DSA جداول تجزئة

جدول تجزئة

.جدول التجزئة هو عبارة عن بنية بيانات مصممة لتكون سريعة في العمل

والسبب في تفضيل جداول التجزئة في بعض الأحيان بدلاً من المصفوفات أو القوائم المرتبطة هو أن البحث عن البيانات وإضافتها .وحذفها يمكن أن يتم بسرعة كبيرة، حتى بالنسبة للكميات الكبيرة من البيانات

في أ، يستغرق العثور على الشخص "بوب" وقتًا لأن علينا الانتقال من عقدة إلى أخرى، والتحقق من كل عقدة، حتى يتم العثور على ."العقدة التي تحتوي على "بوب

ويمكن أن يكون العثور على "بوب" في جدول تجزئة سريعًا إذا كنا نعرف الفهرس، ولكن عندما نعرف فقط اسم "بوب"، نحتاج إلى .مقارنة كل عنصر (كما هو الحال مع القوائم المرتبطة)، وهذا يستغرق وقتًا

ولكن مع الجدول التجزئي، فإن العثور على "بوب" يتم بسرعة كبيرة لأن هناك طريقة للانتقال مباشرةً إلى مكان تخزين "بوب"، .باستخدام ما يسمى بدالة تجزئة

بناء جدول تجزئة من الصفر

.للحصول على فكرة عن ماهية جدول التجزئة، دعنا نحاول بناء واحد من الصفر، لتخزين الأسماء الأولى الفريدة داخله

:سنبني مجموعة التجزئة في 5 خطوات

1. .البدء بمصفوفة
2. .تخزين الأسماء باستخدام دالة تجزئة
3. .البحث عن عنصر باستخدام دالة تجزئة
4. .التعامل مع التصادمات
5. .مثال على شيفرة مجموعة التجزئة الأساسية والمحاكاة

الخطوة 1: البدء بمصفوفة

:باستخدام مصفوفة، يمكننا تخزين الأسماء هكذا

```
my_array = ["بيت"، "جونز"، "ليزا"، "بوب"، "سيري"]
```

."للعثور على "بوب" في هذه المصفوفة، نحتاج إلى مقارنة كل اسم، عنصرًا بعنصر، حتى نجد "بوب

إذا كانت المصفوفة مرتبة أبجديًا، يمكننا استخدام البحث الثنائي للعثور على الاسم بسرعة، لكن إدراج أو حذف الأسماء في المصفوفة .يعني عملية كبيرة لتبديل العناصر في الذاكرة

لجعل التفاعل مع قائمة الأسماء سريعًا جدًا، دعنا نستخدم جدول تجزئة لهذا الغرض بدلًا من ذلك، أو مجموعة تجزئة، وهي نسخة .مبسطة من جدول التجزئة

لتبسيط الأمر، دعنا نفترض أن هناك 10 أسماء على الأكثر في القائمة، لذا يجب أن تكون المصفوفة بحجم ثابت من 10 عناصر. عند الحديث عن جداول التجزئة، يُطلق على كل عنصر من هذه العناصر اسم دلو.

```
my_hash_set = [بلا، بلا، بلا، بلا، بلا، بلا، بلا، بلا، بلا، بلا]
```

الخطوة 2: تخزين الأسماء باستخدام دالة تجزئة

تأتي الآن الطريقة الخاصة التي نتفاعل بها مع مجموعة التجزئة التي نصنعها.

نريد تخزين الاسم مباشرةً في مكانه الصحيح في المصفوفة، وهنا يأتي دور دالة التجزئة.

يمكن صنع دالة التجزئة بعدة طرق، الأمر متروك لمنشئ جدول التجزئة. إحدى الطرق الشائعة هي إيجاد طريقة لتحويل القيمة إلى رقم لكل حرف، Unicode يساوي أحد أرقام فهرس مجموعة التجزئة، في هذه الحالة رقم من 0 إلى 9. في مثالنا هذا سنستخدم رقم ونلخصها ونجري عملية تعديل 10 للحصول على أرقام الفهرس من 0 إلى 9.

