتغيرات ببساطة عن طريق تخصيصها لبعضها البعض.

دعونا نرى كيفية تبديل متغيرين باستخدام بايثون من خلال إدخال متغير جديد:

```
a = 8
b = 10

c = a
a = b
b = c
print("a =", a)
print("b = ", b)
```

```
a = 10
b = 8
```

في الكود أعلاه، أعلنت أولاً عن متغيرين هما a = 8، و b = 10. ثم أقوم بإدخال متغير آخر كـ c = a مما يعني أنني أقوم بتعيين قيمة a إلى c ثمفي السطر التالي الذي أعينه قيمة b إلى a. حتى الآن لدينا c = a و a = b و b = 10. لذافي السطر التالي، سأخصص قيمة c لـ b.في النهاية، يتبقى لنا a = 10، b = 8.

الآن دعونا نرى كيفية استخدام الطريقة الأخرى لمبادلة المتغيرات وهي مبادلة قيم المتغيرات مع بعضها البعض دون إدخال متغير آخر. لا يمكننا تعيين قيم المتغيرات لبعضنا البعض سطرًا بسطر كما فعلنافي الطريقة أعلاه، لأننا إذاكتبنا a = b في السطر الأول و b = a في السطر الثاني ، فسننتهي بالحصول على a = 10 ، و b = 10 ، كمافي السطر الأول نحدد b لـ a مما يعني أننا نخصص 10 لـ a ، ثمفي السطر التالي ، نخصص a لـ b مثل 10 لـ b مرة أخرى. لذلك لمبادلة متغيرين دون إدخال متغير جديد، يتعين علينا تبديلهمافي سطر واحد فقط كما هو موضحفي الكود أدناه:

```
a = 8
b = 10
a, b = b, a
print("a = ", a)
print("b = ", b)
```

```
a =  10
b =  8
```

الملخص

هذه هي الطريقة التي يمكننا بها تبديل قيم متغير واحد بمتغير آخر. يمكنك استخدام نفس الطريقة لتبديل القوائم (lists) والقواميس (dictionaries) والمجموعات (sets) والصفوف (tuples) أيضًا بدلاً من القيم الرقمية فقط. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية تبديل المتغيرات باستخدام بايثون.

56) ترتيب مصفوفات NumPy باستخدام بايثون
Sorting NumPy Arrays using Python

كونك طالبًافي علوم الكمبيوتر، يجب أن تكون قد مررت بمفهوم الفرز الخوارزميات (algorithms). في بايثون، نستخدم خوارزميات الفرز (sorting algorithms) لفرز العناصر في قائمة. ولكن ماذا لو أردنا فرز مصفوفاتNumPy ؟في هذه المقالة، سوف أطلعك على كيفية فرز مصفوفات NumPy باستخدام بايثون.

فرز مصفوفاتNumPy

هناك مجموعة من خوارزميات الفرزفي علوم الكمبيوتر، على سبيل المثال:

1. فرز بالإدراج Insertion Sort.
2. فرز الاختيار Selection Sort.
3. فرز الدمج Merge Sort.
4. فرز الفقاعات Bubble Sort وغيرها الكثير.

تُستخدم كل هذه الخوارزميات لفرز القيمفي قائمة أو مصفوفة. توفر مكتبة NumPy دالة داخلية لفرز القيم داخل مصفوفة NumPy. فيما يلي كيفية فرز مصفوفة NumPy باستخدام دالة sort المضمنة:

```
import numpy as np
a = np.array([76 ,23 ,89 ,5 ,34])
print(np.sort(a))
```

تعمل دالة sortفي مكتبة NumPy بنفس دالة sortفي لغة برمجة بايثون. ولكن، ماذا لو أردنا كتابة خوارزمية فرز لفرز مصفوفة NumPy باستخدام بايثون دون استخدام دالة sort؟في القسم أدناه، سأطلعك على كيفية فرز مصفوفات NumPy باستخدام بايثون.

فرز مصفوفات NumPy باستخدام بايثون

يعد فرز القيمفي هيكل البيانات أحد الموضوعات المفضلة لمقابلات البرمجة. مصفوفة NumPy هي أيضًا هيكل بيانات مثل قائمة أو مصفوفة، لذلك يجب أن تعرف أيضًا كيفية فرز قيم مصفوفة NumPy باستخدام بايثون دون استخدام أي دالة فرز. فيما يلي كيفية فرز مصفوفة NumPy باستخدام بايثون:

```
def sorting(x):
    for i in range(len(x)):
        swap = i + np.argmin(x[i:])
        (x[i], x[swap]) = (x[swap], x[i])
    return x
print(sorting(a))
```

الملخص

تعتمد الخوارزمية أعلاه على فرز الاختيار (selection sort). إنها خوارزمية شائعة جداً لفرز القيمفي هيكل البيانات. آمل أن تكون قد أحببت هذه المقالة حول كيفية فرز قيم مصفوفة NumPyباستخدام بايثون دون استخدام أي دالة مضمنة.

57) التحقق من صحة الجناس الناقصة باستخدام بايثون
Validate Anagrams using Python

الجناس الناقص (Anagram) هو كلمة أو عبارة تشكل كلمة أو عبارة مختلفة عند إعادة ترتيب أحرف الكلمة. على سبيل المثال، الكلمات "despair" و "praised " هي الجناس الناقصة.في هذه المقالة، سوف أطلعك على كيفية التحقق من صحة الجناس الناقصة باستخدام بايثون.

التحقق من صحة الجناس الناقصة باستخدام بايثون

يعد التحقق من صحة كلمات الجناس الناقص أحد الأسئلة المفضلةفي مقابلات البرمجة. الفكرة هي كتابة خوارزمية للتحقق مما إذا كانت كلمة الإدخال تخلق كلمة ذات معنى عند إعادة ترتيبها. لذلك للتحقق من صحة الجناس الناقص باستخدام بايثون، نحتاج إلى إدخال كلمتين والتحقق مما إذا كانت الكلمة الاولى word1 تتطابقفي أي حال مع الكلمة الثانية word2بعد إعادة ترتيب الكلمات.

على سبيل المثال، الكلمات "cinema" و "Iceman"، لنفترض أن word1 هنا هي "cinema "، لذلك نحتاج إلى كتابة خوارزمية للتحقق مما إذا كان بإمكاننا إنشاء كلمة "Iceman" بعد إعادة ترتيب أحرف الكلمة " cinema ". فيما يلي كيف يمكننا التحقق من صحة الجناس الناقصة باستخدام بايثون:

```
def anagram(word1, word2):
    word1 = word1.lower()
    word2 = word2.lower()
    return sorted(word1) == sorted(word2)

print(anagram("cinema", "iceman"))
print(anagram("cool", "loco"))
print(anagram("men", "women"))
```

```
True
True
False
```

في الكود أعلاه، بدأت بكتابة دالة بايثون كـ "anagram" والتي تتضمن معلمتين (word1، word2). الآن أثناء تهيئة الكلمات، قمت بتحويلها إلى أحرف صغيرة، ثم أتحقق مما إذا كانت word1 تساوي word2 بعد فرز كلتا الكلمتين.

الملخص

هذه هي الطريقة التي يمكننا بها التحقق من صحة كلمات الجناس الناقص باستخدام لغة برمجة بايثون. إنه أحد أهم الأسئلةفي أي مقابلات البرمجة. آمل أن تكون قد أحببت هذه المقالة حول كيفية التحقق من صحة الجناس الناقصة باستخدام بايثون.

58) أنشاء جداول باستخدام بايثون Creating Tables using Python

بايثون هي لغة برمجة سهلة للغاية ومتعددة الاستخدامات. يقدم مكتبات ووحدات نمطية لكل مهمة تقريبًا يمكنك التفكير فيها. أثناء العمل مع البيانات باستخدام بايثون، يصعب أحيانًا تقديمها بتنسيق جدولي باستخدام دوال التنسيق القياسية التي توفرها بايثون. لذلكفي هذه المقالة، سوف آخذك خلال برنامج تعليمي حول وحدة (tabulate) لإنشاء جداول باستخدام بايثون.

وحدة tabulate في بايثون

تسمح لنا وحدة tabulate في بايثون بإنشاء وعرض البيانات بتنسيق جدولي مما يجعل البيانات تبدو أكثر قابلية للقراءة. يمكن استخدامه لتنظيم بياناتك لجعلها أكثر قابلية للفهم. فيما يلي بعض هياكل البياناتفي بايثون التي تدعمها وحدة (tabulate):

1. القوائم (lists)
2. قاموس (dictionaries)
3. مصفوفة NumPy.
4. إطار بيانات Pandas.

لا يتم تثبيت وحدة tabulate مسبقًافي مكتبة بايثون القياسية بحيث يمكنك تثبيتها بسهولة باستخدام الأمر pip؛

```
pip install tabulate
```

أنشاء جداول باستخدام بايثون

آمل أن تكون قد فهمت الآن بعض الميزات المهمة التي توفرها وحدة الجدولةفي بايثون. الآن دعونا نرى كيفية إنشاء جداول باستخدام بايثون باستخدام وحدة tabulate. فيما يلي كيفية إنشاء جدول بسيط للغاية باستخدام بايثون:

```
from tabulate import tabulate
data = [["Name", "Place", "Gender"], ["Aman", "New Delhi",
"Male"], ["Hritika", "New Delhi", "Female"], ["Krishna", "UP",
"Male"]]
print(tabulate(data))
```

```
-------  ---------  ------
Name     Place      Gender
Aman     New Delhi  Male
Hritika  New Delhi  Female
Krishna  UP         Male
-------  ---------  ------
```

الكود أعلاه يسردفي تنسيق جدولي، دعونا الآن نلقي نظرة على كيفية فصل الترويسات (headers) عن القيم (values):

```
print(tabulate(data, headers='firstrow'))
```

```
Name     Place      Gender
-------  ---------  --------
Aman     New Delhi  Male
Hritika  New Delhi  Female
Krishna  UP         Male
```

يمكننا أيضًا تصميم هذا الجدول عن طريق إضافة شبكة (grid)، وإليك كيفية القيام بذلك:

```
+---------+-----------+----------+
| Name    | Place     | Gender   |
+=========+===========+==========+
| Aman    | New Delhi | Male     |
+---------+-----------+----------+
| Hritika | New Delhi | Female   |
+---------+-----------+----------+
| Krishna | UP        | Male     |
+---------+-----------+----------+
```

يمكننا أيضًا جعل الشبكة تبدو أفضل:

```
╒═════════╤═══════════╤══════════╕
│ Name    │ Place     │ Gender   │
╞═════════╪═══════════╪══════════╡
│ Aman    │ New Delhi │ Male     │
├─────────┼───────────┼──────────┤
│ Hritika │ New Delhi │ Female   │
├─────────┼───────────┼──────────┤
│ Krishna │ UP        │ Male     │
╘═════════╧═══════════╧══════════╛
```

الملخص

إذن هذه هي الطريقة التي يمكنك بها تقديم بياناتكفي شكل جداول. إنها طريقة جيدة لتنسيق البياناتفي جداول لأنها تجعل البيانات تبدو أكثر قابلية للقراءة. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء جداول باستخدام بايثون.

59) البحث الثنائي المتكرر باستخدام بايثون Recursive Binary Search using Python

يعني التكرار (Recursion) حل المشكلات عن طريق تقسيم مشكلة معقدة إلى مشكلات أصغر ثم حلها خطوة بخطوة.في هذه المقالة، سوف أطلعك على تنفيذ البحث الثنائي المتكرر (recursive binary search) باستخدام بايثون، مما يعني تنفيذ خوارزمية البحث الثنائي باستخدام الطريقة التكرارية (recursive method).

البحث الثنائي المتكرر

يعني البحث الثنائي العثور على عنصرفي مصفوفة مرتبة عن طريق تقسيم فاصل البحث بشكل متكرر إلى نصفين ويعني البحث الثنائي المتكرر تقسيم عملية البحث الثنائي بأكملها إلى مشاكل أصغر. ببساطة، يُعرف الحل التكراري للبحث الثنائي بالبحث الثنائي التكراري.

فيما يلي الخصائص التي يجب أن تفي بها جميع الحلول التكرارية:

1. الحل التكراري يجب أن يكون له حالة أساسية (base case).
2. يجب أن يكون للحل التكراري حالة متكررة (recursive case).
3. يجب أن يحرز الحل التكراري تقدمًا نحو الحالة الأساسية.

الحالة الأساسية هي حالة أخيرة تمثل أصغر تقسيم فرعي لمشكلة معقدة. لذلك وفقًا لخصائص التكرارية أعلاه، لتنفيذ البحث الثنائي التكراري، يجب أن تحتوي الخوارزمية الخاصة بنا على حالة أساسية وحالة تكرارية ويجب أن تتقدم الحالة التكرارية إلى الحالة الأساسية وإلا فلن تتوقف الخوارزمية أبدًا وستؤدي إلى حلقة لا نهائية.

البحث الثنائي المتكرر باستخدام بايثون

تعمل خوارزمية البحث الثنائي على تحسين وقت البحث المطلوب لتحديد موقع عنصرفي مصفوفة مرتبة. عادة، يتم اتباع نهج تكراري لتنفيذ خوارزمية البحث الثنائي، ولكن يمكننا أيضًا تنفيذها بشكل متكرر من خلال تنفيذهافي إصدارات أصغر.

لتنفيذ خوارزمية البحث الثنائي التكراري، يجب علينا أولاً إيجاد الهدف بترتيب مصنف، ويتم فحص القيمة الوسطى لتحديد ما إذا كان هو الهدف. إذا لم تكن القيمة الوسطى هي الهدف، يتم تقسيم التسلسل إلى نصفين، ثم يتم فحص النصف الأول أو الثاني للعثور على القيمة المستهدفة من خلال النظر إلى العنصر الأوسط. إليك كيفية تنفيذ بحث ثنائي متكرر باستخدام بايثون:

```
def rec_binarySearch(target, sequence, first, last):
```

```
    if first > last:
        return False
    else:
        mid = (last + first) // 2
        if sequence[mid] == target:
            return True
        elif target < sequence[mid]:
            return rec_binarySearch(target, sequence, first,
mid-1)
        else:
            return rec_binarySearch(target, sequence, mid + 1,
last)
```

الملخص

التكرار (Recursion) هي أداة قوية للغاية للبرمجة وحل المشكلات. يمكن استخدامه لحل وتنفيذ مجموعة واسعة من الخوارزميات لحل المشاكل التكرارية الأساسية لمشاكل التراجع المتقدمة.في هذه المقالة، اكتشفنا كيفية تنفيذ خوارزمية البحث الثنائي التكراري باستخدام لغة برمجة بايثون. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية تنفيذ البحث الثنائي التكراري باستخدام بايثون.

