# Element

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت

---

یک آرایه به طول $n$ از اعداد صحیح به نام $a$ به شما داده شده است. حال باید به $q$ پرسش پاسخ دهید. در پرسش $i$م عدد $x_i$ داده می‌شود. شما باید بگویید به ترتیب چند عدد در آرایه از $x_i$ کوچک‌تر هستند، چند عدد با آن برابرند و چند عدد از آن بزرگ‌تر هستند.

## ورودی

خط اول ورودی تنها شامل عدد $n$ طول آرایه است. در خط دوم $n$ عدد $a_i$ که با فاصله از هم جدا شده‌اند آمده است. در خط سوم ورودی تنها عدد $q$ تعداد پرسش‌ها می‌آید. در $q$ خط بعدی، در خط $i$م تنها عدد $x_i$ آمده است.

$$1 \leq n \leq 10^6$$

$$1 \leq a_i \leq 10^6$$

$$1 \leq q \leq 10^4$$

$$1 \leq x_i \leq 10^6$$

## خروجی

خروجی برنامه باید شامل $q$ خط باشد که در خط $i$م آن سه عدد پاسخ پرسش $i$ با فاصله از هم آمده‌اند، به ترتیب از چپ به راست چند عدد در آرایه از $x_i$ کوچک‌تر هستند، چند عدد با آن برابرند و چند عدد از آن بزرگ‌تر هستند.

# مثال

## ورودی نمونه

```
7
3 5 2 5 1 10 5
5
5
2
1
10
100
```

## خروجی نمونه

```
3 3 1
1 1 5
0 1 6
6 1 0
7 0 0
```

پس از مرتب سازی به آرایه زیر می رسیم :

```
[1,2,3,5,5,5,10]
```

برای چهار درخواست اول جواب واضح است و مقدار داده شده در درخواست آخر در آرایه نیست و از همه‌ی ۷ عنصر آرایه بزرگتر است.

# Dior

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت

---

یک آرایه به طول $n$ از اعداد صحیح متمایز به نام $a$ داریم.

این آرایه را با الگوریتم مرتب‌سازی سریع (*Quick Sort*) از کوچک به بزرگ مرتب کرده‌ایم و ترتیب انتخاب $pivot$ها در الگوریتمی که ما پیاده کرده‌ایم به شرح زیر به شما داده شده است:

```
pivots_list = an empty list
QuickSort(A, p, r)
    if p < r
        q <- partition(A, p, r)
        pivots_list.append(A[q])
        QuickSort(A, p, q-1)
        QuickSort(A, q+1, r)
    if p == r
        pivots_list.append(A[p])
```

ترتیب انتخاب $pivot$ها همان ترتیب موجود در لیست `pivots_list` است.

حال فرض کنید آرایه $a$ با ترتیب `pivots_list` که در ورودی به شما داده می‌شود با الگوریتم مرتب‌سازی سریع مرتب شده است. چند مقایسه در کل اجرای الگوریتم صورت گرفته است؟

فرض کنید وقتی عدد `A[q]` در یک قسمت از آرایه به طول $r - p + 1$ به عنوان $pivot$ انتخاب می‌شود باید با همه‌ی اعداد دیگر آن قسمت مقایسه شود؛ یعنی به تعداد مقایسه‌ها $r - p$ واحد افزوده می‌شود.

## ورودی

در خط اول ورودی به شما عدد صحیح $n$، طول آرایه داده شده است.

در خط دوم $n$ عدد صحیح متمایز که با فاصله از یک‌دیگر جدا شده‌اند آمده، که اعداد آرایه‌ی $a$ را نشان می‌دهد.

در خط سوم و پایانی ورودی $n$ عدد صحیح که با فاصله از یک‌دیگر جدا شده‌اند آمده، که ترتیب انتخاب $pivot$ها ( `pivots_list` ) را نشان می‌دهد. این اعدد در واقع ترتیبی دیگر از اعداد آرایه‌ی $a$ هستند.

$$1 \leq n \leq 10^5$$

$$1 \leq a_i \leq 10^9$$

## خروجی

در تنها خط خروجی، یک عدد تعداد مقایسه‌های صورت گرفته را چاپ کنید.