مثال

```
def hash_function(value):
مجموع_من_الأحرف = 0
في القيمة char لـ
sum_of_chars += ord(char)

إرجاع sum_of_chars % 10

طباعة("""ب' لديه رمز التجزئة:",hash_function('Bob'))
```

على 98. بجمع هذه الرموز معًا نحصل "b" على 111، والحرف "o" والحرف Unicode 66، على نقطة رمز "B" يحتوي الحرف على 275. مودولو 10 من 275 يساوي 5، لذا يجب تخزين "بوب" كعنصر مصفوفة عند الفهرس 5.

الرقم الذي تُرجعه دالة التجزئة يسمى رمز التجزئة.

رقم يونيكود: يتم تخزين كل شيء في حواسيبنا على شكل أرقام، ونقطة رمز يونيكود هي رقم فريد موجود لكل حرف. على سبيل المثال، له رقم يونيكود (يسمى أيضاً نقطة كود يونيكود) `65`. فقط جربه في المحاكاة أدناه. انظر لمزيد من المعلومات حول كيفية `A` الحرف تمثيل الأحرف كأرقام.

في الرياضيات). تقسم عملية مودولو عدد ما على (\mod\) مودولو: عملية رياضية، تُكتب على شكل `%` في معظم لغات البرمجة (أو عدد آخر، وتعطينا التذكير الناتج. لذا، على سبيل المثال، `7 % 3` سيعطينا التذكير `1`. (قسمة 7 تفاحات على 3 أشخاص، يعني أن كل شخص يحصل على تفاحتين، مع وجود تفاحة واحدة).

بعد تخزين "بوب" حيث يخبرنا رمز التجزئة (الفهرس 5)، تبدو مصفوفتنا الآن هكذا

```
my_hash_set = [None,None,None,None,None,'Bob',None,None,None,None]
```

يمكننا استخدام دالة التجزئة لمعرفة مكان تخزين الأسماء الأخرى "بيت" و"جونز" و"ليزا" و"سيري" أيضًا.

بعد استخدام دالة التجزئة لتخزين هذه الأسماء في الموضع الصحيح، ستبدو المصفوفة بهذا الشكل:

```
my_hash_set = [بلا، 'جونز',بلا، 'ليزا',بلا، 'بوب',بلا، 'سيري',بلا، 'سيري',بيت',بلا]
```

الخطوة 3: البحث عن اسم باستخدام دالة تجزئة

لقد أنشأنا الآن مجموعة تجزئة أساسية للغاية، لأننا لم نعد بحاجة إلى التحقق من المصفوفة عنصرًا تلو الآخر لمعرفة ما إذا كان "بيت" موجودًا فيها، يمكننا فقط استخدام دالة التجزئة للانتقال مباشرةً إلى العنصر الصحيح!

لمعرفة ما إذا كان "بيت" مخزّنًا في المصفوفة، نعطي اسم "بيت" لدالة التجزئة الخاصة بنا، فنحصل على رمز التجزئة 9، وننتقل مباشرةً إلى العنصر الموجود عند المؤشر 9، وها هو هناك. وجدنا "بيت" دون التحقق من أي عناصر أخرى.

مثال

```
my_hash_set = [بلا، 'جونز',بلا، 'ليزا',بلا، 'بوب',بلا، 'سيري',بلا، 'بيت',بلا]

def hash_function(value):
مجموع_من_الأحرف = 0
في القيمة char ل:
sum_of_chars += ord (char)

إرجاع sum_of_chars % 10

(تعريف يحتوي (الاسم:
(فهرس = تجزئة_وظيفة(الاسم
الاسم = = my_hash_hash_set[index] إرجاع

(('طباعة ("'بيت' موجود في مجموعة التجزئة:",يحتوي على('بيت
```

عند حذف اسم من مجموعة التجزئة الخاصة بنا، يمكننا أيضًا استخدام دالة التجزئة للانتقال مباشرةً إلى مكان الاسم، وتعيين قيمة هذا العنصر إلى لا شيء.

