[Vector]. Bài 1. Push Pop

Cho vector và 2 thao tác.

Thao tác 1 : Thêm 1 phần tử và cuối vector.

Thao tác 2 : Xóa phần tử khỏi cuối vector nếu vector không rỗng.

Nhiệm vụ của bạn là thực hiện 1 loạt các thao tác này và in ra mảng sau khi thực hiện xong mọi thao tác. Nếu vector rỗng in ra EMPTY, ngược lại in ra các phần tử trong vector trên 1 dòng.

Input Format

Dòng 1 là N : số thao tác được thực hiện. N dòng tiếp theo mỗi dòng mô tả thao tác, nếu thao tác là 1 sẽ có thêm phần tử được thêm vào cuối.

Constraints

1<=N, M<=200; Các phần tử thêm vào vector là số nguyên int 32bit.

Output Format

In ra EMPTY nếu vector rỗng, ngược lại in ra các phần tử trong vector trên 1 dòng.

Sample Input 0

```
7
1 58
2
2
1 52
1 81
1 12
1 2
```

Sample Output 0

```
52 81 12 2
```

[Vector]. Bài 2. Erase Insert

Cho vector và 2 thao tác.

- Thao tác 1 : Chèn phần tử vào vị trí bất kì trong vector.
- Thao tác 2 : Xóa phần tử ở vị trí bất kì trong vector.

- Đối với thao tác thứ 1, giả sử vector đang có N phần tử, chỉ số chèn hợp lệ sẽ là từ 0 tới N, ngoài ra các vị trí không hợp lệ sẽ không thực hiện chèn. Tương tự đối với thao tác thứ 2, giả sử vector đang có N phần tử thì chỉ số xóa hợp lệ sẽ là từ 0 tới N - 1, nếu vị trí xóa không hợp lệ hoặc vector sẽ không thực hiện xóa.

Nhiệm vụ của bạn là thực hiện 1 loạt các thao tác này và in ra mảng sau khi thực hiện xong mọi thao tác. Nếu vector rỗng in ra EMPTY, ngược lại in ra các phần tử trong vector trên 1 dòng.

Input Format

- Dòng 1 là M : số lượng phần tử trong vector.
- Dòng 2 là M số trong vecotor.
- Dòng 3 là N : số thao tác được thực hiện. N dòng tiếp theo mỗi dòng mô tả thao tác, nếu thao tác là 1 sẽ có thêm phần tử được thêm vào cuối.

Constraints

1<=N<=200; Các phần tử thêm vào vector là số nguyên int 32bit.

Output Format

In ra EMPTY nếu vector rỗng, ngược lại in ra các phần tử trong vector trên 1 dòng.

Sample Input 0

```
5
40 87 73 47 22
5
1 3 21
2 3
2 0
2 1
1 0 70
```

Sample Output 0

```
70 87 47 22
```

[Vector]. Bài 3. Sắp xếp

Cho vector V có N phần tử, nhiệm vụ của bạn là sắp xếp các phần tử trong vector theo thứ tự tăng dần, giảm dần sau đó in ra bằng cách dùng iterator.

Để sắp xếp vector bạn dùng hàm sort trong thư viện

```
#include <algorithm>

//Tăng dần
sort(v.begin(), v.end());
```

```
//Giảm dần
sort(v.begin(), v.end(), greater<int>());
```

Input Format

Dòng 1 là N : số lượng phần tử trong vector. Dòng 2 là N số trong vector.

Constraints

1<=N<=1000. Các phần tử của vector là số nguyên 32bit.

Output Format

Dòng 1 in ra vector tăng dần, dòng 2 in ra vector giảm dần.

Sample Input 0

```
8
992 763 670 344 67 268 298 852
```

Sample Output 0

```
67 268 298 344 670 763 852 992
992 852 763 670 344 298 268 67
```

[Vector]. Bài 4. Duyệt

Cho vector V có N phần tử, nhiệm vụ của bạn in ra các phần tử từ chỉ số L tới chỉ số R sau đó in ra các phần tử từ chỉ số R tới chỉ số L bằng cách dùng iterator.

