DA-5

```
#include <iostream>
#include <vector>
#include <cstdlib>

using namespace std;

// Partition using the last element as the pivot
int deterministicPartition(vector<int>& arr, int low, int high) {
    int pivot = arr[high];
    int i = low - 1;

    for (int j = low; j <= high - 1; j++) {
        if (arr[j] < pivot) {
            i++;
            swap(arr[i], arr[j]);
        }
    }
    swap(arr[i + 1], arr[high]);
    return (i + 1);
}

// Randomized partition
int randomizedPartition(vector<int>& arr, int low, int high) {
    srand(time(NULL));
    int random = low + rand() % (high - low);
    swap(arr[random], arr[high]);
    return deterministicPartition(arr, low, high);
}

void deterministicQuickSort(vector<int>& arr, int low, int high) {
    if (low < high) {
        int pi = deterministicPartition(arr, low, high);
        deterministicQuickSort(arr, low, pi - 1);
        deterministicQuickSort(arr, pi + 1, high);
    }
}

void randomizedQuickSort(vector<int>& arr, int low, int high) {
    if (low < high) {
        int pi = randomizedPartition(arr, low, high);
        randomizedQuickSort(arr, low, pi - 1);
        randomizedQuickSort(arr, pi + 1, high);
    }
}

int main() {
    vector<int> arr = {10, 7, 8, 9, 1, 5};
```

```
    int n = arr.size();
    deterministicQuickSort(arr, 0, n - 1);
    cout << "Sorted array using deterministic quicksort: \n";
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;

    arr = {10, 7, 8, 9, 1, 5};
    randomizedQuickSort(arr, 0, n - 1);
    cout << "Sorted array using randomized quicksort: \n";
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;

    return 0;
}
```

Output

```
/tmp/4i8vwxOk9J.o
Sorted array using deterministic quicksort:
1 5 7 8 9 10
Sorted array using randomized quicksort:
1 5 7 8 9 10
```