## مثال

### ورودی نمونه ۱

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

### خروجی نمونه ۱

```
8
```

### ورودی نمونه ۲

```
5
1 27 324 415 666
666 415 324 27 1
```

### خروجی نمونه ۲

```
10
```

# Invincible

- محدودیت زمان: ۱ ثانیه
- محدودیت حافظه: ۲۵۶ مگابایت

---

فرض کنید می‌خواهیم به روی آرایه $a$ به طول $n$ که اعداد آن بین ۰ تا $m-1$ هستند مرتب‌سازی سطلی انجام دهیم (*Bucket Sort*). مراحل مرتب سازی به صورت زیر است:

- فرض کنید طول هر سطل برابر با $k$ است، یعنی سطل اول بازه‌ی $[0, k)$ را پوشش می‌دهد، سطل دوم بازه‌ی $[k, 2k)$ را پوشش می‌هد و ...
- هر عدد آرایه را در سطلی که بازه‌ی آن عدد را پوشش می‌دهد، قرار می‌دهیم (مثلا اگه $a_2$ برابر با $3$ باشد و سطل دوم بازه‌ی $[2, 4)$ را پوشش دهد، $a_2$ در سطل دوم قرار می‌گیرد.)
- سپس هر سطل را با مرتب سازی درجی (*Insertion Sort*) مرتب می‌کنیم.

با کمی ساده سازی می‌توان فرض کرد که هزینه این مرتب‌سازی برابر با عبارت زیر است:

$$\sum_{i=1}^{m/k} size_i^2$$

که $size_i$ برابر است با تعداد اعدادی از آرایه $a$ که در سطل $i$ام قرار گرفته‌اند.

فرض کنید می‌خواهیم یک عدد از آرایه را حذف کنیم به طوری که هزینه مرتب‌سازی آرایه حاصل (با روش توضیح داده شده) بیشترین مقدار نسبت به حالت قبل حذف کاهش یابد، کدام عدد را باید حذف کنیم؟

## ورودی

در خط اول ورودی سه عدد $n$ و $m$ و $k$ با فاصله از هم آمده است. در خط دوم $n$ عدد که اعداد آرایه $a$ هستند با فاصله از هم آمده‌اند.

$$2 \leq n \leq 10^5$$

$$2 \leq m \leq 2 \times 10^5$$

$$0 \leq a_i < m$$

$$1 \leq k \leq m$$

- تضمین می‌شود که $m$ بر $k$ بخش‌پذیر است.

## خروجی

در تنها خط خروجی عددی که حذف آن باعث بیشترین مقدار کاهش هزینه مرتب‌سازی می‌شود را چاپ کنید (اگر چند جواب وجود داشت یکی را به دلخواه چاپ کنید).

## مثال

### ورودی نمونه ۱

```
5 6 3
5 2 2 2 5
```

### خروجی نمونه ۱

```
2
```

بعد از اجرای *Bucket Sort* دو سطل ایجاد می‌شود.

سطل اول $[0, 3)$: $[2, 2, 2]$

سطل دوم $[3, 6)$: $[5, 5]$

برای کمینه کردن هزینه یک عنصر باید از سطل اول حذف کنیم.

## ورودی نمونه ۲

```
10 12 4
2 7 1 0 2 1 2 9 7 9
```

## خروجی نمونه ۲

```
2
```

بعد از اجرای *Bucket Sort* سه سطل ایجاد می‌شود.

سطل اول $[0, 4)$: $[0, 1, 1, 2, 2, 2]$

سطل دوم $[4, 8)$: $[7, 7]$

سطل سوم $[8, 12]$: $[9, 9]$

برای کمینه کردن هزینه یک عنصر باید از سطل اول حذف کنیم.

# تشریحی اول

الف)یک آرایه به طول $n$ داریم که عناصر آن در بازه $[1, n^2]$ قرار گرفته‌اند.الگوریتمی ارائه دهید که بتواند این آرایه را در زمان خطی ($O(n)$) مرتب کند.

ب)یک آرایه تشکیل شده از ۰ ، ۱ و ۲ داریم. الگوریتمی ارائه دهید که بتواند در زمان $O(n)$ این آرایه را مرتب کند.

# تشریحی دوم

الگوریتم *quick sort* را در نظر بگیرید.مشخص کنید به ازای ورودی زیر در هر مرحله از فراخوانی partition ، آرایه به چه صورت تجزیه می شود.پاسخ باید شامل تکه های تجزیه شده و ترتیب قرارگیری عناصر در آن تکه از آرایه باشد.

```
input : 24, 18, 30, 22, 18, 15, 10, 22
```

# تشریحی سوم

فرض کنید دو آرایه $A$ و $B$ از اعداد طبیعی بین $[1, n^4]$ داریم.الگوریتمی ارائه دهید که در زمان خطی چک کند که آیا وجود دارد $a \in A$ ، $b \in B$ و $x \in \mathbb{N}$ که $x$ همان ورودی مسئله است که شرط $x = a + b$ را برآورده کند.