60) الحلقات العكسية باستخدام بايثون Backward For Loop using Python

يعد استخدام حلقة for على أي كائن بايثون أمرًا سهلاً للغاية مقارنة بلغات البرمجة الأخرى مثل C ++ و Java. ولكن هل سبق لك أن حاولت استخدام حلقة for بشكل عكسي؟ إنها سهلة مثل حلقة for العامة التي تستخدمها أثناء التكرار فوق أي كائن بايثون. لذلك إذا كنت تريد معرفة كيفية استخدام حلقة عكسية backward for loop، فهذه المقالة مناسبة لك.في هذه المقالة، سوف آخذك من خلال برنامج تعليمي حول كيفية استخدام backward for loopالحلقات العكسية باستخدام بايثون.

الحلقات العكسية باستخدام بايثون

تستخدم حلقة For للتكرار على أي كائن بايثون. إذا كنت تريد إرجاع فهرس أي قيم، فعليك دائمًا تفضيل حلقة for. فيما يلي كيفية استخدام حلقة for على قائمة:

```
list_ = ["Aman", "Kharwal", "Akanksha", "Hritika", "Shiwangi"]
for i in list:_
    print(i)
```

```
Aman
Kharwal
Akanksha
Hritika
Shiwangi
```

الآن بالعودة إلى استخدام حلقة for إلى الوراء، فإن بايثون لديها دالة مضمنة تُعرف باسم reversed() والتي يمكن استخدامها لعكس ترتيب كائن بايثون أثناء استخدام حلقة for. فيما يلي كيفية استخدام backward for loop باستخدام بايثون:

```
for i in reversed(list_):
    print(i)
```

```
Shiwangi
Hritika
Akanksha
Kharwal
Aman
```

ملخص

يمكن استخدام الدالة المضمنةفي بايثون والمعروفة باسم reversed() لاستخدام حلقة for إلى الوراء. هذه هي الطريقة التي يمكنك بها بسهولة استخدام حلقة for للخلفي لغة برمجة بايثون.

أتمنى أن تكون قد أحببت هذه المقالةفي برنامج تعليمي حول كيفية استخدام backward for loop باستخدام لغة برمجة بايثون.

61) خوارزمية Dijkstra باستخدام بايثون Dijkstra's Algorithm using Python

تُعرف خوارزمية Dijkstra أيضًا باسم خوارزمية أقصر مسار أحادي المصدر (single-source shortest path). يتم استخدامه للعثور على أقصر مسار بين عقد الرسم البياني حيث تكون تكلفة كل مسار مختلفة.في هذه المقالة، سوف آخذك عبر خوارزمية Dijkstra وتنفيذها باستخدام بايثون.

خوارزمية Dijkstra

تُستخدم خوارزمية Dijkstra للعثور على أقصر مسار بين عقد الرسم البياني.في تطبيقات العالم الحقيقي، يتم استخدامه للعثور تلقائيًا على الاتجاهات بين المواقع المادية، حيث أن الاتجاهات التي تحصل عليها على خرائط Google هي مثال على خوارزمية Dijkstra.

يمكننا أيضًا استخدام خوارزمية Breadth-First Search للعثور على أقصر مسار، ولكن المشكلة هي أنها تفترض أن تكلفة اجتياز كل مسار هي نفسها. بينما تساعدنا خوارزمية Dijkstraفي العثور على أقصر مسار حيث تختلف تكلفة كل مسار.

في خوارزمية Breadth-First Search، ننتقل من عقدة واحدة إلى جميع العقد الأخرى، مما يعني أننا نتبع طريقة من يأتي أولاً يخدم أولاً. ولكنفي خوارزمية Dijkstra، بدلاً من اتباع طريقة من يأتي أولاً يخدم أولاً، نتعامل مع أقرب العقد أولاً بحيث يستغرق عددًا قليلاً جدًا من الخطوات للعثور على أقصر مسار.

خوارزمية Dijkstra باستخدام بايثون

آمل أن تكون قد فهمت الآن ما هي خوارزمية Dijkstra. لتنفيذه، يتعين علينا اختيار العقدة الأولى الأقرب إلى المصدر للعثور على أقصر مسار. دعنا الآن نرى كيفية تنفيذ خوارزمية Dijkstra باستخدام بايثون:

```
from heapq import *
from collections import defaultdict

def dijkstra(edges, strat_node, end_node):
    g = defaultdict(list)
    for start, end, weight in edges:
        g[start].append((weight, end))
    q, visited = [(0, strat_node,())], set()
    while q:
        (cost,v1,path) = heappop(q)
        if v1 not in visited:
```

```
            visited.add(v1)
            path = (v1, path)
            if v1 == end_node:
                return (cost, path)
            for c, v2 in g.get(v1, ()):
                if v2 not in visited:
                    heappush(q, (cost+c, v2, path))
        print (q)
    return float("inf")

if __name__ == "__main__":

    edges = [
        ("A", "B", 7),
        ("A", "D", 5),
        ("B", "C", 8),
        ("B", "D", 9),
        ("B", "E", 7),
        ("C", "E", 5),
        ("D", "E", 7),
        ("D", "F", 6),
        ("E", "F", 8),
        ("E", "G", 9),
        ("F", "G", 11)
    ]

    print ("=== Dijkstra ===")
    print ("A >> G:")
    print (dijkstra(edges, "A", "G"))
```

```
=== Dijkstra ===
A >> G:
[(5, 'D', ('A', ())), (7, 'B', ('A', ()))]
[(7, 'B', ('A', ())), (12, 'E', ('D', ('A', ()))), (11, 'F', ('D', ('A', ())))]
[(14, 'E', ('B', ('A', ()))), (21, 'G', ('E', ('D', ('A', ())))), (15, 'C', ('B', ('A', ()))), (22,
'G', ('F', ('D', ('A', ()))))]
[(15, 'C', ('B', ('A', ()))), (21, 'G', ('E', ('D', ('A', ())))), (22, 'G', ('F', ('D', ('A', ()))))]
[(21, 'G', ('E', ('D', ('A', ())))), (22, 'G', ('F', ('D', ('A', ()))))]
(21, ('G', ('E', ('D', ('A', ())))))
```

الملخص

في هذه المقالة، قدمت لك خوارزمية Dijkstra وتنفيذها باستخدام بايثون. إنها طريقة أفضل للعثور على أقصر طريق عندما تختلف تكلفة كل مسار. تُعد الاتجاهات التي تحصل عليهافي خرائط Google أحد الأمثلة حيث يتم استخدام خوارزمية Dijkstra. أتمنى أن تكون قد أحببت هذه المقالة حول خوارزمية Dijkstra باستخدام بايثون.

62) جداول التجزئة باستخدام بايثون Hash Tables using Python

تشبه جداول التجزئة (Hash tables) القواميس (dictionaries)في لغة بايثون، فهي هياكل بيانات تُستخدم لتخزين واسترداد كمية كبيرة من البيانات بتنسيق المفاتيح والقيم.في هذه المقالة سوف أقدم لكم مفهوم جداول التجزئة باستخدام بايثون.

جداول التجزئة

في الحوسبة، تعد جداول التجزئة واحدة من أهم هياكل البيانات التي تشبه القواميسفي لغة برمجة بايثون. يعتمد جدول التجزئة على مفهوم التجزئة (hashing) الذي يوفر طريقة لتخزين واسترداد البيانات بكفاءةفي تعقيدات الزمان والمكان.

يستخدم مفهوم جداول التجزئة على نطاق واسعفي تطبيقات مثل:

1. فهرسة قاعدة البيانات.
2. تصميم المترجم.
3. التخزين المؤقت.
4. مصادقة كلمة المرور.
5. تحليل الأخطاء وغيرها الكثير.

تستند جداول التجزئة إلى مفهوم التجزئة، مما يعني استخدام دالة التجزئة المستخدمة لتعيين المفتاح (key) والقيم (values). نظرًا لأنه يتم استخدامه لتعيين أزواج المفاتيح والقيمة، فإنه يُعرف باسم hashmap.

جداول التجزئة باستخدام بايثون

يعتمد تنفيذ جدول التجزئة على مفهوم تعيين المفتاح والقيمة تمامًا مثل القواميسفي بايثون. بمعنى آخر، تُستخدم هياكل البيانات هذه لتعيين كل مفتاح فريد لقيمته. عند تنفيذ جداول التجزئة، نحتاج إلى التأكد من أن كل مفتاح إدخال يجب أن يمر عبر دالة تجزئة ستحول نوع بياناته الأولي إلى قيمة عدد صحيح تسمى التجزئة (hash).

تأتي بايثون مع دالة hash() المدمجة التي تعمل على تسريع عملية تنفيذ جدول التجزئة بالكامل. إليك كيفية تنفيذ جداول التجزئة باستخدام بايثون:

```
class hashtable:
    def __init__(self, items):
        self.bucket_size = len(items)
```

```
        self.buckets = [[] for i in range(self.bucket_size)]
        self.assign_buckets(items)

    def assign_buckets(self, items):
        for key, value in elements:
            hash_value = hash(key)
            index = hash_value % self.bucket_size
            self.buckets[index].append((key, value))

    def get_value(self, input_keys):
        hash_value = hash(input_keys)
        index = hash_value % self.bucket_size
        bucket = self.buckets[index]
        for key, value in bucket:
            if key == input_keys:
                return(value)
```

الملخص

إذن هنا كيف يمكننا تنفيذ جداول التجزئة باستخدام بايثون. جدول التجزئة هو مفهوم مهم لهياكل البيانات والخوارزميات المستخدمة لتحويل المفاتيح من أي نوع بيانات إلى أعداد صحيحة ثم تعيين المفاتيح إلى قيمة فريدة. إنها هيكل بيانات فعالة للغاية من حيث تعقيد الوقت والمكان ويمكن استخدامهافي تطبيقات مثل فهرسة قواعد البيانات ومصادقة كلمة المرور والعديد من التطبيقات المعقدة الأخرى التي تتضمن تعيين المفتاح والقيمة. آمل أن تكون قد أحببت هذه المقالة حول مفهوم بنية بيانات جدول التجزئة وتنفيذه باستخدام بايثون.

63) الطوابير باستخدام بايثون Queues using Python

الطابور (queue) هي هيكل بيانات حيث نقوم بإدخال العناصر من الخلف وإزالة العناصر من الأمام. يتبع مبدأ First In، First Out هياكل البيانات.في هذه المقالة، سأوجهك خلال تنفيذ الطابور باستخدام بايثون.

الطوابير

الطوابير هي هياكل بياناتفي علوم الكمبيوتر تشبه القوائم (list) في بايثون حيث يمكنك إدراج العناصر وحذفها. الطوابير تشبه المكدسات (stacks) حيث يمكنك إدراج العناصر وحذفها بترتيب معين، ولكن على عكس المكدسات التي تتبع مبدأ أحدث هياكل البيانات الواردة أولاً، تتبع قوائم الانتظار مبدأ آخر ما يرد أولاً يصرف أولاً هياكل البيانات حيث العنصر الأول المضاف هو العنصر الأول الذي تمت إزالته.

يمكنك التفكيرفي هياكل بيانات الطوابير كخط من الأشخاص الذين ينتظرون شراء تذاكر لعرض ما. هنا يكون أول شخصفي الطابور هو أول من يشتري التذكرة الأولى وهكذا. لذلك يمكننا القول إن هيكل بيانات الطابورفي علوم الكمبيوتر تحاكي الطابور الحقيقي.

الطابور هي مثل المصفوفات (arrays) حيث نعمل بطول أكبر بواحد من فهرس العنصر الأخير في المصفوفة.في الطوابير، نستخدم نهجًا مشابهًا.في القسم أدناه، سأوجهك خلال تنفيذ الطوابير باستخدام لغة برمجة بايثون.

الطوابير باستخدام بايثون

فيما يلي الدوال التي يوفرها هيكل بيانات الطابور:

1. **enqueue**: تُستخدم لإدراج عنصر جديدفي الطابور.
2. **dequeue**: يتم استخدامه لإزالة عنصر من الطابور.
3. **is_empty**: تقوم بإرجاع True إذا كان الطابور فارغة وتعيد القيمة false إذا لم يكن الطابور فارغاً.
4. **size**: كما يوحي الاسم ، يقوم بإرجاع عدد العناصرفي الطابور.

آمل أن تفهم الآن مفهوم الطوابيرفي علوم الكمبيوتر. الآن، بناءً على فهم هياكل بيانات الطوابير التي حصلت عليها من المفاهيم والدوال المذكورة أعلاه، دعنا نرى كيفية تنفيذ الطوابير باستخدام بايثون:

```
class queue:
    def __init__(self):
        self.items = []
    def is_empty(self):
        return self.items == []
    def enqueue(self, item):
        self.items.insert(0, item)
    def dequeue(self):
        return self.items.pop()
    def size(self):
        return len(self.items)
```

كما ترونفي قسم الكود أعلاه، لقد أعلنت جميع دوال هيكل بيانات الطابور، والآن يمكنك بسهولة تهيئة جميع هذه الدوال واحدة تلو الأخرى لمعرفة كيفية عمل هيكل بيانات الطابور.

الملخص

تتبع هيكل بيانات الطابور مبدأ أولاًفي هياكل البيانات الصادرة أولاً. إنه يحاكي الطابور الواقعي حيث يحصل أول شخصفي الطابور على التذكرة الأولى. أتمنى أن تكون قد أحببت هذه المقالة حول مفهوم الطوابير وتنفيذها باستخدام لغة برمجة بايثون.

64) التحقق من صحة شجرة البحث الثنائية باستخدام بايثون Validate a Binary Search Tree using Python

يعد التحقق من صحة شجرة البحث الثنائية (Validating a binary search tree) سؤالًا رائعًا لإجراء المقابلات البرمجية. يعني كتابة خوارزمية للتحقق مما إذا كانت الشجرة الثنائية عبارة عن شجرة بحث ثنائية أم لا.في هذه المقالة، سوف أطلعك على كيفية كتابة خوارزمية للتحقق من صحة شجرة بحث ثنائية باستخدام بايثون.

كيفية التحقق من صحة شجرة البحث الثنائية؟

شجرة البحث الثنائية (BST) هي شجرة ثنائية يتم وضع عناصرها بترتيب خاص بحيث تكون جميع القيمفي كل شجرة بحث ثنائية أقل من تلك الموجودةفي الشجرة الفرعية الموجودة على اليمينفي كل شجرة بحث ثنائية.

الشجرة الثنائية (binary tree) هي شجرة تحتوي كل عقدة فيها على عقدتين فرعيتين. تُعرف العقدة الفرعية الأولى بالعقدة الفرعية اليسرى وتعرف العقدة الفرعية الثانية بالعقدة الفرعية اليمنى.في حين أن شجرة البحث الثنائية عبارة عن شجرة ثنائية حيث تحتوي كل عقدة على مفتاح فريد بحيث تكون كل عقدة يسرى أقل من كل عقدة يمنى.