الخطوة 4: التعامل مع التصادمات

لنضف أيضًا "ستيوارت" إلى مجموعة التجزئة.

نعطي "ستيوارت" إلى دالة التجزئة، ونحصل على رمز التجزئة 3، مما يعني أن "ستيوارت" يجب أن يُخزَّن في الفهرس 3.

تؤدي محاولة تخزين "ستيوارت" إلى ما يسمى بالتصادم، لأن "ليزا" مخزنة بالفعل في الفهرس 3.

لإصلاح التصادم، يمكننا إفساح المجال للمزيد من العناصر في نفس الدلو، وحل مشكلة التصادم بهذه الطريقة يُسمّى "التسلسل". يمكننا إفساح المجال للمزيد من العناصر في نفس الدلو عن طريق تنفيذ كل دلو كقائمة مرتبطة أو كمصفوفة.

بعد تنفيذ كل دلو على شكل مصفوفة، لإعطاء مساحة لأكثر من اسم في كل دلو، يمكن أيضًا تخزين "ستيوارت" في الفهرس 3، وتبدو مجموعة التجزئة الآن هكذا

```
my_hash_set = [
[لا شيء],
["جونز"],
[بلا],
['ليزا'، 'ستيوارت'],
[بلا],
['بوب'],
[لا يوجد],
['سيري'],
['بيت'],
```

[بلا]
]
يعني البحث عن "ستيوارت" في مجموعة التجزئة الآن أنه باستخدام دالة التجزئة ينتهي بنا الأمر مباشرةً في الدلو 3، ولكن يجب أن نتحقق أولاً من "ليزا" في ذلك الدلو، قبل أن نجد "ستيوارت" كعنصر ثانٍ في الدلو 3.

الخطوة 5: مثال على كود مجموعة التجزئة والمحاكاة

لإكمال شيفرة مجموعة التجزئة الأساسية لدينا، لنحصل على دوال لإضافة الأسماء والبحث عنها في مجموعة التجزئة، والتي أصبحت الآن مصفوفة ثنائية الأبعاد.

قم بتشغيل المثال البرمجي أدناه، وجربه بقيم مختلفة للحصول على فهم أفضل لكيفية عمل مجموعة تجزئة.

مثال

```
my_hash_set = [
[لا شيء],
['جونز'],
[بلا],
['ليزا'],
[بلا],
['بوب'],
[لا يوجد],
['سيري'],
['بيت'],
[بلا]
]

def hash_function(value):
إرجاع مجموع(ord(char) ل char في القيمة) % 10

: (تعريف إضافة (قيمة
(الفهرس = دالة التجزئة (القيمة
[الفهرس] my_hash_set = الدلو
إذا لم تكن القيمة في الدلو
(يُلحق الدلو (القيمة

: (تعريف يحتوي (قيمة
(الفهرس = دالة التجزئة (القيمة
[الفهرس] my_hash_set = الدلو
إرجاع القيمة في الدلو

('إضافة ('ستيوارت

(my_hash_set) طباعة
(('طباعة ('يحتوي على ستيوارت: ', يحتوي على ('ستيوارت
```

تعرض الصفحتان التاليتان تطبيقات أفضل وأكثر تفصيلاً لمجموعات التجزئة وجداول التجزئة.

جرب محاكاة Hash Set أدناه للحصول على فكرة أفضل عن كيفية عمل Hash Set من حيث المبدأ.

استخدامات جداول التجزئة

جداول التجزئة رائعة لـ

- (التحقق من وجود شيء ما في مجموعة (مثل العثور على كتاب في مكتبة.
- (تخزين العناصر الفريدة والعثور عليها بسرعة (مثل تخزين أرقام الهواتف.
- (ربط القيم بالمفاتيح (مثل ربط الأسماء بأرقام الهواتف.

