**Nicholas and Permutation**

**Link submit:** <http://codeforces.com/problemset/problem/676/A>

**Solution:**

|  |  |
| --- | --- |
| C++ | <https://ideone.com/xjkEHf> |
| Java | <https://ideone.com/jo07pF> |
| Python | <https://ideone.com/xqtycn> |

**Tóm tắt đề:**

Cho bạn mảng gồm n số nguyên có giá trị phân biệt từ 1 đến n.

Cho phép bạn thực hiện hoán đổi một lần duy nhất vị trí của số nhỏ nhất (1) hoặc số lớn nhất (n) với bất kỳ số nào khác trong mảng. Nhiệm vụ của bạn là tính xem khoảng cách xa nhất có thể đạt được giữa hai số này là bao nhiêu.

**Input:**

Dòng đầu tiên chứa một số nguyên *n* (2 ≤ n ≤ 100) – số lượng phần tử trong mảng.

Dòng tiếp theo chứa n số nguyên phân biệt *ai* (1 ≤ ai ≤ n) là phần tử thứ i của mảng.

**Output:**

In ra một số duy nhất là khoảng cách xa nhất có thể giữa hai số 1 và n.

**Ví dụ:**

|  |  |
| --- | --- |
| 5 4 5 1 3 2 | 3 |

|  |  |
| --- | --- |
| 7 1 6 5 3 4 7 2 | 6 |

|  |  |
| --- | --- |
| 6 6 5 4 3 2 1 | 5 |

**Giải thích ví dụ:** Quy ước mảng đánh dấu bắt đầu từ 0:

***Ví dụ 1:*** Cách tối ưu khoảng cách giữa 1 và 5 là hoán đổi 1 với số cuối cùng của mảng là 2. Lúc này vị trí tương đối trong mảng của 5 sẽ là 1 và của 1 là 4. Như vậy khoảng cách giữa tối đa giữa hai số này là |1 - 4| = 3.

***Ví dụ 2:*** Cách tối ưu khoảng cách giữa 1 và 7 là hoán đổi 7 với số cuối cùng của mảng là 2. Lúc này vị trí tương đối trong mảng của 1 sẽ là 0 và của 7 là 6. Như vậy khoảng cách tối đa giữa hai số này là |0 - 6| = 6.

***Ví dụ 3:*** Khoảng cách giữa 1 và 6 đã được tối ưu và bằng |0 - 5| = 5.

**Hướng dẫn giải:**

Nhận xét: Để tối ưu khoảng cách, cách tốt nhất là ta hoán vị một trong hai số về đầu mảng hoặc cuối mảng. Như vậy, với mảng đánh dấu bắt đầu từ 0, gọi pos\_1 và pos\_n lần lượt là vị trí của số 1 và n trong mảng. Xét các trường hợp sau:

* Trường hợp 1: Dời 1 về đầu mảng, lúc này vị trí tương đối của 1 là pos\_1 = 0:
  + Khoảng cách d1 = pos\_n
* Trường hợp 2: Dời 1 về cuối mảng, lúc này vị trí tương đối của 1 là pos\_1 = n - 1:
  + Khoảng cách d2 = n - 1 - pos\_n
* Trường hợp 3: Dời n về đầu mảng, tức pos\_n = 0:
  + Khoảng cách d3 = pos\_1
* Trường hợp 4: Dời n về cuối mảng, tức pos\_n = n - 1:
  + Khoảng cách d4 = n - 1 - pos\_1

Khoảng cách lớn nhất trong 4 khoảng cách trên chính là đáp án cần tìm.

**Độ phức tạp:** **O(n)** với n là kích thước của mảng.