التحقق من صحة شجرة البحث الثنائية: بيان المشكلة

في المقابلات البرمجية، يكون بيان المشكلة الذي تحصل عليه للتحقق من صحة الشجرة الثنائية كما يلي:

"لقد تم إعطاؤك جذر شجرة ثنائية، وتحتاج إلى كتابة خوارزمية لتحديد ما إذا كانت شجرة بحث ثنائية صالحة أم لا."

التحقق من صحة شجرة البحث الثنائية باستخدام بايثون

وفقًا لفهم ماهية شجرة البحث الثنائية وما يقرأه بيان المشكلة، إليك كيفية التحقق من صحة شجرة بحث ثنائية باستخدام بايثون:

```
class binarytree:
    def __init__(self, val):
        self.val = val
        self.leftnode = leftnode
        self.rightnode = rightnode
import sys
class BinarySearchTree:
    def validate_BST(self, root: binarytree) -> bool:
        return self.valid(root, sys.maxsize, -sys.maxsize)
```

```
    def valid(self, root, max_, min_):
        if root == None:
            return True
        else:
            return False
        return self.valid(root.leftnode, root.val, min_) and
self.valid(root.rightnode, max_, root.val)
```

الملخص

إذن هذه هي الطريقة التي تحتاج إليها لكتابة خوارزمية للتحقق من صحة شجرة البحث الثنائية باستخدام لغة برمجة بايثون. إنه مفهوم مهم يجب أن تعرفه عن المقابلات البرمجية. أثناء التحقق من صحة شجرة البحث الثنائية، تذكر دائمًا أن كل عقدة يسرى أصغر من كل عقدة يمنى. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية كتابة خوارزمية للتحقق من صحة شجرة بحث ثنائية باستخدام لغة برمجة بايثون.

65) المكدسات باستخدام بايثون Stacks using Python

المكدسات (Stacks) عبارة عن أنواع بيانات مجردة يتم استخدامها بشكل شائعفي جميع لغات البرمجة تقريبًا. المكدس عبارة عن هيكل بيانات تحاكي مكدسات العالم الحقيقي مثل مكدس البطاقات، ومكدس من اللوحات، وما إلى ذلك.في هذه المقالة، سأقدم لك مفهوم التكديسفي الحوسبة وتنفيذها باستخدام بايثون.

المكدسات

المكدس عبارة عن هيكل بيانات تشبه قائمةفي بايثون حيث يمكنك إضافة العناصر وإزالتها. هناك بعض المصطلحات المهمة التي ستعرفها عند تنفيذ حزم Stacks بأي لغة برمجة:

1. إزالة عنصر من المكدس يسمى (popping).
2. إدخال عنصرفي كومة يسمى (pushing).

تتبع المكدسات مبدأ هياكل البيانات Last-in-First-out، حيث يكون العنصر الأخير الذي تم إدراجه هو العنصر الأول الذي يتم إخراجه. بشكل عام لديها خمس دوال:

1. **is_empty**: يتم إرجاع True إذا كانت المكدسات فارغة وإرجاع False إذا لم يكن المكدس فارغًا.
2. **push**: يقوم بإدراج عنصرفي أعلى المكدس.
3. **pop**: يزيل العنصر العلوي من المكدس ويعيده.
4. **peep**: إرجاع العنصر العلوي من المكدس لكنه لا يزيله.
5. **size**: يقوم بإرجاع عدد صحيح يمثل عدد العناصر الموجودةفي المكدس.

تنفيذ المكدسات باستخدام بايثون

لذلك، نظرًا لأنني قدمت لك جميع دوال المكدس، سيكون من السهل عليك الآن فهم تنفيذ الحزم باستخدام لغة برمجة بايثون. إليك كيفية تنفيذ هيكل بيانات مكدس باستخدام بايثون:

```
class Stack:
    def __init__(self):
        self.items = []
    def is_empty(self):
        return self.items == []
    def push(self, item):
        self.items.append(item)
    def pop(self):
        return self.items.pop()
    def peek(self):
        l = len(self.items)-1
        return self.items[l]
```

```
    def size(self):
        return len(self.items)
```

الآن دعنا نضيف بعض العناصر إلى المكدسات ونقوم بتهيئة بعض دوال المكدس باستخدام بايثون:

```
stack = Stack()
print(stack.is_empty())

for i in range(0, 10):
    stack.push(i)
print(stack.size())
print(stack.items)
```

```
True
10
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

الملخص

المكدسات هي هيكل بيانات مهمةفي علوم الكمبيوتر تتبع مبدأ آخر ما يصرف أولاً من هياكل البيانات. بعض التطبيقات التي يتم فيها استخدام هيكل بيانات المكدس هي:

- تحليل اللغة بدون سياق.
- تقييم التعبيرات الحسابية.
- إدارة المكالمات الوظيفية.
- اجتياز الأشجار وخوارزميات الرسم البياني.

لذلك آمل أن تكون قد أحببت هذه المقالة حول مفهوم هيكل بيانات المكدس وتنفيذها باستخدام بايثون.

66) الكلمات المتناظرة باستخدام بايثون Palindrome Words using Python

الكلمات المتناظرة (Palindrome words) هي تلك الكلمات التي تُقرأ بنفس الطريقة من اليسار إلى اليمين كما من اليمين إلى اليسار. تعد كتابة خوارزمية للتحقق مما إذا كانت الكلمة متناظرة أم لا سؤالًا مهمًافي المقابلات البرمجية.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج للتحقق من الكلمات المتناظرة باستخدام بايثون.

الكلمات متناظرة: بيان المشكلة

يعد العثور على كلمة متناظرة موضوعًا مهمًا يجب أن تعده لأي مقابلة برمجة. الكلمات المتناظرة هي تلك الكلمات التي تُقرأ بنفس الطريقة عندما نقرأها من كلا الطرفين الأيمن والأيسر. على سبيل المثال، lol، mom، إلخ.في مقابلات البرمجة، يتم إعطاؤك جملة ويطلب منك كتابة خوارزمية للعثور على كلمات متناظرة من تلك الجملة.

لذلك، لمعرفة ما إذاكانت الكلمة المتناظرة موجودة أم لافي الجملة، يُطلب منا عادةً إعادة الكلمات مرة واحدة فقط إذا ظهرت هذه الكلمة أكثر من مرةفي الجملة. إذن بيان المشكلة يذهب على النحو التالي:

اكتب خوارزمية للعثور على جميع الكلمات المتناظرةفي جملة معينة وإذا ظهرت الكلمة أكثر من مرةفي الجملة، فيجب عليك دائمًا إرجاع هذه الكلمة مرة واحدة فقط. تلميح: كلمات متناظرة هي كلمات تعطي نفس النتيجة عندما تقرأ الكلمة من البداية أو النهاية.

الكلمات المتناظرة باستخدام بايثون

للعثور على كلمات متجانسة باستخدام بايثون، نحتاج أولاً إلى إزالة علامات الترقيم من الجملة، ثم نحتاج إلى الانتقال إلى بقية الشروط. الآن دعونا نرى كيفية العثور على الكلمات المتناظرة باستخدام بايثون من أي جملة:

```
def palindrome(sentence):
    for i in (",.'?/><}{{}}'"):
        sentence = sentence.replace(i, "")
    palindrome = []
    words = sentence.split(' ')
    for word in words:
        word = word.lower()
        if word == word[::-1]:
            palindrome.append(word)
    return palindrome
```

هذه هي الطريقة التي يمكنك بها كتابة خوارزمية للعثور على كلمة متناظرة باستخدام بايثون. الآن دعنا ندخل جملة ونستخدم هذه الخوارزمية:

```
sentence = input("Enter a sentence : ")
print(palindrome(sentence))
```

```
Enter a sentence : LOL, My interview went good. My Mom will be so happy.
['lol', 'mom']
```

الملخص

هذه هي الطريقة التي يمكننا بها إيجاد كلمات متناظرة من أي جملة. هذا سؤال مقابلة برمجة. أتمنى أن تكون قد أحببت هذا المقال حول كيفية العثور على الكلمات المتناظرة باستخدام بايثون.

67) خوارزمية Breadth-First Search باستخدام بايثون
Breadth-First Search Algorithm using Python

تعد خوارزمية Breadth-First Search(BFS) عبارة عن خوارزمية رسم بياني تُستخدم لاجتياز رسم بياني للعثور على عقدة معينة للتأكد من أننا قمنا بزيارة جميع العقد عن طريق عبور طبقةفي كل خطوة.في هذه المقالة، سأقدم لك خوارزمية Breadth-First Search باستخدام بايثون.

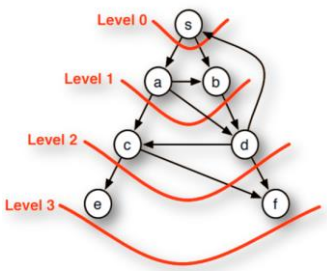
Breadth-First Search

Breadth-First Search عبارة عن خوارزمية بحث عن الرسم البياني يمكن استخدامها لحل مجموعة متنوعة من المشكلات مثل:

1. إيجاد جميع الرؤوس التي يمكن الوصول إليها من قمة الرأس.
2. معرفة ما إذا كان رسم بياني غير موجه متصل.
3. إيجاد أقصر مسار من رأس واحد إلى جميع الرؤوس الأخرى.
4. لتحديد ما إذا كان الرسم البياني ثنائيًا أم لا.
5. لربط قطر رسم بياني غير موجه.
6. تقسيم الرسم البياني.

يمكن تطبيق Breadth-first search على كل من الرسوم البيانية الموجهة (directed) وغير الموجهة (undirected). يبدأ من الراس المصدر source vertex (s) ويبدأفي استكشاف الرسم البياني للخارجفي جميع الاتجاهات على مستوى حسب المستوى.في هذه العملية، يزور أولاً جميع الرؤوس المجاورة لـ s، ثم يزور الرؤوس التي تبلغ مسافتها اثنين من s، ثم مسافة ثلاثة وهكذا.

العملية المذكورة أعلاه موضحةفي الصورة أدناه. توضح الصورة أدناه كيف يقوم Breadth-First Search بزيارة القمم على جميع المستويات واحدًا تلو الآخر.



Breadth-First Search باستخدام بايثون

لتنفيذ خوارزمية Breadth-First Search باستخدام بايثون، نحتاج أولاً إلى إنشاء هيكل بيانات الطابور وهي هيكل بيانات مجردة تُستخدم لإدراج البيانات وحذفها. لذلك سأقوم أولاً بإنشاء كلاس بايثون"Queue":

```
class Queue():
    def __init__(self):
        self.size = 0
        self.list = []

    def enqueue(self, data):
        self.list.append(data)
        self.size += 1

    def dequeue(self):
        try:
            self.size -= 1
            return self.list.pop(0)
        except Exception as error:
            print(f'{error} is not possible')

    def xprint(self, index):
        print(self.list[index])
```

دعنا الآن نرى كيفية تنفيذ خوارزمية Breadth-first باستخدام بايثون:

```
def breadth_first(graph, root):

    queue = Queue()
    visited_nodes = list()
    queue.enqueue(root)
    visited_nodes.append(root)
    current_node = root

    while queue.size > 0:
        current_node = queue.dequeue()
        adj_nodes = graph[current_node]
        remaining_elements = sorted(set(adj_nodes) -
set(visited_nodes))

        if len(remaining_elements) > 0:
            for element in remaining_elements:
                visited_nodes.append(element)
                queue.enqueue(element)

    return visited_nodes
```

لذلكفي قسم الكود أعلاه، بدأت بتعريف دالة بايثون مثل "breadth_first" التي تقبل معلمتين (الرسم البياني graph والجذر root). ثم أقوم بتهيئة المتغيرات المساعدة.

ثم أستخدم حلقة while للتشغيل حتى يصبح حجم الطابور أكبر من 0، مما يشير إلى العقد التي لم نقم بزيارتها. لذلك قمت للتو بإنشاء دالة لتنفيذ خوارزمية BFS باستخدام بايثون، فيما يلي

كود برنامج التشغيل حيث أنشأت قاموس الرسم البياني الذي سيطبع نتائج دالة breadth_first التي تمر على الرسم البياني والجذر:

```
if __name__ == '__main__':

    graph = dict()

    graph['A'] = ['B', 'G', 'D']
    graph['B'] = ['A', 'F', 'E']
    graph['C'] = ['F', 'H']
    graph['D'] = ['F', 'A']
    graph['E'] = ['B', 'G']
    graph['F'] = ['B', 'D', 'C']
    graph['G'] = ['A', 'E']
    graph['H'] = ['C']

    print(breadth_first(graph, 'A'))
```

```
['A', 'B', 'D', 'G', 'E', 'F', 'C', 'H']
```

الملخص

يمكن استخدام خوارزمية BFS لحساب العديد من الخصائص الأخرى للرسم البياني مثل حساب المسافة أو أقصر مسار. يمكن وصف هذه الخوارزمية على أنها خوارزمية بحث خاصة بالرسم البياني حيث يمكننا اختيار الحد بالكامل في كل خطوة. آمل أن تكون قد أحببت هذه المقالة حول خوارزمية Breadth-First وتطبيقها في بايثون.

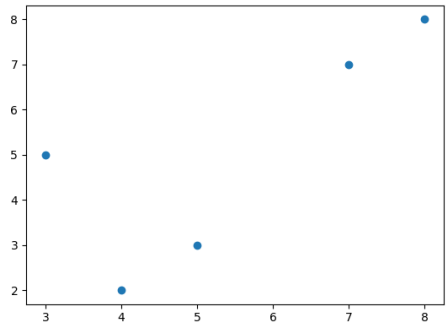
68) رسم التعليقات التوضيحية باستخدام بايثون Plotting Annotations using Python

يعتبر رسم التعليقات التوضيحية (Plotting annotations) أثناء عرض الرسوم البيانية ممارسة جيدة لأنه يجعل الرسوم البيانية الخاصة بك تشرح نفسها بنفسها. قد يكون من الصعب أحيانًا فهم نقاط البيانات التي تشير إلى أي خاصية خاصةفي مخطط التبعثر (scatter plot).في هذه المقالة، سأقدم لك برنامجًا تعليميًا حول رسم التعليقات التوضيحية باستخدام بايثون.

رسم التعليقات التوضيحية باستخدام بايثون

يعتبر رسم التعليقات التوضيحية أثناء تصوير بياناتك ممارسة جيدة لجعل الرسوم البيانية تشرح نفسها بنفسها. يصعب علينا أحيانًا فهم المنحنيات والنقاط التي تمثل نقاط البيانات.في مثل هذه الحالات، يكون استخدام التعليقات التوضيحية مفيدًا جدًا.في لغة برمجة بايثون، توفر مكتبة matplotlib matplotlib.pyplot.annotate مما يجعل من السهل إضافة تعليقات توضيحية لأي نوع من الرسوم البيانية.

على سبيل المثال، ألقِ نظرة على الشكل أدناه، فليس من السهل على الجميع فهم ما هي هذه مخطط التبعثر وما تشير إليه نقاط هذه المخطط.

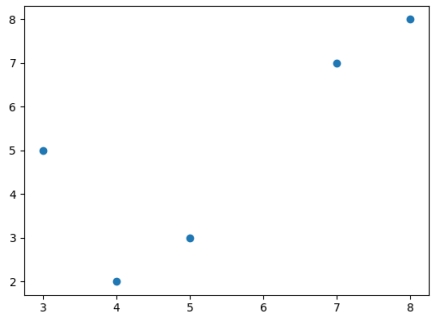


مخطط مبعثر بسيط باستخدام Matplotlib

لذلكفي مثل هذه المواقف، يمكن أن يساعدنا تخطيط التعليقات التوضيحيةفي فهم نقاط البيانات وشرحها. لذلك دعونا أولاً نرسم الشكل أعلاه باستخدام بايثون دون استخدام أي تعليقات توضيحية:

```
import matplotlib.pyplot as plt
x = [3, 5, 7, 8, 4]
y = [5, 3, 7, 8, 2]
```

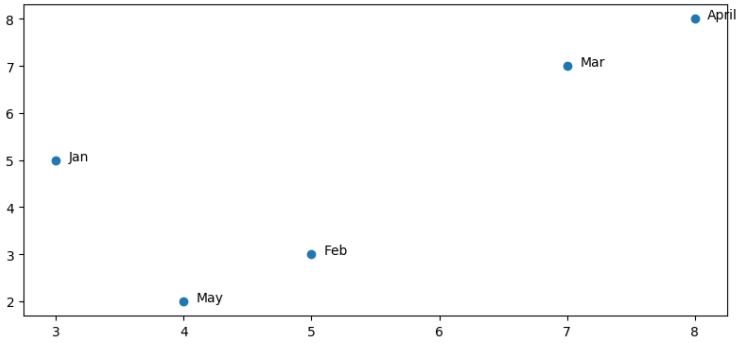
```
plt.scatter(x, y)
plt.show()
```



مخطط مبعثر بدون تعليقات توضيحية

كما ترى، قمنا برسم الشكل بدون استخدام التعليقات التوضيحية. دعنا الآن نرى كيفية وضع تعليق توضيحي على هذا الرسم البياني باستخدام بايثون لجعله واضحًا بذاته. سأقوم بتمثيل نقاط البيانات على أنها النتائج الشهرية:

```
import matplotlib.pyplot as plt
x = [3, 5, 7, 8, 4]
y = [5, 3, 7, 8, 2]
labels = ["Jan", "Feb", "Mar", "April", "May"]
plt.scatter(x, y)
for i, j in enumerate(labels):
    plt.annotate(j, (x[i]+0.10, y[i]), fontsize=10)
plt.show()
```



هذه هي الطريقة التي يمكننا بها بسهولة كتابة تعليق توضيحي لأي نوع من المخطط باستخدام بايثون.في الكود أعلاه، قمت للتو بتقديم قائمة جديدة من التسميات التي تمثل تعليقات توضيحية

لنقاط البيانات الخاصة بمخطط التبعثر، ثم أقوم فقط بتعيين كل عنصر من العناصر الموجودةفي القائمة لكل نقطة بيانات.

الملخص

لذلك يمكن أن يساعدك استخدام التعليقات التوضيحيةفي فهم تصورات البيانات وشرحها بسهولة بالغة. هنا برنامج تعليمي كامل حيث يمكنك معرفة المزيد حول شرح الرسوم البيانية باستخدام بايثون. آمل أن تكون قد أحببت هذه المقالة حول تخطيط التعليقات التوضيحية باستخدام بايثون.

69) تحويل العملات في الوقت الحقيقي مع بايثون Real-time Currency Converter with Python

محول العملات (currency converter) هو تطبيق يستخدم لتحويل قيمة عملة إلى عملة أخرى.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج لإنشاء محول عملاتفي الوقت الفعلي باستخدام بايثون.

كيفية إنشاء محول العملات في الوقت الحقيقي باستخدام بايثون؟

لتحويل عملة إلى أخرى، نحتاج إلى كتابة برنامج لقبول مدخلات المستخدم حيث سيدخل المستخدم مبلغ المال، ومن ثم يتعين على المستخدم اختيار نوع العملة التي يريد المستخدم التحقق من قيمتها.

ثم يجب على برنامجنا فقط عرض النتيجة عن طريق حساب المبلغ المحول كناتج. يجب أن يُظهر تطبيق محول العملات الجيد المبلغ المحولفي الوقت الفعلي، والذي لن يكون ممكنًا إلا إذا كان برنامجنا يعمل على معدلات التحويلفي الوقت الفعلي.

لاستخدام أسعار الصرففي الوقت الفعلي، يمكننا استخدام مكتبة forex-python التي تُستخدم كأداة مجانيةفي بايثون للعمل مع أسعار الصرف وتحويل العملات.

ميزات مكتبة Forex-Python:

بعض الميزات الهامة التي توفرها هذه المكتبة هي:

1. قائمة بجميع أسعار الصرف.
2. سعر BitCoin لجميع العملات.
3. تحويل المبلغ إلى BitCoins.
4. احصل على الأسعار التاريخية لأي يوم منذ عام 1999.
5. سعر تحويل العملة (على سبيل المثال؛ USD إلى INR).
6. تحويل المبلغ من عملة إلى أخرى. ("10 دولارات أمريكية" إلى روبية هندية).
7. رموز العملات.
8. أسماء العملات.

يستخدم rateapi وهو واجهة برمجة تطبيقات API مجانية للعمل مع أسعار الصرففي الوقت الفعلي والتاريخية المنشورة من قبل البنك المركزي الأوروبي.

تحويل العملات في الوقت الحقيقي مع بايثون

لإنشاء محول عملات حقيقي باستخدام بايثون، نحتاج أولاً إلى تثبيت مكتبة forex-python التي يمكن تثبيتها بسهولة باستخدام الأمر pip؛ نقطة تثبيت forex-python.

دعنا الآن نرى كيفية كتابة برنامج لإنشاء محول عملاتفي الوقت الفعلي باستخدام بايثون:

```
from forex_python.converter import CurrencyRates
c = CurrencyRates()
amount = int(input("Enter the amount: "))
from_currency = input("From Currency: ").upper()
to_currency = input("To Currency: ").upper()
print(from_currency, " To ", to_currency, amount)
result = c.convert(from_currency, to_currency, amount)
print(result)
```

Output:

```
Enter the amount: 56000
From Currency: USD
To Currency: INR
USD To INR 56000
4083923.247964
```

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء محول عملات حقيقي باستخدام لغة برمجة بايثون.

70) خوارزمية FizzBuzz باستخدام بايثون FizzBuzz Algorithm using Python

تعد خوارزمية FizzBuzz واحدة من الأسئلة المفضلةفي مقابلات البرمجة. يشير Fizz و Buzz إلى أي رقم مضاعف لـ 3 و 5.في هذه المقالة، سوف أطلعك على كيفية تنفيذ خوارزمية FizzBuzz باستخدام بايثون.

خوارزمية FizzBuzz

تأتي خوارزمية FizzBuzz من لعبة الأطفال. لطالما كانت هذه الخوارزمية واحدة من الأسئلة المفضلةفي مقابلة البرمجة.في هذه المشكلة، يتم إعطاؤك نطاقًا من الأعداد الصحيحة وتحتاج إلى إنتاج مخرجات وفقًا للقواعد المذكورة أدناه:

1. إذا كان العدد الصحيح (x) يقبل القسمة على 3، فيجب استبدال المخرجات بـ "Fizz".
2. إذا كان العدد الصحيح (x) قابلاً للقسمة على 5، فيجب استبدال المخرجات بـ "Buzz".
3. إذا كان العدد الصحيح (x) قابلاً للقسمة على 3 و 5، فيجب استبدال الناتج بـ "FizzBuzz".

مشكلة البرمجة هذه شائعة بين الرقمين 3 و 5، ولكن قد تتمكن من رؤية المزيد من الأرقام المعقدة، لكن منطق حل المشكلة سيظل كما هو.

خوارزمية FizzBuzz باستخدام بايثون

في هذا القسم، سوف أطلعك على كيفية تنفيذ خوارزمية FizzBuzz باستخدام بايثون.

```
for i in range(1, 20):
    if i % 3 == 0 and i % 5 == 0:
        print("FizzBuzz")
    elif i % 3 == 0:
        print("Fizz")
    elif i % 5 == 0:
        print("Buzz")
    else:
        print(i)
```

```
Output:
1
2
Fizz
4
Buzz
Fizz
7
Buzz
11
Fizz
13
14
FizzBuzz
16
17
Fizz
19
```

الملخص

يشير Fizz وBuzz إلى الأرقام القابلة للقسمة على 3 و 5. إذا كان الرقم قابلاً للقسمة على 3، يتم استبداله بـ "Fizz"، إذا كان الرقم قابلاً للقسمة على 5، يتم استبداله بـ "Buzz"، وإذا كان الرقم يمكن القسمة على 3 و 5 ثم يتم استبدال الرقم بـ "FizzBuzz".

أتمنى أن تكون قد أحببت هذه المقالة حول تنفيذ خوارزمية FizzBuzz باستخدام لغة البرمجة بايثون.

71) استخراج الكلمات المفتاحية باستخدام بايثون Extract Keywords using Python

تلعب الكلمات المفتاحية (Keywords) دورًا مهمًا عند قراءة نص طويل لفهم موضوع النص وسياقه. تقوم محركات البحث أيضًا بتحليل الكلمات المفتاحية للمقالة قبل فهرستها.في هذه المقالة، سوف أطلعك على كيفية استخراج الكلمات المفتاحية باستخدام بايثون.

حسنًا، يمكننا أيضًا تدريب نموذج التعلم الآلي الذي سيستخرج الكلمات المفتاحية، ولكن هنا سأقوم فقط بإرشادك حول كيفية استخدام مكتبة بايثون لهذه المهمة حتى يتمكن حتى المبتدئين من فهم كيفية عمل استخلاص الكلمات المفتاحية قبل تدريب نموذج التعلم الآلي.

استخراج الكلمات المفتاحية باستخدام بايثون

هناك العديد من مكتبات بايثون لمهمة استخراج الكلمات المفتاحية، وأفضلها spaCy و Rake-Nltk و YAKE.في هذا البرنامج التعليمي، سأستخدم Rake-NLTK لأنه صديق للمبتدئين وسهل التثبيت. يمكنك تثبيته بسهولة باستخدام الأمر pip؛

```
pip install rake-nltk
```

يرمز RAKE إلى الاستخراج التلقائي السريع للكلمات المفتاحية. إنه مصمم فقط لاستخراج الكلمات المفتاحية باستخدام مكتبة NLTKفي بايثون. دعنا الآن نرى كيفية استخدام هذه المكتبة لاستخراج الكلمات المفتاحية.

سأبدأ أولاً باستيراد وحدة Rake من مكتبة rake-nltk:

```
from rake_nltk import Rake
rake_nltk_var = Rake()
```

الآن سوف أقوم بتخزين بعض النصفي متغير:

```
text = """ I am a programmer from India, and I am here to
guide you
with Data Science, Machine Learning, Python, and C++ for
free .
I hope you will learn a lot in your journey towards Coding ,
Machine Learning and Artificial Intelligence with me""".
```

دعنا الآن نستخرج الكلمات المفتاحية من النص ونطبع الناتج:

```
Output:
['journey towards coding', 'machine learning', 'data science', 'c ++', 'artificial intelligence',
'python', 'programmer', 'lot', 'learn', 'india', 'hope', 'guide', 'free']
```

الملخص

تساعدنا عملية استخراج الكلمات المفتاحيةفي تحديد أهمية الكلماتفي النص. يمكن استخدام هذه المهمة أيضًافي نمذجة الموضوع. من المفيد جدًا استخراج الكلمات المفتاحية لفهرسة المقالات على الويب حتى يتمكن الأشخاص الذين يبحثون عن الكلمات المفتاحية من الحصول على أفضل المقالات لقراءتها.

يتم استخدام هذه التقنية أيضًا بواسطة محركات البحث المختلفة. من الواضح أنهم لا يستخدمون أي مكتبة ولكن تظل العملية كما هي لاستخراج الكلمات المفتاحية. يمكنك تعلم كيفية تدريب نموذج التعلم الآلي لاستخراج الكلمات المفتاحية من هنا.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية استخراج الكلمات المفتاحية باستخدام لغة برمجة بايثون.

72) قراءة البيانات من جداول بيانات Google باستخدام بايثون Read Data From Google Sheets using Python

جداول بيانات Google (Google Sheets) هي خدمة جداول بيانات عبر الإنترنت من Google تتيح لك إنشاء جداول بياناتفي السحابة. تختلف قراءة جداول بيانات Google عن قراءة ملف Microsoft Excel أو CSV باستخدام بايثون. لذلكفي هذه المقالة، سأقوم بتوجيهك حول كيفية قراءة البيانات من جداول بيانات Google باستخدام بايثون لعلوم البيانات.

يمكنك بسهولة إنشاء جدول بياناتفي جداول بيانات Google، فهو مشابه جدًا لجداول بيانات Microsoft Excel. لهذه المهمة، يجب أن يكون لديك جدول بيانات على "جداول بيانات Google" بحيث يمكنك إنشاء ملف Excel أو تحميله إلى "جداول بيانات Google".

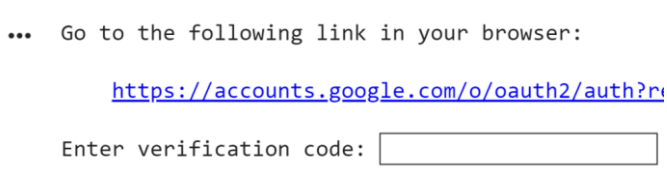
قراءة البيانات من أوراق Excel باستخدام Python

قراءة البيانات من جداول بيانات Google غير ممكنةفي IDE أو محرر كودفي أنظمتنا. لهذه المهمة، تحتاج إلى استخدام Google Colab، وهي خدمة أخرى من Google لإنشاء Jupyter notebooks. يجب أن تكون قد استخدمت Google Colabفي أي مشروع علم بيانات من قبل.

لقراءة مجموعة بيانات من جداول بيانات Google، سأستخدم مجموعة بيانات تم تحميلها بالفعل إلى جداول بيانات Google. فلنتعرف على كيفية ربط Google Colab بجداول البيانات:

```
from google.colab import auth
auth.authenticate_user()
import requests
import gspread
from oauth2client.client import GoogleCredentials
gc =
gspread.authorize(GoogleCredentials.get_application_default())
```

بعد تشغيل الكود أعلاه، ستحصل على رابطفي الإخراج وهو ليس سوى عملية المصادقة للحصول على الرمز الذي تحتاجه لتوصيل Google Colab بجداول البيانات.



ما عليك سوى النقر فوق الارتباط كما هو موضح أعلاه وسيتم نقلك إلى رمز المصادقة الخاص بك. انسخ هذا الرمز وألصقهفي مربع النص كما هو موضح أعلاه واضغط على Enter.

الخطوة النهائية: استخدام بايثون Pandas

لقد ربطت Google Colab بجداول البيانات، فلنرى الآن كيفية قراءة جداول بيانات Google باستخدام بايثون لعلوم البيانات باستخدام مكتبة Pandasفي بايثون:

```
import pandas as pd
sheetname="enrollment" # Enter Sheet name without using
extention
sh = gc.open(sheetname)
worksheet = sh.sheet1
values_list = worksheet.get_all_values()
df = pd.DataFrame(values_list)
df.head()
```

	0	1	2	3	4	5	6	7	8	9	10	11
0	S NO.	JOINING DATE	STUDENT NAME	ENROLLMENT NO.	COURSE	SOURCE	ADDRESS	QUALIFICATION	NEXT FEE DUE DATE	COURSE DURATION	CLASS SCHEDULE	REMARKS
1												
2	1	10-04-2016	OM PRAKASH		ADV. EXCEL	URBAN PRO	DELHI	12TH CLEAR	CLEAR	3 MONTHS	WEEKENDS	COURSE COMPLETE
3	2	10-04-2016	SWATI BANSAL		ADV. EXCEL	URBAN PRO	DELHI	12TH CLEAR	CLEAR	3 MONTHS	WEEKENDS	COURSE COMPLETE
4	3	10-04-2016	ABHINAV BHANDARI		ADV. EXCEL	URBAN PRO	DELHI	12TH CLEAR	CLEAR	3 MONTHS	WEEKENDS	COURSE COMPLETE

الملخص

إليك كيفية قراءة أي مجموعة بيانات من جداول بيانات Google باستخدام بايثون. إذا قمت بتنفيذ هذا الرمزفي IDE أو محرر التعليمات البرمجية، فستحصل على أخطاء، لذلك استخدم هذه الطريقة فقطفيGoogle Colab .

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية قراءة البيانات من جداول بيانات Google بلغة برمجة بايثون.

73) إنشاء الفاتورة في بايثون Creating Invoice with Python

الفاتورة (Invoice) هي قائمة حساب تستخدم كدليل على معاملة بين المشتري والبائع.في هذه المقالة، سوف أطلعك على كيفية إنشاء فاتورة بلغة برمجة بايثون.

إنشاء الفاتورة في بايثون

لإنشاء فاتورة باستخدام بايثون، سأستخدم أساسيات لغة برمجة بايثون. إنها مهمة على مستوى المبتدئين، لذا فهي ستساعدك على تحسين مهاراتكفي البرمجة. لا نحتاج إلى استخدام الحلقات (loops) هنا، فقط طباعة البيانات والتنسيق هو كل ما نحتاجه لهذه المهمة.

سأبدأ بالإعلان عن ست متغيرات كاسم لثلاثة منتجات وسعرها، يمكنك إضافة المزيد من المنتجات إلى قائمتك:

```
product1_name, product1_price = 'Books', 50.95
product2_name, product2_price = 'Computer', 598.99
product3_name, product3_price = 'Monitor', 156.89
```

الآن، دعنا نخزن اسم وعنوان الشركة وهو أمر مهم جدًا لإظهاره أعلى الإيصال:

```
company_name = 'Thecleverprogrammer, inc'.
company_address = '144 Kalka ji'.
company_city = 'New Delh
```

الآن سوف أقوم بتخزين رسالة ترحيبفي متغير لإظهارهفي نهاية الفاتورة، وبعد ذلك سأقوم أيضًا بإنشاء حد للفاتورة:

```
message = 'Thanks for shopping with us today'!
 #create a top border
print(50 * '*')
```

الآن سوف أقوم بطباعة اسم الشركة بتنسيق جدولي، ولن نقوم بطباعة وتنفيذفي هذه المرحلة، ولجعلها تبدو أفضل سأقوم بإنشاء سطور بعد عنوان الشركة بصيغة "=" :

```
print('\t\t{}'.format(company_name.title()))
print('\t\t{}'.format(company_address.title()))
print('\t\t{}'.format(company_city.title()))
# print a line between sections
print('=' * 50)
```

دعنا الآن نطبع الأسماء وسعر المنتجات بالتنسيق الجدولي:

```
print('\tProduct Name\tProduct Price')
# create a print statement for each item
print('\t{}\t\t${}'.format(product1_name.title(),
product1_price))
```

```
print('\t{}\t${}'.format(product2_name.title(),
product2_price))
print('\t{}\t\t${}'.format(product3_name.title(),
product3_price))
```

الآن مرة أخرى سأطبع سطرًا باستخدام "=" ثم سأقوم بطباعة إجمالي المنتجات المذكورة أعلاه:

```
print('=' * 50)
# print out header for section of total
print('\t\t\tTotal')
# calculate total price and print out
total = product1_price + product2_price + product3_price
print('\t\t\t${}'.format(total))
# print a line between sections
print('=' * 50)
```

الآن مرة أخرى سأطبع سطرًا باستخدام "=" ثم سأقوم بطباعة إجمالي المنتجات المذكورة أعلاه:

```
print('\n\t{}\n'.format(message))
```

```
**********************************************
              Thecleverprogrammer, Inc.
              144 Kalka Ji.
              New Delhi
==================================================
       Product Name    Product Price
       Books           $50.95
       Computer        $598.99
       Monitor         $156.89
==================================================
                       Total
                       $806.83
==================================================

       Thanks for shopping with us today!

**********************************************
```

الملخص

يمكنك الآن تشغيل الكود الخاص بك، فلا تترددفي تعديل هذا البرنامج عن طريق إضافة المزيد من المنتجات. يمكنك الحصول على الكود الكامل المستخدمفي هذه المقالة لإنشاء فاتورة باستخدام Python من الأسفل.

```
# create a product and price for three items
product1_name, product1_price = 'Books', 50.95
product2_name, product2_price = 'Computer', 598.99
product3_name, product3_price = 'Monitor', 156.89

# create a company name and information
company_name = 'Thecleverprogrammer, inc.'
company_address = '144 Kalka ji.'
company_city = 'New Delhi'

# declare ending message
```

```
message = 'Thanks for shopping with us today!'

# create a top border
print('*' * 50)

# print company information first using format
print('\t\t{}'.format(company_name.title()))
print('\t\t{}'.format(company_address.title()))
print('\t\t{}'.format(company_city.title()))

# print a line between sections
print('=' * 50)

# print out header for section of items
print('\tProduct Name\tProduct Price')

# create a print statement for each item
print('\t{}\t\t${}'.format(product1_name.title(),
product1_price))
print('\t{}\t${}'.format(product2_name.title(),
product2_price))
print('\t{}\t\t${}'.format(product3_name.title(),
product3_price))

# print a line between sections
print('=' * 50)

# print out header for section of total
print('\t\t\tTotal')

# calculate total price and print out
total = product1_price + product2_price + product3_price
print('\t\t\t${}'.format(total))

# print a line between sections
print('=' * 50)

# output thank you message
print('\n\t{}\n'.format(message))

# create a bottom border
print('*' * 50)
```

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء فاتورة بلغة برمجة بايثون.

74) لعبة المغامرة القائمة على النص مع بايثون Text-Based Adventure Game with Python

لعبة المغامرة القائمة على النص (Text-Based Adventure Game) هي لعبة مغامرة أساسية للغاية يمكن لأي مبتدئفي بايثون العمل عليها.في هذه المقالة، سوف أطلعك على كيفية إنشاء لعبة مغامرات أساسية للغاية تعتمد على النصوص باستخدام بايثون.

ما هي لعبة المغامرة القائمة على النص؟

لعبة المغامرة النصية هي لعبة بسيطة للغاية تستند إلى النص.في هذه اللعبة، يتوفر للمستخدمين خيارات للتعامل مع الموقف ومع كل إدخال يقدمه المستخدم، ستستمر اللعبةفي الزيادة من خلال وضع المزيد من المواقف والمزيد من الخيارات. هذه مهمة جيدة للمبتدئين ليضعوا أيديهم عليها.

في القسم أدناه، ستتعلم كيفية إنشاء لعبة أساسية تعتمد على النصوص باستخدام بايثون. سأوضح لك هنا الفكرة الأساسية لكيفية إنشاء هذه اللعبة ومن ثم يمكنك تعديل أو زيادة حجم هذه اللعبة بمزيد من المواقف ومدخلات المستخدم التي تناسبك.

لعبة المغامرة القائمة على النص مع بايثون

دعنا الآن نرى كيفية إنشاء لعبة مغامرات نصية باستخدام لغة برمجة بايثون:

```
name = str(input("Enter Your Name: "))
print(f"{name} you are stuck at work")
print(" You are still working and suddenly you you saw a
ghost, Now you have two options")
print("1.Run. 2.Jump from the window")
user = int(input("Choose 1 or 2: "))
if user == 1:
    print("You did it")
elif user == 2:
    print("You are not that smart")
else:
    print("Please Check your input")
```

```
Enter Your Name: Aman
Aman you are stuck at work
You are still working and suddenly you you saw a ghost, Now you have two options
1.Run. 2.Jump from the window
Choose 1 or 2: 1
You did it
```

فهم الكود

في الكود أعلاه، أبدأ بمطالبة المستخدم بإدخال اسمه. ثم أقوم بطباعة أن المستخدم عالقفي العمل (the user is stuck at work). ثم استخدمت مرة أخرى عبارة print لإخبار المستخدم بالموقف، وبيان print التالي يخبر المستخدم بالخيارين المختلفين اللذين يمتلكهما.

أخيرًا، أنا فقط أستخدم عبارات if و elif لإظهار النتائج بناءً على مدخلات المستخدم.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء لعبة نصية باستخدام لغة برمجة بايثون.

75) لعبة MAD LIBS باستخدام بايثون لعبة Mad Libs باستخدام بايثون

MAD LIBS هي واحدة من أكثر ألعاب الكلمات تسليةفي العالم. حتى أن البعض قد يسميها واحدة من أعظم التورية على الإطلاق.في هذه المقالة، سأوجهك خلال مهمة بسيطة للمبتدئين: إنشاء لعبة بسيطة Mad Libs باستخدام بايثون.

لعبة MAD LIBS باستخدام بايثون

يعد إنشاء لعبة Mad Libs أحد أبسط المهام للمبتدئين الذين يرغبونفي دخول عالم تطوير البرمجيات. تركز هذه اللعبة بشكل أساسي على اللعب بالسلاسل النصية والمتغيرات والتسلسل، وبنهاية كتابة هذا البرنامج سوف تتعلم كيفية التعامل مع البيانات التي يدخلها المستخدم.

يجب أن يكون تصميم الكود الخاص بـ Mad Libs مع بايثون بطريقة يجب أن تطلب من المستخدمين إدخال سلسلة من الإدخالات التي سيتم اعتبارها Mad Lib. يمكن أن تكون مدخلات المستخدم أي شيء، سواء كانت صفة أو اسمًا أو ضميرًا. بمجرد إدخال جميع الإدخالات، يجب أن يأخذ برنامجك الإدخالات وينظمها جميعًافي شكل قصة.

الآن دعنا نرى كيفية كتابة برنامج لإنشاء لعبة Mad Libs باستخدام بايثون:

```
color = input("Enter a Color: ")
pluralNoun = input("Enter a Plural Noun: ")
celebrity = input("Enter the name of a celebrity: ")
print("Roses are ", color)
print(pluralNoun, " is blue")
print("I like ", celebrity)
```

Output:

```
Enter a Color: red
Enter a Plural Noun: blue
Enter the name of a celebrity: Thecleverprogrammer
Roses are red
blue is blue
I like Thecleverprogrammer
```

آمل أن تكون قد أحببت هذه المقالة حول كيفية إنشاء Mad Libs مع بايثون.

76) انشاء الاختصارات باستخدام بايثون Acronyms Using Python

الاختصار (acronym) هو شكل قصير من الكلمة التي تم إنشاؤها بواسطة كلمات أو عبارات طويلة مثل البرمجة اللغوية العصبية لمعالجة اللغة الطبيعية.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج لإنشاء الاختصارات باستخدام بايثون.

إنشاء الاختصارات باستخدام بايثون

لإنشاء الاختصارات باستخدام بايثون، تحتاج إلى كتابة برنامج بايثون الذي يولد شكلًا قصيرًا من كلمة من جملة معينة. يمكنك القيام بذلك عن طريق التقسيم والفهرسة للحصول على الكلمة الأولى ثم دمجها. دعونا نرى كيفية إنشاء اختصار باستخدام بايثون:

```
user_input = str(input("Enter a Phrase: "))
text = user_input.split()
a = " "
for i in text:
    a = a+str(i[0]).upper()
print(a)
```

```
Enter a Phrase: Artificial Intelligence
AI
```

في الكود أعلاه، آخذ أولاً إدخال مستخدم سلسلة نصية، ثم أستخدم دالة split() في بايثون لتقسيم الجملة. ثم أعلنت عن متغير جديد "a" لتخزين اختصار العبارة.

ثمفي النهاية، أقوم بتشغيل حلقة for فوق المتغير "text" الذي يمثل تقسيم إدخال المستخدم. أثناء تشغيل حلقة for، نقوم بتخزين قيمة الفهرس لـ str[0] لكل كلمة بعد الانقسام وتحويلها إلى تنسيق أحرف كبيرة باستخدام الدالة upper().

الملخص

هذا برنامج بايثون رائع لاختبار مهاراتك المنطقية. تساهم هذه الأنواع من البرامج كثيرًافي مقابلات البرمجة الخاصة بك. لذلك يجب أن تستمرفي تجربة مثل هذه البرامج لتطوير فهم جيد لإنشاء خوارزميات لأداء جيدفي مقابلات البرمجة الخاصة بك.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء الاختصارات باستخدام لغة برمجة بايثون.

77) انشاء منبه مع بايثون Creating Alarm Clock with Python

المنبه (alarm clock) هو ساعة ذات وظيفة يمكن تنشيطها للرنينفي وقت محدد مسبقًا، وتستخدم لإيقاظ شخص ما.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج بايثون لإنشاء منبه باستخدام بايثون.

كيف تصنع منبه باستخدام بايثون؟

كما يوحي العنوان، فإن مهمتنا هنا هي كتابة سكريبت بايثون يُنشئ ساعة منبه. بالنسبة لهذه المهمة، سأستخدم وحدة DateTimeفي بايثون لإنشاء منبه ومكتبة الصوتفي بايثون لتشغيل صوت التنبيه.

تأتي وحدة DateTime مثبتة مسبقًافي لغة برمجة بايثون حتى تتمكن من استيرادها بسهولةفي برنامجك. يمكن تثبيت مكتبة playound بسهولة باستخدام أمر pip؛

