1. Maximum Knowledge

There is going to be a conference for d days. The schedule of n meetings is given as three arrays, s (start), e (end), and a (additional knowledge gained). The lth meeting is available from [s[i], e[i]] days (both inclusive) and provides a knowledge gain of a[i]. A limit of k meetings can be attended in one day.

Find the maximum knowledge that can be gained in a day.

Example

$$d = 10$$

$$n = 4$$

$$k=2$$

$$e = [8, 9, 7, 5]$$



22. Question 22

A shopkeeper in HackLand assigns each item in a shop a unique popularity rating. To order the items in decreasing popularity from left to right, the shopkeeper can swap any 2 items in one operation. Determine the minimum number of operations needed to reorder the items correctly.

Example

First switch 3 and 4 to get popularity' = [4, 3, 1, 21.

Then switch 1 and 2 to get [4, 3, 2, 1]. The array is reordered in 2 operations.

Function Description

Complete the function minimumSwaps in the editor below.

minimumSwaps has the following parameter(s): int popularity[n]: an array of integers that represents the popularity of each item Returns:

int: the minimum number of swaps to order the items properly

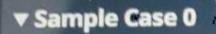
Constraints

• 1≤n≤2×10⁵

Constraints

- · 1 ≤ n ≤ 2 × 105
- 1 ≤ popularity[i] ≤ n

▶ Input Format for Custom Testing



Sample Input 0

Sample Output 0

1

Explanation 0n = 3 popularity= [3, 1, 2]

Switch 1 and 2 and the items in the array [3, 2, 1] are reordered in 1 operation. The return value is 1.

```
1.
             1st one
int Find(int n, int k,int d, vector<int> s, vector<int> e){
    vector<vector<int>> v;
    for(int i=0;i<n;i++) v.push_back({s[i],e[i]});
    sort(v.begin(),v.end());
    priority_queue<int,vector<int>,greater<int>> pq;
    int i=0,day=v[0][0];
    int ans=0:
    while(i<n or pq.size()){
      if(pq.size()==0) day=v[i][0];
      while(i<n and v[i][0]<=day){
        pq.push(v[i][1]):
        i++;
      }
     int cnt=0;
     while(pq.size() and cnt<k){
        pq.pop();
        cnt++;
       ans++;
     }
     day++;
  if(day>d) break;
     while(pq.size() and pq.top()<day) pq.pop();
  }
  return ans;
}
```

```
2<sup>nd</sup>
int minSwaps(int arr[], int n)
{
  int len = n;
  map<int, int> map;
  for (int i = 0; i < len; i++)
     map[nums[i]] = i;
  sort(nums, nums + n);
  reverse(nums,nums+n);
  bool visited[len] = { 0 };
  int ans = 0:
  for (int i = 0; i < len; i++) {
     if (visited[i] || map[nums[i]] == i)
       continue:
     int j = i, cycle_size = 0;
     while (!visited[j]) {
       visited[j] = true;
       j = map[nums[j]];
       cycle_size++;
     }
     if (cycle_size > 0) {
       ans += (cycle_size - 1);
  return ans;
}
```