پر کردن جدول

1 . یکی از روش های هش کردن ، **Linear Hashing** است . لیست اعداد (29 , 64 , 13 , 90 , 11 , 105 , 88) را به روش Linear و با کمک تابع هش زیر در جدولی به اندازه 10 اضافه کنید .

h(k) = k mod 10

2 . یکی دیگر از روش های هش کردن ، **Closed Hashing** است که می توان در آن collision هارا کنترل کرد . کلید های زیر را از چپ به راست به هردو روش زیر در جدولی به اندازه 10 وارد کنید . کدام روش سریع تر است؟ چرا؟

numbers : 102 , 25 , 42 , 71 , 95 , 47 , 33 , 94 , 6 , 12

h1(k) = k mod m

a) linear probing : h(k , i) = (h1(k) + i) mod m

b) quadratic probing : h(k , i) = (h1(k) + i^2) mod m

3 . این دفعه با کمک روش زنجیره ای (Chaining) ، کلید های زیر را در جدول درج کنید :

numbers : 16 , 20 , 39 , 11 , 94 , 23 , 88 , 13 , 44 , 12 , 73

مفهوم رو برسون

تفاوت بین روش‌های **جایگذاری باز (Open Addressing)** و **زنجیره‌ای (Chaining)** برای حل برخورد چیست؟

در چه شرایطی از جداول هش استفاده نمی‌کنیم و چه ساختارهای داده‌ای جایگزینی مناسب‌تر هستند؟

ماتریس گمشده

اطلاعات یک ماتریس را به صورت زیر داشتیم :

- خط اول شامل دو عدد صحیح n , m که ابعاد ماتریس بودند .
- خط بعدی شامل عددهایی که آن ماتریس نگه می دارد .

فردی تمام این اطلاعات را به هم ریخته است . کمک کنید تا ابعاد ماتریس مورد نظر را پیدا کنیم! (حتما از `unordered_map` در الگوریتم خود استفاده کنید)

ورودی

خط اول ورودی تعداد عدد های موجود را نشان می دهد . خط دوم ورودی تمام عددها به صورت به هم ریخته موجودند .

خروجی

ابعاد ماتریس مورد نظر را در خروجی چاپ کنید .

مثال :

نمونه ورودی ۱

```
3
1 2 1
```

خروجی نمونه ۱

```
1 1
```

ورودی نمونه ۲

```
6
2 1 4 5 3 3
```

خروجی نمونه ۲

```
4 1
```

ورودی نمونه ۳

```
8
1 2 6 3 8 5 5 3
```

خروجی نمونه ۳

```
1 6
```

پیدا کن اشتراک را

می دانیم که می توان اشتراک دو آرایه را از روش های مختلفی به دست آورد . در این تمرین از شما خواسته شده با کمک `unordered_set` یا `unordered_map` (ببین کدومش اینجا کاربرد داره!) اشتراک آرایه های داده شده در ورودی را در خروجی چاپ کنید .

ورودی

- خط اول به ترتیب سایز آرایه اول و دوم داده می شود .
- خط دوم عناصر آرایه اول داده می شوند .
- خط سوم عناصر آریه دوم داده می شوند .

خروجی

- عناصری که بین دو آرایه مشترک هستند .

ورودی نمونه :

```
5 7
10 2 3 7 4
2 4 6 10 1 14 9
```

خروجی نمونه :

```
10 2 4
```

جواهر

لیا ، یک جواهرساز با استعداد ، به تازگی یک کیسه پر از سنگ‌های قیمتی و عادی دریافت کرده است . اما همه‌ی این سنگ‌ها ارزشمند نیستند! بعضی از آن‌ها **جواهرات نایاب** هستند و بعضی دیگر فقط سنگ‌های معمولی .

لیا لیستی از **انواع جواهرات خاص** که در بازار ارزش بالایی دارند، دارد و می‌خواهد بفهمد که چند تا از سنگ‌هایی که در کیسه دارد، جواهر هستند .

