Lab Assignment - 06

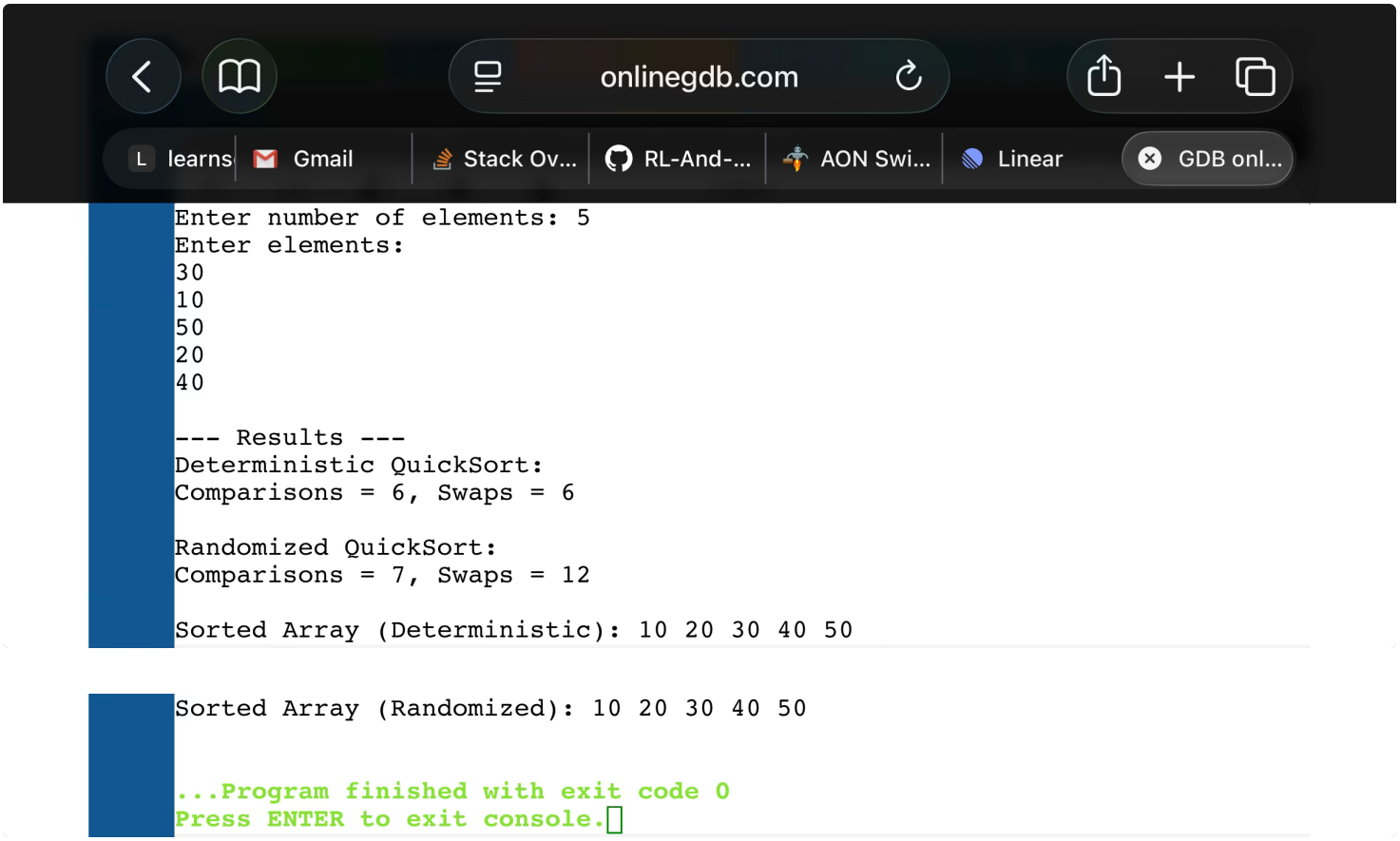
Program

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// Global counters for analysis
long long comparisons_det = 0, comparisons_rand = 0;
long long swaps_det = 0, swaps_rand = 0;

// Function to swap two elements
void swap(int *a, int *b, long long *swap_counter) {
    int temp = *a;
    *a = *b;
    *b = temp;
    (*swap_counter)++;
}

// Partition for deterministic quicksort (pivot = last element)
int partition_deterministic(int arr[], int low, int high) {
    int pivot = arr[high];
    int i = low - 1;
    for (int j = low; j < high; j++) {
        comparisons_det++;
        if (arr[j] < pivot) {
            i++;
            swap(&arr[i], &arr[j], &swaps_det);
        }
    }
    swap(&arr[i + 1], &arr[high], &swaps_det);
    return i + 1;
}
```

```
// Deterministic QuickSort
void quicksort_deterministic(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition_deterministic(arr, low, high);
        quicksort_deterministic(arr, low, pi - 1);
        quicksort_deterministic(arr, pi + 1, high);
    }
}

// Partition for randomized quicksort
int partition_randomized(int arr[], int low, int high) {
    int random_index = low + rand() % (high - low + 1);
    swap(&arr[random_index], &arr[high], &swaps_rand); // put random pivot at end
    int pivot = arr[high];
    int i = low - 1;
    for (int j = low; j < high; j++) {
        comparisons_rand++;
        if (arr[j] < pivot) {
            i++;
            swap(&arr[i], &arr[j], &swaps_rand);
        }
    }
    swap(&arr[i + 1], &arr[high], &swaps_rand);
    return i + 1;
}

// Randomized QuickSort
void quicksort_randomized(int arr[], int low, int high) {
    if (low < high) {
        int pi = partition_randomized(arr, low, high);
        quicksort_randomized(arr, low, pi - 1);
        quicksort_randomized(arr, pi + 1, high);
    }
}

// Utility to copy arrays
```

```
void copy_array(int src[], int dest[], int n) {
    for (int i = 0; i < n; i++) dest[i] = src[i];
}

int main() {
    srand(time(0));
    int n;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    int arr[n], arr_copy[n];
    printf("Enter elements:\n");
    for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
    // Copy array for randomized version
    copy_array(arr, arr_copy, n);
    // Deterministic QuickSort
    quicksort_deterministic(arr, 0, n - 1);
    // Randomized QuickSort
    quicksort_randomized(arr_copy, 0, n - 1);

    // Output
    printf("\n--- Results ---\n");
    printf("Deterministic QuickSort:\nComparisons = %lld, Swaps = %lld\n",
           comparisons_det, swaps_det);
    printf("\nRandomized QuickSort:\nComparisons = %lld, Swaps = %lld\n",
           comparisons_rand, swaps_rand);
    printf("\nSorted Array (Deterministic): ");
    for (int i = 0; i < n; i++) printf("%d ", arr[i]);
    printf("\n");
    printf("Sorted Array (Randomized): ");
    for (int i = 0; i < n; i++) printf("%d ", arr_copy[i]);
    printf("\n");
    return 0;
}
```

Output


onlinegdb.com
learns
Gmail
Stack Ov...
RL-And-...
AON Swi...
Linear
GDB onl...
Enter number of elements: 5
Enter elements:
30
10
50
20
40
--- Results ---
Deterministic QuickSort:
Comparisons = 6, Swaps = 6
Randomized QuickSort:
Comparisons = 7, Swaps = 12
Sorted Array (Deterministic): 10 20 30 40 50
Sorted Array (Randomized): 10 20 30 40 50
...Program finished with exit code 0
Press ENTER to exit console.