```
pip install playsound
```

أتمنى أن تكون قادرًا على تثبيتهفي أنظمتك، فلنرى الآن كيفية كتابة برنامج لإنشاء تنبيه باستخدام بايثون.

المنبه مع بايثون

قبل كتابة البرنامج، يجب أن تعلم أنك بحاجة أيضًا إلى نغمة تنبيه ترنفي وقت التنبيه. لذا يمكنك تنزيل نغمة تنبيه من هنا. الآن بما أننا جاهزون بالمكتبات ونغمة المنبه، فلنرى كيفية كتابة برنامج لإنشاء منبه باستخدام بايثون:

```
from datetime import datetime
from playsound import playsound
alarm_time = input("Enter the time of alarm to be
set:HH:MM:SS\n")
alarm_hour=alarm_time[0:2]
alarm_minute=alarm_time[3:5]
alarm_seconds=alarm_time[6:8]
alarm_period = alarm_time[9:11].upper()
print("Setting up alarm..")
while True:
    now = datetime.now()
    current_hour = now.strftime("%I")
    current_minute = now.strftime("%M")
    current_seconds = now.strftime("%S")
    current_period = now.strftime("%p")
    if(alarm_period==current_period):
        if(alarm_hour==current_hour):
            if(alarm_minute==current_minute):
```

```
            if(alarm_seconds==current_seconds):
                print("Wake Up!")
                playsound('audio.mp3')
                break
```

يجب أن يكون إدخال المستخدم بتنسيق ساعات: دقائق: ثم ثوانٍ. ستبدأفي الاستماع إلى الأغنية حيث ستصل إلى الوقت المحدد. لاختبار الكود الخاص بك، اضبط الوقت بعد دقيقتين أو 3 دقائق من وقت إدخال المستخدم.

الملخص

يمكن تنفيذ هذه الفكرةفي تطبيقات البرامج أيضًا، لذلك لديك الآن فكرة عما يمكن أن يكون مشروعًا جيداًفي بايثون بخلاف مجرد تصميم واجهة المستخدم للتطبيق.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية كتابة برنامج لإنشاء منبه باستخدام بايثون.

78) تقطيع البريد الإلكتروني مع بايثون Email Slicer with Python

تعد أداة تقطيع البريد الإلكتروني (Email slicer) برنامجًا مفيدًا للغاية لفصل اسم المستخدم واسم المجال لعنوان البريد الإلكتروني.في هذه المقالة، سأشرح كيفية كتابة برنامج لإنشاء أداة تقطيع البريد الإلكتروني باستخدام بايثون.

تقطيع البريد الإلكتروني مع بايثون

لإنشاء أداة تقطيع البريد الإلكتروني باستخدام بايثون، تتمثل مهمتنافي كتابة برنامج يمكنه استرداد اسم المستخدم واسم المجال للبريد الإلكتروني. على سبيل المثال، انظر إلى الصورة أدناه والتي تعرض المجال واسم المستخدم لـ "support@thecleverprogrammer.com":

لذلك نحتاج إلى تقسيم البريد الإلكتروني إلى سلسلتين باستخدام "@"كفاصل. دعونا نرى كيفية فصل البريد الإلكتروني واسم النطاق باستخدام بايثون:

```
email = input("Enter Your Email: ").strip()
username = email[:email.index("@")]
domain_name = email[email.index("@")+1:]
format_ = (f"Your user name is '{username}' and your domain is
'{domain_name}'")
print(format_)
```

```
Enter Your Email: support@thecleverprogrammer.com
Your user name is 'support' and your domain is 'thecleverprogrammer.com'
```

الكود أعلاه بسيط للغاية وسهل الفهم. نأخذ مدخلات المستخدم ونستخدم دالة stripفي نفس الوقت لإزالة المساحة البيضاء إن وجدت. ثم نجد فهرس الرمز "@" لإدخال المستخدم. ثم نقوم بتخزين الفهرسفي متغير يعرف باسم domain_name لتقسيم البريد الإلكتروني إلى قسمين؛ اسم المستخدم والمجال.

أخيرًا، نقوم فقط بالتنسيق لطباعة الإخراج. يمكن تحسين الكود أعلاه بمزيد من الأفكار حسب احتياجاتك. كمبتدئ، يجب أن تجرب هذه الأنواع من البرامج لتحسين مهاراتكفي البرمجة. على المدى الطويل، سيساعدك أيضًافي بناء الخوارزميات الخاصة بك وزيادة قدرتك على التفكير المنطقي.

آمل أن تكون قد أحببت هذه المقالة حول كيفية إنشاء ادة تقطيع البريد الإلكتروني باستخدام لغة برمجة بايثون.

79) منشئ القصة مع بايثون Story Generator with Python

هل تعتقد أن الاستخدام الأكثر تعقيداً للوحدة random في بايثون هو أخذ العينات العشوائية؟ لا، يمكننا أيضًا إنشاء قصص عشوائية أو أي شيء يتجاوز ذلك باستخدام الوحدة random. في هذه المقالة، سوف أطلعك على كيفية إنشاء منشئ قصة باستخدام بايثون.

منشئ القصة مع بايثون

مهمتنا هي إنشاء قصة عشوائية في كل مرة يقوم فيها المستخدم بتشغيل البرنامج. سوف أقوم أولاً بتخزين أجزاء القصص في قوائم مختلفة، ثم يمكن استخدام الوحدة random لتحديد الأجزاء العشوائية من القصة المخزنة في قوائم مختلفة:

```
import random
when = ['A few years ago', 'Yesterday', 'Last night', 'A long
time ago','On 20th Jan']
who = ['a rabbit', 'an elephant', 'a mouse', 'a turtle','a
cat']
name = ['Ali', 'Miriam','daniel', 'Hoouk', 'Starwalker']
residence = ['Barcelona','India', 'Germany', 'Venice',
'England']
went = ['cinema', 'university','seminar', 'school', 'laundry']
happened = ['made a lot of friends','Eats a burger', 'found a
secret key', 'solved a mistery', 'wrote a book']
print(random.choice(when) + ', ' + random.choice(who) + ' that
lived in ' + random.choice(residence) + ', went to the ' +
random.choice(went) + ' and ' + random.choice(happened))
```

Output:

On 20th Jan, a rabbit that lived in Germany, went to the laundry and made a lot of friends

قمت أولاً باستيراد الوحدة random ثم قمت بإنشاء أجزاء من القصص في قوائم مختلفة، ثم قمت فقط باختيار أجزاء من القوائم بشكل عشوائي لإنشاء قصة عشوائية.

كمبتدئ، تبحث عن إنشاء تطبيقات برمجية أكثر، وهذا ليس خطأ. لكن قضاء المزيد من الوقت مع البرامج التي تتطلب منك التفكير المنطقي سيساعدك دائمًا بغض النظر عن جانب البرمجة الذي يثير اهتمامك.

الملخص

هناك عدد قليل من المجالات حيث يمكن تحسين الكود أعلاه، ولكن على المستوى الأساسي، فإنه يلبي العديد من متطلبات إنشاء كلمات المرور الآمنة وفقًا لمعايير اليوم. بصفتك مبتدئًا في

بايثون أو أي لغة أخرى، يجب أن تستمرفي تجربة هذه الأنواع من البرامج لأنها تساعدك على استكشاف المزيد من الوظائف وستساعدك على المدى الطويلفي تصميم الخوارزميات الخاصة بك والقيام بعمل رائعفي مقابلات البرمجة.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية كتابة برنامج لإنشاء مولد قصة عشوائي باستخدام بايثون.

80) برنامج بايثون لإنشاء كلمة مرور Python Program to Generate Password

لإنشاء كلمة مرور باستخدام بايثون، نحتاج إلى إنشاء برنامج يأخذ طول كلمة المرور وينشئ كلمة مرور عشوائية بنفس الطول.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج بايثون لإنشاء كلمة مرور.

برنامج بايثون لإنشاء كلمة مرور

لكتابة برنامج بايثون لإنشاء كلمة مرور، أعلن عن سلسلة من الأرقام + الأحرف الكبيرة + الأحرف الصغيرة + الأحرف الخاصة. خذ عينة عشوائية من سلسلة الطول التي قدمها المستخدم:

```
import random
passlen = int(input("enter the length of password"))
s="abcdefghijklmnopqrstuvwxyz01234567890ABCDEFGHIJKLMNOPQRSTUV
WXYZ!@#$%^&*()?"
p = "".join(random.sample(s,passlen ))
print(p)
```

```
enter the length of password7
^H0%koE
```

في الكود أعلاه، قمت أولاً باستيراد وحدة randomفي بايثون، ثم طلبت إدخال المستخدم لطول كلمة المرور. ثم قمت بتخزين الأحرف والأرقام والأحرف الخاصة التي أرغبفي أخذهافي الاعتبار أثناء إنشاء كلمة المرور. ثم أقوم بأخذ عينات عشوائية من خلال ضم طول كلمة المرور والمتغير s، والتي ستنشئفي النهاية كلمة مرور عشوائية.

الملخص

هناك عدد قليل من المجالات حيث يمكن تحسين الكود أعلاه، ولكن على المستوى الأساسي، فإنه يلبي العديد من متطلبات إنشاء كلمات المرور الآمنة وفقًا لمعايير اليوم. بصفتك مبتدئًافي بايثون أو أي لغة أخرى، يجب أن تستمرفي تجربة هذه الأنواع من البرامج لأنها تساعدك على استكشاف المزيد من الدوال وستساعدك على المدى الطويلفي تصميم الخوارزميات الخاصة بك.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية كتابة برنامج بايثون لإنشاء كلمة مرور.

81) رمي النرد محاكي مع بايثون Dice Roll Simulator with Python

يمكن إجراء محاكاة لرمي النرد (Dice Roll) عن طريق اختيار عدد صحيح عشوائي بين 1 و 6 حيث يمكننا استخدام الوحدة random في لغة برمجة بايثون.في هذه المقالة، سوف أطلعك على كيفية إنشاء محاكاة لرمي النرد Dice Roll Simulator باستخدام بايثون.

محاكي رمي النرد مع بايثون

لمحاكاة رمي النرد باستخدام بايثون، سأستخدم الوحدة random في بايثون. يمكن استيراد الوحدة random بسهولة إلى التعليمات البرمجية الخاصة بك لأنها مثبتة مسبقًافي لغة برمجة بايثون.

بعد استيراد الوحدة random، يمكنك الوصول إلى جميع الدوال المضمنةفي الوحدة النمطية. إنها قائمة طويلة جدًا، ولكن لأغراضنا، سنستخدم الدالة random.randint(). ترجع هذه الدالة عددًا صحيحًا عشوائيًا بناءً على البداية والنهاية التي نحددها.

أصغر قيمة لرمي النرد هي 1 وأكبرها هي 6، ويمكن استخدام هذا المنطق لمحاكاة رمي النرد. هذا يعطينا قيم البداية والنهاية لاستخدامهافي دالة random.randint() الخاصة بنا. الآن دعونا نرى كيفية محاكاة رمي النرد باستخدام بايثون:

```
#importing module for random number generation
import random

#range of the values of a dice
min_val = 1
max_val = 6

#to loop the rolling through user input
roll_again = "yes"

#loop
while roll_again == "yes" or roll_again == "y":
    print("Rolling The Dices...")
    print("The Values are :")

    #generating and printing 1st random integer from 1 to 6
    print(random.randint(min_val, max_val))

    #generating and printing 2nd random integer from 1 to 6
    print(random.randint(min_val, max_val))

    #asking user to roll the dice again. Any input other than
yes or y will terminate the loop
```

```
    roll_again = input("Roll the Dices Again?")
```

```
Rolling The Dices…
The Values are :
5
4
Roll the Dices Again?yes
Rolling The Dices…
The Values are :
1
3
```

الملخص

هذه مهمة جيدة ليبدأ بها شخص مبتدئفي بايثون. يساعدك هذا النوع من البرامج على التفكير المنطقي وعلى المدى الطويل، يمكن أن يساعدك أيضًافي إنشاء خوارزميات. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية إنشاء جهاز محاكاة رمي النرد باستخدام بايثون.

82) إنشاء لعبة اختبار باستخدام بايثون Creating a Quiz Game with Python

هل أنت من محبي الاختبارات (quizzes)؟ هل تريد أن تصنع واحدة بنفسك؟في هذه المقالة، سوف أطلعك على كيفية إنشاء لعبة اختبار باستخدام بايثون. سوف أقوم بإنشاء اختبار حيوان (animal quiz) هنا. على الرغم من أن الأسئلة تتعلق بالحيوانات، يمكن تغيير هذا الاختبار بسهولة لتغطية أي موضوع آخر.

منطق لعبة الاختبار مع بايثون

تسأل لعبة الاختبار اللاعب أسئلة عن الحيوانات. لديهم ثلاث فرص للإجابة على كل سؤال لا ترغبفي إجراء الاختبار عليهفي غاية الصعوبة. كل إجابة صحيحة ستحرز نقطة.في نهاية اللعبة، سيكشف البرنامج النتيجة النهائية للاعب.

تستخدم لعبة الاختبار هذه دالة (function)؛ كتلة من التعليمات البرمجية باسم يؤدي مهمة محددة. تتيح لك الدالة استخدام نفس الرمز عدة مرات، دون الحاجة إلى كتابة كل شيءفي كل مرة. تحتوي بايثون على الكثير من الدوال المضمنة، ولكنها تتيح لك أيضًا إنشاء دوالك.

يجب أن يستمر البرنامجفي التحقق مما إذا كانت هناك أي أسئلة يجب طرحها وما إذا كان اللاعب قد استنفد كل فرصه. يتم تخزين النتيجةفي متغير أثناء اللعبة. بمجرد الإجابة على جميع الأسئلة، تنتهي اللعبة.

إنشاء لعبة الاختبار باستخدام بايثون

حان الوقت الآن لإنشاء الاختبار الخاص بك! أولاً، سأقوم بإنشاء الأسئلة وآلية التحقق من الإجابة. بعد ذلك، سأضيف الكود الذي يمنح اللاعب ثلاث محاولات للإجابة على كل سؤال:

```
def check_guess(guess, answer):
    global score
    still_guessing = True
    attempt = 0
    while still_guessing and attempt < 3:
        if guess.lower() == answer.lower():
            print("Correct Answer")
            score = score + 1
            still_guessing = False
        else:
            if attempt < 2:
                guess = input("Sorry Wrong Answer, try again")
            attempt = attempt + 1
    if attempt == 3:
        print("The Correct answer is ",answer )
```

```
score = 0
print("Guess the Animal")
guess1 = input("Which bear lives at the North Pole? ")
check_guess(guess1, "polar bear")
guess2 = input("Which is the fastest land animal? ")
check_guess(guess2, "Cheetah")
guess3 = input("Which is the larget animal? ")
check_guess(guess3, "Blue Whale")
print("Your Score is "+ str(score))
```