Input Format

Dòng 1 là N : số lượng phần tử trong vector. Dòng 2 là N số trong vector. Dòng 3 là L và R

Constraints

1<=N<=1000. 0<=L<=R

Output Format

Dòng 1 in ra vector từ L tới R, dòng 2 in ra vector từ R tới L.

Sample Input 0

```
10
8 1 8 3 8 5 3 5 9 7
3 6
```

Sample Output 0

```
3 8 5 3
3 5 8 3
```

[Vector]. Bài 5. Lật ngược vector

Cho vector V có N phần tử, nhiệm vụ của bạn là lật ngược vector V và in ra. Sau khi lật ngược toàn bộ vector, bạn tiếp tục lật ngược các phần tử từ chỉ số L tới chỉ số R sau đó in ra vector. Để lật ngược vector V :

```
reverse(V.begin(), V.end())
```

, để lật ngược vector V từ chỉ số L tới chỉ số R :

```
reverse(V.begin() + L, V.begin() + R + 1);
```

Input Format

Dòng 1 là N : số lượng phần tử trong vector. Dòng 2 là N số trong vector. Dòng 3 là L và R

Constraints

1<=N<=1000. 0<=L<=R

Output Format

Dòng 1 in ra vector sau lần lật 1, dòng 2 in ra vector sau lận lật 2.

Sample Input 0

```
13
5 3 4 1 6 3 0 3 1 4 8 4 1
3 3
```

Sample Output 0

```
1 4 8 4 1 3 0 3 6 1 4 3 5
1 4 8 4 1 3 0 3 6 1 4 3 5
```

Sample Input 1

```
10
7 8 1 7 1 9 1 4 0 9
3 5
```

Sample Output 1

```
9 0 4 1 9 1 7 1 8 7
9 0 4 1 9 1 7 1 8 7
```

[Vector]. Bài 6. max_element, min_element, accumulate

Với vector V bạn có thể dùng hàm max_element để tìm phần tử lớn nhất, min_element để tìm phần tử nhỏ nhất, accumulate để tính tổng. Cú pháp (Đối với mảng các bạn dùng (a, a + n)) :

```
- cout << *min_element(V.begin(), V.end());
- cout << *max_element(V.begin(), V.end());
- cout << accumulate(V.begin(), V.end(), 0);
```

Input Format

Dòng 1 là N : số lượng phần tử trong vector. Dòng 2 là N số trong vector.

Constraints

1<=N<=1000. Các phần tử của vector là số nguyên 32bit.

Output Format

Dòng 1 in ra phần tử nhỏ nhất, dòng 2 in ra phần tử lớn nhất, dòng 3 in ra tổng các phần tử.

Sample Input 0

```
10
2 2 2 7 6 6 6 9 5 7
```

Sample Output 0

```
2
9
52
```

[Vector]. Bài 7. Vector và pair

Cho N điểm trong hệ tọa độ Oxy, bạn hãy dùng vector

```
pair<int, int>
```

để lưu tọa độ các điểm này. Sau đó duyệt vector và tính khoảng cách từ các điểm này về gốc tọa độ và lưu vào 1 vector sau đó in ra các phần tử trong vector khoảng cách này lấy 2 số sau dấu phẩy.

Input Format

Dòng 1 là N : số lượng điểm. N dòng tiếp theo mỗi dòng gồm 2 số là tung độ và hoành độ.

Constraints

1<=N<=1000; Tọa độ là số nguyên có trị tuyệt đối không quá 100;

Output Format

In ra đáp án của bài toán.

