تغییرات آرایوی

- محدودیت حافظه: ۲۵۶ مگابایت

در این سؤال می‌خواهیم یک *database* ساده به‌وسیله‌ی آرایه بسازیم. فرض کنید همه‌ی اطلاعاتمان به صورت عدد هستند که در آرایه‌ای قرار می‌گیرند. می‌توانیم عددی به آرایه اضافه کنیم، یا اینکه عددی از آن حذف کنیم. به صورت دقیق‌تر فرض کنید اعداد$a_1, a_2, ..., a_n$ در آرایه قرار دارند در هر مرحله یا عددی مثل x را در جایگاه i اضافه می‌کنیم یا اینکه عدد موجود در جایگاه iام را حذف می‌کنیم. با اضافه کردن یک عدد جدید در جایگاه i تمامی اعدادی که قبلاً در جایگاه‌های i تا n بودند یک واحد به جلو می‌روند. برای مثال فرض کنید آرایه در ابتدا به صورت زیر است:

$$< 1, 10, 3, 5 >$$

اگر عدد 100 را به جایگاه دوم در این آرایه اضافه کنیم آرایه به صورت زیر خواهد شد:

$$< 1, 100, 10, 3, 5 >$$

در واقع اعداد ۱۰ و ۳ و ۵ یک واحد به جلو خواهند رفت.

همچنین اگر عدد جایگاه دوم را حذف کنیم آرایه به صورت زیر خواهد شد:

$$< 1, 3, 5 >$$

و اعداد ۳ و ۵ که پس از ۱۰ بودند یک واحد عقب‌تر می‌آیند (در ابتدا در جایگاه‌های ۳ام و ۴ام بودند ولی هم اکنون در جایگاه‌های ۲ام و ۳ام اند).

ورودی

در خط اول ورودی به شما عدد qداده می‌شود. که این عدد بیانگر تعداد عملیات حذف یا اضافه‌کردن‌ای است که قرار است روی آرایه انجام شود.

سپس در q خط بعدی در هر خط ورودی به یکی از دو صورت زیر خواهد بود:

- `+ i x`
- `- i`

که حالت اول بیانگر این است که عدد x قرار است در جایگاه i اضافه شود و حالت دوم بیانگر این است که عددی که در جایگاه iبوده قرار است حذف شود.

$$1 \leq q \leq 100$$

همه‌ی اعداد در ورودی حداکثر 10^9 هستند. همچنین تضمین می‌شود که هیچ‌وقت عنصر به جایگاهی بزرگتر از آرایه‌ی فعلی اضافه نشده و یا از آن حذف نمی‌شود.

در ابتدا آرایه **خالی** است.

خروجی

پس از هر عملیات اعداد آرایه را به ترتیب با فاصله خروجی دهید توجه کنید اگر آرایه خالی بود `EMPTY` خروجی دهید.

مثال

ورودی نمونه ۱

```
7
+ 1 1
+ 2 3
+ 2 10
```

```
+ 4 5
+ 2 100
- 2
- 2
```

خروجی نمونه ۱

```
1
1 3
1 10 3
1 10 3 5
1 100 10 3 5
1 10 3 5
1 3 5
```

توجه کنید اگر طول آرایه برابر len باشد و ما بخواهیم عنصری به آخر آن اضافه کنیم می‌نویسیم:

```
+ len+1 x
```

و این‌گونه عنصری به آخر اضافه می‌شود. همانطور که می‌بینید خط‌های

- `+ 1 1`
- `+ 2 3`
- `+ 4 5`

این‌گونه اند.

ورودی نمونه ۲

```
10
+ 1 1
+ 2 2
+ 3 3
```

```
+ 4 4
+ 5 5
- 1
- 3
- 2
- 2
- 1
```

خروجی نمونه ۲

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

توجه کنید که خط آخر خالی است یعنی دیگر عضوی باقی‌نمانده است.

ورودی نمونه ۳

```
3
+ 1 1
- 1
+ 1 2
```

خروجی نمونه ۳

```
1
EMPTY
```

2

نکته: در زدن کد این سؤال حواستان باشد که در زبان‌های برنامه‌نویسی اندیس‌ها به جای اینکه از ۱ شروع شوند از ۰ شروع می‌شوند و عنصر اول ۰ و عنصر آخر $n-1$ است. همچنین می‌توانید در پیاده‌سازی آرایه‌ای بزرگ به اندازه‌ی ۱۰۰ در نظر بگیرید و n متغیری باشد که طول مورد استفاده از آرایه را عوض می‌کند. به عبارتی لازم نیست تعداد خانه‌های حافظه‌ی مورد استفاده را دستکاری کنید.

▼ راهنمایی برای پیاده‌سازی

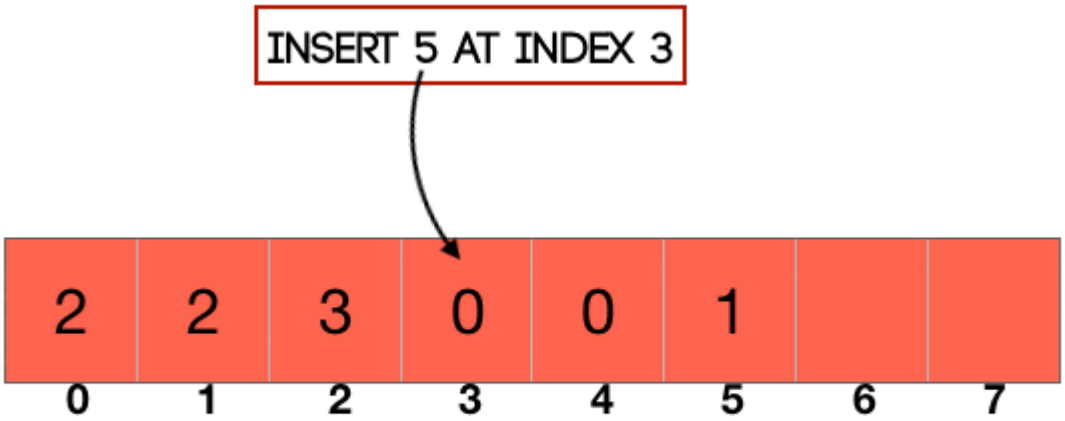
فرض کنید یک عنصر را حذف می‌کنیم در اینصورت باید عناصر بعد از آن در آرایه یک واحد عقب بروند. به این حرکت *shift* دادن می‌گویند. شبه‌کد زیر اعداد بعد از جایگاه $index$ را یک واحد عقب می‌برد و طول آرایه را یک واحد کم می‌کند. با این‌کار عملاً آرایه عنصر $index$ام اش را از دست می‌دهد.

```
> remove_and_shift_array
a[] : an array containing our data
n : the length of the array
index : the index of the place we want to remove
-----------------------------------------------
1. for i from index + 1 to n
2.     swap(a[i], a[i-1])
3. n <- n - 1
```

همچنین برای اضافه کردن یک عنصر باید ابتدا اندازه‌ی آرایه را یک واحد بزرگ کنیم و سپس همه‌ی عناصر را یک واحد جلو ببریم. این عدد قرار است در جایگاه $index$ قرار بگیرد و مقدارش برابر x شود.

```
> add_element_to_array
a[] : an array containing our data
n : the length of the array
index : the index of the place a new number is supposed to be added
x : the value of the number that is supposed to be added
-----------------------------------------------------------------
1. n <- n + 1
2. for i from n to index + 1:
```

```
3.      swap(a[i], a[i-1])
4. a[index] <- x
```



در شبه‌کد اولی پیمایش از ابتدا تا به انتهاست و در دومی از انتها به ابتدا.