```
Correct Answer
Which is the fastest land animal? cheetah
Correct Answer
Which is the larget animal? blue whale
Correct Answer
Your Score is 3
```

الملخص

انشئ الاختبار الخاص بك! اجعله أطول أو أصعب، واستخدم أنواعًا مختلفة من الأسئلة، أو حتى قم بتغيير موضوع الاختبار. يمكنك تجربة بعض أو كل هذه الحيل والتعديلات، ولكن تذكر حفظها كملف بايثون منفصل حتى لا تفسد اللعبة الأصلية.

آمل أن تكون قد أحببت هذه المقالة حول كيفية إنشاء لعبة اختبار باستخدام بايثون.

83) طباعة نص ملون باستخدام بايثون Printing Colored Text with Python

في بايثون، تتيح لنا وحدة Colorama إنشاء نص تيرمينال ملون بسهولة.في هذه المقالة، سأأخذك عبر برنامج تعليمي حول كيفية طباعة النص الملون باستخدام بايثون باستخدام وحدة Coloramaفي بايثون.

ما هو Colorama في بايثون؟

باستخدام وحدة Colorama، يمكننا طباعة نص ملون باستخدام بايثون. يمكننا استخدامه واستدعاء المتغيرات المدمجة الخاصة به والتي هي أسماء مستعارة لرموز ANSI المطلوبة. هذا يجعل الكود الخاص بنا أكثر قابلية للقراءة ويعمل بشكل أفضل مع موجهات أوامر Windows بعد استدعاء colorama.init() في بداية السكريبت الخاص بك.

توفر وحدة Colorama ثلاث خيارات تنسيق رئيسية: أمامي (Fore) وخلفي (Back) ونمط (Style). تسمح لنا هذه بتغيير لون النص الأمامي أو الخلفي ونمطه. الألوان المتوفرة للمقدمة والخلفية هي الأسود والأحمر والأخضر والأصفر والأزرق والأرجواني والسماوي والأبيض.

طباعة نص ملون باستخدام بايثون

تقليديًا، تتم طباعة نص بالألوان الكاملة على الجهاز من خلال سلسلة من أحرف الهروب على أنظمة Linux أو OS X. ومع ذلك، لن يعمل هذا مع أنظمة تشغيل Windows. دعنا الآن نرى كيفية طباعة نص ملون باستخدام بايثون باستخدام وحدة Colorama:

```
import colorama
from colorama import Fore, Back, Style
colorama.init(autoreset=True)

print(Fore.BLUE+Back.YELLOW+"Hi My name is Aman Kharwal "+
Fore.YELLOW+ Back.BLUE+"I am your Machine Learning
Instructor")
print(Back.CYAN+"Hi My name is Aman Kharwal")
print(Fore.RED + Back.GREEN+ "Hi My name is Aman Kharwal")
```

```
Hi My name is Aman Kharwal I am your Machine Learning Instructor
Hi My name is Aman Kharwal
Hi My name is Aman Kharwal
```

من الممكن أيضًا تغيير خصائص النص الأخرى باستخدام أحرف هروب ANSI، على سبيل المثال، إذا أردنا جعل النص أغمق أو أفتح. يمكنك معرفة المزيد حول وحدة بايثون هذه من هنا.

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية طباعة الإخراج الملون باستخدام بايثون باستخدام وحدة Colorama.

84) حساب مؤشر كتلة الجسم (BMI) مع بايثون
Calculating BMI with Python

يتم حساب مؤشر كتلة الجسم (Body Mass Index) أو BMI من وزن الشخص وطوله.في هذه المقالة، سوف أطلعك على كيفية إنشاء آلة حاسبة لمؤشر كتلة الجسم باستخدام بايثون.

ما هو مؤشر كتلة الجسم؟

مؤشر كتلة الجسم هو مقياس للوزن النسبي بناءً على كتلة الفرد وارتفاعه. اليوم، يُستخدم مؤشر كتلة الجسم بشكل شائع لتصنيف الأشخاص على أنهم يعانون من نقص الوزن وزيادة الوزن وحتى السمنة. أيضًا، تم اعتماده من قبل الدول للترويج للأكل الصحي.

يمكن اعتبار مؤشر كتلة الجسم بديلاً للقياسات المباشرة لدهون الجسم. إلى جانب ذلك، يعتبر مؤشر كتلة الجسم وسيلة غير مكلفة وسهلة الأداء لفحص فئات الوزن التي قد تسبب مشاكل صحية.

حساب مؤشر كتلة الجسم مع بايثون

يُحسب مؤشر كتلة الجسم بقسمة وزن الفرد بالكيلوجرام على ارتفاعه بالأمتار، ثم قسمة الإجابة مرة أخرى على طوله. دعنا الآن نرى كيفية إنشاء آلة حاسبة لمؤشر كتلة الجسم باستخدام بايثون:

```
Height=float(input("Enter your height in centimeters: "))
Weight=float(input("Enter your Weight in Kg: "))
Height = Height/100
BMI=Weight/(Height*Height)
print("your Body Mass Index is: ",BMI)
if(BMI>0):
       if(BMI<=16):
              print("you are severely underweight")
       elif(BMI<=18.5):
              print("you are underweight")
       elif(BMI<=25):
              print("you are Healthy")
       elif(BMI<=30):
              print("you are overweight")
       else: print("you are severely overweight")
else:("enter valid details")
```

```
Enter your height in centimeters: 170
Enter your Weight in Kg: 67
your Body Mass Index is: 23.18339100346021
you are Healthy
```

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية حساب مؤشر كتلة الجسم (BMI) باستخدام لغة برمجة بايثون.

85) تحويل فهرنهايت إلى مئوية باستخدام بايثون
Converting Fahrenheit to Celsius with Python

تستخدم معظم الدول حول العالم المقياس المئوي (Celsius) للإشارة إلى درجات الحرارة، لكن الولايات المتحدة لا تزال تستخدم مقياس فهرنهايت (Fahrenheit).في هذه المقالة، سوف آخذك عبر برنامج بسيط للغاية للمبتدئين لتحويل الفهرنهايت إلى درجة مئوية باستخدام لغة برمجة بايثون.

برنامج بايثون لتحويل فهرنهايت إلى مئوية

حساب تحويل درجة الحرارة بسيط. علينا تحويل درجة الحرارة لأن درجات الحرارة المئوية والفهرنهايت لها نقطتا بداية مختلفة؛ 0 درجة مئوية تساوي 32 درجة فهرنهايت. لتحويل الفهرنهايت إلى درجات مئوية، كل ما علينا فعله هو طرح 32 من درجة الحرارة فهرنهايت.

أحيانًا يكون حجم الوحدات مختلفًا أيضًا. تقسم الدرجة المئوية نطاق درجة الحرارة بين نقطتي التجمد والغليان للماء وهو 100 درجة، بينما يقسم فهرنهايت هذا النطاق إلى 180 درجة، لذلك سأضرب أيضًا القيمةفي 9/5 لتحويل 180 درجة إلى 100.

دعونا نرى كيفية القيام بذلك باستخدام بايثون:

```
def convert(s):
    f = float(s)
    c = (f - 32) * 5/9
    return c

print(convert(78))
```

```
25.555555555555557
```

هذه هي الطريقة التي يمكننا بها تحويل درجات الحرارة باستخدام لغة برمجة بايثون. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية كتابة برنامج لتحويل فهرنهايت إلى مئوية باستخدام بايثون.

86) أخذ مدخلات مستخدم متعددة باستخدام بايثون Taking Multiple User Inputs using python

تساعدنا دالة input() في بايثون على إعطاء مدخلات للمستخدم أثناء كتابة البرنامج. ولكن كيف تأخذ مدخلات مستخدم متعددةفي التيرمينال؟في هذه المقالة، سوف أطلعك على كيفية أخذ مدخلات مستخدم متعددة باستخدام بايثون باستخدام حلقة while.

بيان مشكلة أخذ إدخالات مستخدم متعددة باستخدام بايثون

افترض أنه تمت مطالبتك بكتابة برنامج بايثون يتفاعل مع مستخدمفي نافذة وحدة التحكم. قد تقبل إدخالاً لإرسالها إلى قاعدة بيانات، أو قراءة الأرقام لاستخدامهافي عملية حسابية.

مهما كان الغرض، يجب عليك برمجة حلقة تقرأ مدخلًا واحدًا أو أكثر من مدخلات المستخدم من مستخدم يكتب على لوحة المفاتيح ويطبع نتيجة لكل منها. بمعنى آخر، عليك كتابة برنامج حلقة طباعة كلاسيكية.

مدخلات متعددة باستخدام بايثون باستخدام حلقة while

الآن دعنا نرى كيفية حل بيان المشكلة أعلاه عن طريق أخذ مدخلات متعددة باستخدام بايثون باستخدام حلقة while.في بايثون، قد يبدو الكود القياسي لمثل هذه الحلقة التفاعلية كما يلي:

```
while True:
    reply = input("Enter Text: ")
    if reply == 'stop': break
    print(reply)
```

```
Enter Text: hello
hello
Enter Text: how are you
how are you
Enter Text: stop
```

فهم الكود

يستفيد الكود من حلقة whileفي بايثون، وهي أكثر جملة حلقة عامةفي بايثون. تُستخدم دالة الإدخال المضمنة هنا لإدخال وحدة التحكم العامة، وتقوم بطباعة سلسلة الوسيطة الاختيارية الخاصة بها كموجه، وتعيد الاستجابة التي أدخلها المستخدم كسلسلة.

تظهر هنا أيضًا عبارةif أحادية السطر تستخدم القاعدة الخاصة للكتل المتداخلة. يظهر نص عبارة ifفي صف الرأس بعد النقطتين بدلاً من وضع مسافة بادئةفي صف جديد أدناه.

أخيرًا، يتم استخدام تعليمة break للخروج من تعليمة while على الفور. إنه يقفز ببساطة من تعليمة حلقة while ويستمر البرنامج بعد الحلقة. بدون بيان الخروج هذا، ستتكرر while إلى الأبد، لأن اختبارها لا يزال صحيحًا.

آمل أن تكون قد أحببت هذه المقالة حول كيفية أخذ مدخلات مستخدم متعددة باستخدام بايثون باستخدام حلقة while.

87) تحويل الأرقام الرومانية إلى ارقام عشرية باستخدام بايثون Converting Roman Numbers to Decimals using python

أحد أكثر الأسئلة المفضلةفي مقابلة البرمجة هو تحويل الأرقام الرومانية إلى ارقام عشرية.في هذه المقالة، سوف أطلعك على كيفية كتابة برنامج بايثون لتحويل الأرقام الرومانية إلى أرقام عشرية.

كيفية تحويل الأرقام الرومانية إلى أعداد عشرية؟

تذكر أن الأرقام الأساسية ليست هي الأرقام التي يستخدمها الرومان لأن لديهم قيم عد مثل I: 1 ، V: 5 ، X: 10 ، C: 100 ، D: 500 ، M: 1000 ، إلخ.

لذلك نحتاج إلى اتباع المنطق أعلاه لكتابة برنامج لتحويل الأرقام الرومانية إلى أرقام عشرية باستخدام بايثون. لذلك دعونا نلقي نظرة على عملية تحويل الأرقام الرومانية إلى ارقام عشرية:

1. اعملفي طريقك عبر سلسلة الأرقام الرومانية من اليسار إلى اليمين، وفحص حرفين متجاورينفي كل مرة. إذا كنت ترغبفي ذلك، يمكنك أيضًا تحديد اتجاه الحلقات، ولكن لا يهم ما دامت المقارنات يتم تنفيذها وفقًا لذلك.
2. إذاكانت القيمة الموجودة على اليسار أعلى من القيمة الموجودة على اليمين، فقم بطرح العددفي هذا الموضع من القيمة النهائية. خلاف ذلك، فقط قم بإضافته.
3. بمجرد اكتمال العملية تكون القيمة النهائية هي القيمة العشرية المكافئة للرقم الروماني.

برنامج بايثون لتحويل الأعداد الرومانية إلى ارقام عشرية

دعنا الآن نرى كيفية كتابة برنامج لتحويل الأرقام الرومانية إلى أرقام عشرية. سأتبع فقط الشرح أعلاه والذي لا يعدو كونه خوارزمية تحدد عملية كتابة التعليمات البرمجية لتحويل الأرقام الرومانية إلى ارقام عشرية:

```
tallies = {
    'I': 1,
    'V': 5,
    'X': 10,
    'L': 50,
    'C': 100,
    'D': 500,
    'M': 1000,
    # specify more numerals if you wish
}

def RomanNumeralToDecimal(romanNumeral):
    sum = 0
```

```
    for i in range(len(romanNumeral) - 1):
        left = romanNumeral[i]
        right = romanNumeral[i + 1]
        if tallies[left] < tallies[right]:
            sum -= tallies[left]
        else:
            sum += tallies[left]
    sum += tallies[romanNumeral[-1]]
    return sum
```

أتمنى أن تكون قد أحببت هذه المقالة حول كيفية تحويل رقم روماني إلى رقم عشري باستخدام لغة برمجة بايثون. هذا حل لواحد من أكثر الأسئلة المفضلة لدى المحاورفي مقابلة البرمجة.

88) ارتباط بيرسون باستخدام بايثون Pearson Correlation using Python

عندما ترتبط ميزتان أو أكثر ببعضهما البعض بطريقة إذا زادت قيمة 1 ميزات (features)، فإن قيمة الميزة الأخرى تزداد أو تنقص أيضًا. هذا ما يعنيه الارتباط.في هذه المقالة، سأوجهك خلال تنفيذ ارتباط بيرسون (Pearson Correlation) باستخدام بايثون.

ما هو الارتباط؟

الارتباط (Correlation) يعني إيجاد العلاقة بين المتغيرات.في علم البيانات، نستخدم الارتباط للعثور على الميزات التي ترتبط ارتباطًا إيجابيًا وسلبيًا ببعضها البعض حتى نتمكن من اختيار أفضل الميزات لتدريب نموذج التعلم الآلي.

تكون درجة الارتباط بين 1 ـ و 1. عندما تكون قيمة الارتباط بين السمات 1، فإن هذه الميزات ترتبط ارتباطًا إيجابيًا ببعضها البعض، وعندما تكون قيمة الارتباط بين السمات هي 1 ـ، فهذا يعني أن ترتبط هذه الميزات ارتباطًا سلبيًا ببعضها البعض.

عندما تكون قيمة الارتباط بين الميزات مساوية لـ 0، يمكننا القول أنه لا يوجد ارتباط بين الميزات.في التعلم الآلي، يمكننا استخدام الارتباط للتحقق من العلاقة بين جميع الميزات المتعلقة بالتسمية المستهدفة (target label). حتى نتمكن من تحديد هذه الميزات لتدريب نموذج التعلم الآلي الذي يرتبط ارتباطًا وثيقًا بالتسمية المستهدفة.

ارتباط بيرسون

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r	correlation coefficient
x_i	values of the x-variable in a sample
$\bar{x}$	mean of the values of the x-variable
y_i	values of the y-variable in a sample
$\bar{y}$	mean of the values of the y-variable

ارتباط بيرسون هو أسلوب إحصائي لقياس درجة العلاقة الخطية بين ميزتين أو أكثر. العرض والطلب هما أفضل الأمثلة لفهم ارتباط بيرسون. على سبيل المثال، سيزداد عرض المنتج عندما يزداد الطلب على المنتج، وينخفض المعروض من المنتج عندما يزداد الطلب على هذا المنتج.

وبالتالي، وفقًا للمثال أعلاه، هناك علاقة إيجابية بين الطلب والعرض للمنتج. آمل أن تفهم الآن ما هو الارتباط وما هو ارتباط بيرسون ولماذا نستخدمه قبل تدريب نماذج التعلم الآلي.في القسم أدناه، سوف أطلعك على كيفية حساب الارتباط باستخدام بايثون.

ارتباط بيرسون باستخدام بايثون

قبل أن ننفذ ارتباط بيرسون باستخدام بايثون، دعنا نلقي نظرة على بعض النقاط المهمة لفهم النتيجة:

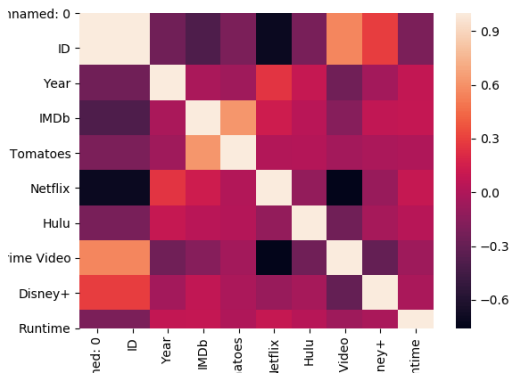
1. تشير القيم الموجبة إلى ارتباط خطي موجب.
2. القيم السالبة تعني ارتباط خطي سلبي.
3. 0 يعني عدم وجود ارتباط خطي.
4. كلما كانت القيمة أقرب إلى 1 أو 1ـ ، كان الارتباط الخطي أقوى.

دعنا الآن نرى كيفية تنفيذ ارتباط بيرسون باستخدام بايثون:

مجموعة البيانات

