

20101576 정재찬.

제가 한 정렬 방법은 분할-정복방법과 pivot을 이용한 정렬입니다.

이 정렬 방법을 선택한 이유 : pivot을 분할된 배열의 맨 오른쪽으로 함으로써 코드가 더 간결해지는 효과가 있다고 생각했기 때문입니다.

소스코드 및 설명 :

```
4주차 정렬 (전역 범위)
1  #include <iostream>
2
3
4  void swap(int* a, int* b) // 두 배열에 저장된 값을 서로 바꾸기위한 함수
5  {
6      int temp = *a;
7      *a = *b;
8      *b = temp;
9  }
10
11 int partition(int arr[], int left, int right)
12 {
13     int pivot = arr[right]; // 나뉘어진 영역의 맨 오른쪽 값을 pivot으로 둔다.
14     int i = left - 1; // i는 마지막에 pivot이 있어야 할 위치를 나타내기 위한 변수.
15     for (int j = left; j <= right - 1; j++) // 맨 오른쪽이 pivot이기 때문에 right-1까지 반복
16     {
17         if (arr[j] <= pivot) // 피벗보다 작으면
18         {
19             i++;
20             swap(&arr[i], &arr[j]); // 배열의 맨 앞에서부터 줄지어서 정렬
21         }
22     }
23     swap(&arr[i + 1], &arr[right]); // 피벗보다 작은것들을 앞에서 세웠으니 그 다음에 pivot을 둬.
24     return (i + 1); // pivot의 위치는 정해졌고, pivot 전과 후는 pivot보다 작고, 큰값만 남음.
25 }
26
27
28 void quickSort(int arr[], int left, int right)
29 {
30     if (left < right)
31     {
32         int next_pivot = partition(arr, left, right); // 다음 pivot값을 알아내고
33
34         quickSort(arr, left, next_pivot - 1); // pivot을 기준으로 이전
35         quickSort(arr, next_pivot + 1, right); // pivot을 기준으로 이후를 다시 quickSort함.
36     }
37 }
38
39 int main()
40 {
41     int N;
42     std::cin >> N;
43     int* arr = new int[N];
44     for (int i = 0; i < N; i++)
45         std::cin >> arr[i];
46
47     quickSort(arr, 0, N-1);
48     for(int i=0;i<N;i++)
49         std::cout << arr[i] << '\n';
50 }
```

결과 :

CIS Online Judge							20101576-정재찬
Home Problems Contests Status Rank About							
Status							
Status All 20101576 Select							
When	ID	Status	Problem	Time	Memory	Language	Author
2023-9-26 16:23:36	19d240632743	Accepted	1005	2ms	3MB	C++	20101576-정재찬
2023-9-26 16:15:01	e7dfccdb87564	Accepted	1005	2ms	3MB	C++	20101576-정재찬