Sample Input 0

```
13
5 27
69 84
92 51
12 70
26 70
76 47
36 94
55 2
26 20
85 97
17 31
33 88
```

```
10 75
```

Sample Output 0

```
27.46 108.71 105.19 71.02 74.67 89.36 100.66 55.04 32.80 128.97 35.36 93.98 75.66
```

[Vector]. Bài 8. Vector và pair 2

Cho N điểm trong hệ tọa độ Oxyz, bạn hãy dùng vector

```
pair<pair<int, int>, int>>
```

để lưu tọa độ các điểm này. Sau đó duyệt vector và in ra tổng của tung độ, hoành độ, cao độ.

Input Format

Dòng 1 là N : số lượng điểm. N dòng tiếp theo mỗi dòng gồm 3 số là tung độ, hoành độ, cao độ.

Constraints

1<=N<=1000; Tọa độ là số nguyên có trị tuyệt đối không quá 100;

Output Format

In ra đáp án của bài toán

Sample Input 0

```
12
65 91 53
64 70 15
50 9 57
69 37 11
31 35 66
73 55 50
63 40 38
33 5 41
67 39 29
85 78 6
67 49 83
41 34 88
```

Sample Output 0

```
209 149 116 117 132 178 141 79 135 169 199 163
```

[Vector]. Bài 9. Đếm tần suất 1

Sử dụng vector và pair để giải quyết bài toán sau. Cho mảng A[] gồm N phần tử, bạn hãy đếm xem mỗi phần tử trong mảng A[] xuất hiện bao nhiêu lần và in ra theo thứ tự xuất hiện trong mảng A[].

Hướng dẫn : Các bạn sử dụng 1 vector rỗng lưu pair trong đó first của pair lưu giá trị và second lưu tần suất, mỗi khi gặp 1 phần tử trong mảng A[] bạn hãy duyệt vector đã có và kiểm tra xem giá trị này đã xuất

hiện chưa, nếu đã xuất hiện bạn tăng second của nó lên còn nếu chưa xuất hiện thì bạn push_back 1 cặp pair mới vào vector, pair này sẽ lưu giá trị bạn đang xét và tần suất là 1.

Input Format

- Dòng 1 : N
- Dòng 2 là các phần tử trong mảng A[] viết cách nhau một dấu cách

Constraints

- 1<=N<=10^4
- -1000<=A[i]<=1000

Output Format

In ra các giá trị xuất hiện trong mảng và tần suất tương ứng.

Sample Input 0

```
14
4 3 6 3 1 1 9 6 9 8 1 3 6 7
```

Sample Output 0

```
4 1
3 3
6 3
1 3
9 2
8 1
7 1
```

[Vector]. Bài 10. Đếm tần suất 2

Sử dụng vector và pair để giải quyết bài toán sau. Cho mảng A[] gồm N kí tự, bạn hãy đếm xem mỗi phần tử trong mảng A[] xuất hiện bao nhiêu lần và in ra theo thứ tự xuất hiện trong mảng A[].

Hướng dẫn : Các bạn sử dụng 1 vector rỗng lưu pair trong đó first của pair lưu giá trị và second lưu tần suất, mỗi khi gặp 1 phần tử trong mảng A[] bạn hãy duyệt vector đã có và kiểm tra xem giá trị này đã xuất hiện chưa, nếu đã xuất hiện bạn tăng second của nó lên còn nếu chưa xuất hiện thì bạn push_back 1 cặp pair mới vào vector, pair này sẽ lưu giá trị bạn đang xét và tần suất là 1.

Input Format

- Dòng 1 : N
- Dòng 2 là các phần tử trong mảng A[] viết cách nhau một dấu cách

Constraints

- 1<=N<=10^4
- Các phần tử của mảng A[] là chữ cái in thường

Output Format

- Dòng 1 in ra số lượng kí tự khác nhau trong mảng
- Các dòng tiếp theo in ra kí tự và tần suất tương ứng

Sample Input 0

```
11
e g e h j e c g g g d
```

Sample Output 0

```
6
e 3
g 4
h 1
j 1
c 1
d 1
```