```
import pandas as pd
movies = pd.read_csv("MoviesOnStreamingPlatforms_updated.csv")
movies['Rotten Tomatoes'] = movies["Rotten
Tomatoes"].str.replace("%", "").astype(float)
movies.drop("Type", inplace=True, axis=1)
correlations = movies.corr(method='pearson')
# Correlation Between All The Features
print(correlations)

# Correlation Between A Particular column "Year"
print(correlations["Year"])

# Visualizing Correlation
import seaborn as sns

import matplotlib.pyplot as plt
sns.heatmap(correlations)
plt.show()
```

```
                Unnamed: 0        ID      Year      IMDb  ...      Hulu  P
Unnamed: 0        1.000000  1.000000 -0.254391 -0.399953  ... -0.219737
ID                1.000000  1.000000 -0.254391 -0.399953  ... -0.219737
Year             -0.254391 -0.254391  1.000000 -0.021181  ...  0.098009
IMDb             -0.399953 -0.399953 -0.021181  1.000000  ...  0.042191
Rotten Tomatoes  -0.201452 -0.201452 -0.057137  0.616320  ...  0.020373
Netflix          -0.708680 -0.708680  0.258533  0.135105  ... -0.107911
Hulu             -0.219737 -0.219737  0.098009  0.042191  ...  1.000000
Prime Video       0.554120  0.554120 -0.253377 -0.163447  ... -0.255641
Disney+           0.287011  0.287011 -0.046819  0.075895  ... -0.034317
Runtime          -0.206003 -0.206003  0.081984  0.088987  ...  0.033985

[10 rows x 10 columns]


Unnamed: 0       -0.254391
ID               -0.254391
Year              1.000000
IMDb             -0.021181
Rotten Tomatoes  -0.057137
Netflix           0.258533
Hulu              0.098009
Prime Video      -0.253377
Disney+          -0.046819
Runtime           0.081984
Name: Year, dtype: float64
```



أتمنى أن تكون قد أحببت هذه المقالة حول ماهية ارتباط بيرسون وتنفيذه باستخدام بايثون.

89) Treemap باستخدام بايثون Treemap using Python

يتم استخدام Treemap لتصوير البيانات الهرمية كمجموعة من المستطيلات المتداخلة. إنها أداة تصوير البيانات لعرض البيانات المهيكلةفي هيكل شجرة باستخدام المستطيلات المتداخلة. في هذه المقالة، سوف أطلعك على كيفية تصوير Treemap باستخدام بايثون.

ما هو Treemap؟

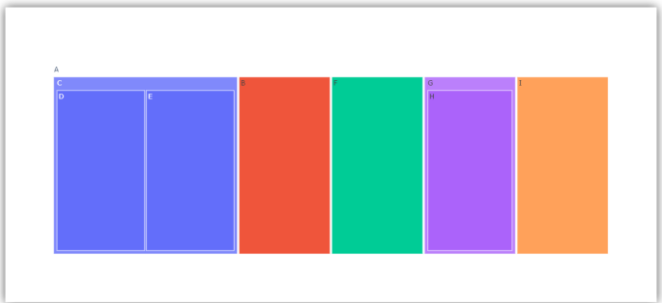
Treemap عبارة عن تقنية تصوير البيانات تُستخدم لتصور البيانات الهرمية باستخدام المستطيلات المتداخلة. يعرض Treemap بيانات هرمية بحيث يتلقى كل فرع من فروع الشجرة مستطيلًا مملوءًا بمستطيلات أصغر مثل الفروع الفرعية.

آمل أن تفهم الآن ماهية Treemap والهيكل الذي تعرض فيه البيانات الهرمية.في القسم أدناه، سأوجهك خلال برنامج تعليمي حول كيفية تصوير Treemap باستخدام لغة برمجة بايثون.

تصوير Treemap باستخدام بايثون

هناك العديد من مكتبات تصوير البياناتفي بايثون يمكننا استخدامها لتصوير Treemap، ولكن أسهل طريقة هي استخدام مكتبة plotlyفي بايثون. فيما يلي كيف يمكننا استخدام الرسم البياني لتصوير Treemap باستخدام بايثون:

```
import plotly.graph_objects as go

fig = go.Figure(go.Treemap(
    labels = ["A","B", "C", "D", "E", "F", "G", "H", "I"],
    parents = ["", "A", "A", "C", "C", "A", "A", "G", "A"]
))

fig.show()
```



الملخص

يتم استخدام Treemap لتصوير البيانات الهرمية كمجموعة من المستطيلات المتداخلة. يعرض البيانات الهرمية بطريقة يتلقى فيها كل فرع من فروع الشجرة مستطيلًا مملوءًا بمستطيلات أصغر مثل الفروع الفرعية. آمل أن تكون قد أحببت هذه المقالةفي برنامج تعليمي حول كيفية تصوير Treemap باستخدام بايثون.

90) تحويل الصورة إلى مصفوفة باستخدام بايثون Converting Image to Array using Python

نحتاج إلى تحويل صورة إلى مصفوفة (array) لاستخدامهافي أي نوع من مهام علم البيانات حيث نحتاج إلى فهم ميزات الصورة. يعد تحويل الصور إلى مصفوفة سهلاً مثل تحويل النص إلى بيانات رقمية. لذلك، إذا كنت تريد معرفة كيفية تحويل الصور إلى مصفوفة، فهذه المقالة مناسبة لك.في هذه المقالة، سأقدم برنامجًا تعليميًا حول كيفية تحويل صورة إلى مصفوفة باستخدام بايثون.

كيفية تحويل صورة إلى مصفوفة باستخدام بايثون؟

يعد تحويل صورة إلى مصفوفة مهمة شديدة الاهمية لتدريب نموذج التعلم الآلي بناءً على ميزات الصورة. نحن نستخدم مكتبة NumPyفي بايثون بشكل أساسي للعمل مع المصفوفات حتى نتمكن أيضًا من استخدامها لتحويل الصور إلى مصفوفة. بخلاف NumPy، يمكننا أيضًا استخدام مكتبة Kerasفي بايثون لنفس المهمة.

لذلكفي القسم أدناه، سوف آخذك من خلال برنامج تعليمي حول كيفية تحويل صورة إلى مصفوفة باستخدام مكتبات NumPy و Kerasفي بايثون.

تحويل صورة إلى مصفوفة باستخدام NumPy:

يمكننا استخدام NumPy لتحويل الصور إلى مصفوفات، لكن ليس لها دالة لقراءة الصور. لذلك نحتاج أولاً إلى استخدام مكتبة PILفي بايثون لقراءة صورة. إذا لم تستخدمه من قبل، فيمكنك تثبيته بسهولة باستخدام الأمر pip:

```
pip install Pillow
```

الآن إليك كيف يمكننا قراءة صورة باستخدام مكتبة PILفي بايثون:

```
from PIL import Image
image = Image.open('aman.png')
```

بعد قراءة الصورة، إليك كيفية تحويلها إلى مصفوفة باستخدام مكتبة NumPyفي بايثون:

```
from numpy import asarray
data = asarray(image)
print(data)
```

```
[[[188 216 238]
  [188 216 238]
  [187 215 237]
  ...
  [203 219 234]
  [203 219 234]
```

```
 [203 219 234]]

[[187 215 237]
 [188 216 238]
 [188 216 238]
 ...
 [203 219 234]
 [204 220 235]
 [204 220 235]]

[[184 212 234]
 [186 214 236]
 [185 213 235]
 ...
 [205 221 236]
 [205 221 236]
 [205 221 236]]

...

[[131 154 172]
 [132 155 173]
 [134 156 174]
 ...
 [123 136 145]
 [126 139 148]
 [129 142 151]]

[[129 152 170]
 [130 153 172]
 [134 156 173]
 ...
 [123 136 145]
 [124 137 146]
 [127 140 149]]

[[130 153 171]
 [131 154 172]
 [132 154 172]
 ...
 [122 135 144]
 [125 138 147]
 [128 141 150]]]
```

تحويل صورة إلى مصفوفة باستخدام Keras:

يمكننا استخدام مكتبة Kerasفي بايثون لقراءة الصور وتحويلها إلى مصفوفات. إليك كيفية قراءة وتحويل صورة إلى مصفوفة باستخدام مكتبة Kerasفي بايثون:

```
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
img = load_img("aman.png")
data = img_to_array(img)
print(data)
```

```
[[[188. 216. 238.]
  [188. 216. 238.]
  [187. 215. 237.]
```

```
...
 [203. 219. 234.]
 [203. 219. 234.]
 [203. 219. 234.]]

[[187. 215. 237.]
 [188. 216. 238.]
 [188. 216. 238.]
 ...
 [203. 219. 234.]
 [204. 220. 235.]
 [204. 220. 235.]]

[[184. 212. 234.]
 [186. 214. 236.]
 [185. 213. 235.]
 ...
 [205. 221. 236.]
 [205. 221. 236.]
 [205. 221. 236.]]

...

[[131. 154. 172.]
 [132. 155. 173.]
 [134. 156. 174.]
 ...
 [123. 136. 145.]
 [126. 139. 148.]
 [129. 142. 151.]]

[[129. 152. 170.]
 [130. 153. 172.]
 [134. 156. 173.]
 ...
 [123. 136. 145.]
 [124. 137. 146.]
 [127. 140. 149.]]

[[130. 153. 171.]
 [131. 154. 172.]
 [132. 154. 172.]
 ...
 [122. 135. 144.]
 [125. 138. 147.]
 [128. 141. 150.]]]
```

الملخص

هذه هي الطريقة التي يمكننا بها تحويل الصور بسهولة إلى مصفوفة باستخدام لغة برمجة بايثون. يعد تحويل صورة إلى مصفوفة مهمة شديدة الاهمية لتدريب نموذج التعلم الآلي بناءً على ميزات الصورة. أتمنى أن تكون قد أحببت هذه المقالة حول كيفية تحويل الصور إلى مصفوفات باستخدام بايثون.

91) تجريف IMDb باستخدام بايثون Scraping IMDb using Python

IMDb هي قاعدة بيانات عبر الإنترنت تحتوي على بيانات حول الأفلام والبرامج التلفزيونية وعروض البث وألعاب الفيديو والمراجعات والتقييمات وجميع البيانات الأخرى المتعلقة بالترفيه. نظرًا لكونها قاعدة بيانات عبر الإنترنت، فإنها توفر واجهة برمجة تطبيقات API حتى نتمكن من جمع البيانات من IMDb لمختلف مهام علوم البيانات. لذلك، إذا كنت تريد معرفة كيفية تجريف البيانات من IMDb، فهذه المقالة مناسبة لك.في هذه المقالة، سوف أطلعك على كيفية التجريف من IMDb باستخدام بايثون.

تجريف IMDb باستخدام بايثون

يرمز IMDb إلى قاعدة بيانات الأفلام على الإنترنت. إنها قاعدة بيانات عبر الإنترنت للأفلام والتلفزيون ومختلف المحتويات الترفيهية الأخرى. تحظى مجموعات بيانات IMDb بشعبية كبيرة بين مجتمع علوم البيانات. نظرًا لكونها قاعدة بيانات عبر الإنترنت، فإنها توفر واجهة برمجة التطبيقات API الخاصة بها لجمع البيانات. يمكنك تثبيت واجهة برمجة التطبيقات هذهفي نظامك باستخدام الأمر pip:

```
pip install imdb
```

الآن دعنا نرى كيف نجرف من IMDb باستخدام بايثون. سأبدأ هذه المهمة بالبحث عن معرف الفيلم بشكل عشوائي لمعرفة الفيلم المرتبط بالمعرف:

```
from imdb import IMDb
movie = IMDb().get_movie('012346')
print(movie)
```

```
Output:
The Kentuckians
```

الآن دعونا نلقي نظرة على مخرجي هذا الفيلم :

```
for i in movie["directors"]:
    print(i)
```

```
Output:
Charles Maigne
```

أخيرًا، سأستخدم إحدى الطرق الأكثر شيوعًا لهذه المكتبة وهي إلقاء نظرة على أفضل 250 فيلمًا في IMDb:

```
movies = IMDb().get_top250_movies()
for i in movies:
    print(i)
```

```
Output:
The Shawshank Redemption
The Godfather
The Godfather: Part II
The Dark Knight
12 Angry Men
Schindler's List
...
```

الملخص

هذه هي الطريقة التي يمكننا بها جمع البيانات من IMDb باستخدام لغة برمجة بايثون. إنها قاعدة بيانات عبر الإنترنت تحتوي على بيانات حول الأفلام والبرامج التلفزيونية وعروض البث وألعاب الفيديو والمراجعات والتقييمات وجميع البيانات الأخرى المتعلقة بالترفيه. آمل أن تكون قد أحببت هذه المقالة حول كيفية تجريف IMDb باستخدام بايثون.

المصدر

- Python Projects with Source Code, Aman Kharwal,https://thecleverprogrammer.com/2021/01/14/python-projects-with-source-code/.