السبب الأكثر أهمية في كون جداول التجزئة رائعة لهذه الأشياء هو أن جداول التجزئة سريعة جدًا مقارنةً بالمصفوفات والقوائم للبحث والحذف، بينما جداول التجزئة لها \(O(n)\ المرتبطة، خاصةً للمجموعات الكبيرة. المصفوفات والقوائم المرتبطة لها تعقيد زمني . فقط في المتوسط! اقرأ المزيد عن تعقيد الوقت \(O(1)\

مجموعة التجزئة مقابل خريطة التجزئة

.يمكن أن يكون جدول التجزئة مجموعة تجزئة أو خريطة تجزئة. تصف الصفحتان التاليتان هياكل البيانات هذه بمزيد من التفصيل

:إليك كيفية اختلاف وتشابه مجموعات التجزئة وخرائط التجزئة

خريطة تجزئة	مجموعة تجزئة	
كل عنصر عبارة عن زوج من مفتاح وقيمة مع مفتاح فريد وقيمة متصلة به.	.كل عنصر هو مفتاح فريد	التفرد والتخزين
البحث عن معلومات بناءً على مفتاح، مثل البحث عمّن يملك رقم هاتف معيّن.	التحقق مما إذا كان عنصر ما في المجموعة، مثل التحقق مما إذا كان اسم ما في قائمة الضيوف.	حالة الاستخدام
.\(O(1)\ نعم، بمتوسط	.\(O(1)\ نعم، متوسط	هل هو سريع في البحث عن العناصر وإضافتها وحذفها؟
نعم	نعم	هل هناك دالة تجزئة تأخذ المفتاح، وتولد رمز تجزئة، وهذا هو الدلو الذي يتم تخزين العنصر فيه؟

تلخيص جداول التجزئة

.يتم تخزين عناصر جداول التجزئة في حاويات تخزين تسمى الدلاء

.يحتوي كل عنصر في جدول التجزئة على جزء فريد يسمى المفتاح

.تأخذ دالة التجزئة مفتاح العنصر لإنشاء رمز تجزئة

يشير رمز التجزئة إلى الدلو الذي ينتمي إليه العنصر، لذا يمكننا الآن الانتقال مباشرةً إلى عنصر جدول التجزئة هذا: لتعديله، أو لحذفه، .أو للتحقق من وجوده. يتم شرح دوال تجزئة محددة بالتفصيل في الصفحتين التاليتين

يحدث التصادم عندما يكون لعنصرين في جدول التجزئة نفس رمز التجزئة، لأن ذلك يعني أنهما ينتميان إلى نفس الدلو. يمكن حل التصادم بطريقتين.

التسلسل هو الطريقة التي يتم بها حل التصادمات في هذا البرنامج التعليمي، وذلك باستخدام المصفوفات أو القوائم المرتبطة للسماح بوجود أكثر من عنصر في نفس الدلو.

العنونة المفتوحة هي طريقة أخرى لحل التصادمات. باستخدام العنونة المفتوحة، إذا أردنا تخزين عنصر ما ولكن يوجد عنصر بالفعل في ذلك الدلو، يتم تخزين العنصر في الدلو التالي المتاح. يمكن القيام بذلك بعدة طرق مختلفة، لكننا لن نشرح العنونة المفتوحة أكثر من ذلك هنا.

الخاتمة

الجداول التجزئة هي أدوات قوية في البرمجة، تساعدك على إدارة البيانات والوصول إليها بكفاءة.

يعتمد استخدامك لمجموعة تجزئة أو خريطة تجزئة على ما تحتاج إليه: فقط لمعرفة ما إذا كان هناك شيء ما موجود، أو للعثور على معلومات مفصلة عنه.