ورودی :

- یک رشته‌ی `jewels` شامل **انواع جواهرات خاص** . هر کاراکتر این رشته یک نوع جواهر را نشان می‌دهد .
- یک رشته‌ی `stones` شامل **تمام سنگ‌هایی** که در کیسه‌ی لیا قرار دارند .

خروجی :

- یک عدد که نشان می‌دهد چند سنگ در کیسه از نوع جواهرات خاص هستند .

مثال :

ورودی نمونه ۱

```
aA
aAAbbbb
```

خروجی نمونه ۱

```
3
```

▼ توضیحات

لیا دو نوع جواهر **'a'** و **'A'** دارد . در کیسه‌ی او ، این جواهرات ۳ بار ظاهر شده‌اند : دو بار 'A' و یک بار 'a' . بنابراین پاسخ `3` است .

ورودی نمونه ۲

```
z
ZZZ
```

خروجی نمونه ۲

```
0
```

▼ توضیحات

لیا فقط نوع 'z' را به‌عنوان جواهر دارد ، اما در کیسه‌اش فقط 'Z' وجود دارد (کوچک و بزرگ بودن حروف مهم است) . پس هیچ جواهری در بین سنگ‌ها نیست .

محدودیت‌ها :

- `jewels` و `stones` فقط شامل حروف انگلیسی کوچک و بزرگ هستند .
- طول `jewels` و `stones` حداقل ۱ و حداکثر ۵۰ کاراکتر است .
- تمام کاراکترهای `jewels` **منحصربه‌فرد** هستند .

نکته کلیدی : لیا می‌تواند از **یک جدول هش (`unordered_set`)** برای ذخیره انواع جواهراتش استفاده کند و سپس خیلی سریع بررسی کند که هر سنگ در کیسه ، جواهر هست یا نه .

طراحی ست

هدف این تمرین طراحی **یک HashSet سفارشی** است . شما باید یک **ساختار داده‌ای** ایجاد کنید که بتواند مقادیر صحیح را **بدون تکرار** ذخیره کند (مانند `unordered_set<int>` در C++) دقت کنید که استفاده از کلاس های پیشفرض C++ مجاز نیست و باید همه چیز را از ابتدا پیاده سازی کنید .

کلاس شما باید شامل توابع زیر باشد :

- `void add (int key)` : مقدار `key` را به `HashSet` اضافه می‌کند. اگر مقدار از قبل وجود داشته باشد، تغییری اعمال نمی‌شود.
- `void remove (int key)` : مقدار `key` را از `HashSet` حذف می‌کند. اگر مقدار وجود نداشته باشد، تغییری اعمال نمی‌شود.
- `bool contains (int key)` : بررسی می‌کند که آیا مقدار `key` در `HashSet` وجود دارد یا خیر. اگر مقدار موجود باشد، `true` و در غیر این صورت `false` برمی‌گرداند.

نکته : عملیات `add` ، `remove` و `contains` باید در زمان تقریبی `O(1)` انجام شوند .

هش بازی

درباره انواع روش های هش کردن و تابع هش تحقیق کنید . هدف از این تمرین ساخت **unordered_map اختصاصی** شماست . تابع هش خود را بنویسید . تصادم ها (collision) را با روش دلخواه خود کنترل کنید و در نهایت توابع زیر را برای unordered_map اختصاصی خودتان پیاده سازی کنید .

1.تابع insert : این تابع با گرفتن کلید و مقدار (value) ، در صورت وجود نداشتن کلید دریافت شده ، آن را به همراه مقدارش اضافه میکند .

2. تابع find : وظیفه این تابع گرفتن یک کلید و سرچ کردن آن کلید در ساختمان داده شماست . اگر که کلید را پیدا کرد، مقدار آن را برمیگرداند و در غیر اینصورت ، exception با عنوان key_error را throw میکند .