

## Experiment - 9

**Student Name:** Pankaj Singh Kanyal  
**Branch:** AIML  
**Semester:** 5th  
**Subject Name:** Advanced Programming Lab

**UID:** 20BCS6668  
**Section/Group:** AIML 4 B  
**Date of Performance:**    /    /2022  
**Subject Code:** 20CSP-334

### 1. AIM:

Design a quick sort with random pivoting using the Lomuto partition scheme.

### 2. Apparatus:

- Texeditor
- Laptop / PC with C++ compiler

### 3. Algorithm/Theory

In QuickSort we first partition the array in place such that all elements to the left of the pivot element are smaller, while all elements to the right of the pivot are greater than the pivot. Then we recursively call the same procedure for left and right subarrays.

Unlike merge sort, we don't need to merge the two sorted arrays. Thus Quicksort requires lesser auxiliary space than Merge Sort, which is why it is often preferred to Merge Sort. Using a randomly generated pivot we can further improve the time complexity of QuickSort.

### 4. Program/Code

```
#include <cstdlib>
#include <time.h>
#include <iostream>
using namespace std;
int partition(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);
}

int partition_r(int arr[], int low, int high)
{ srand(time(NULL));
  int random = low + rand() % (high - low);

  swap(arr[random], arr[high]);

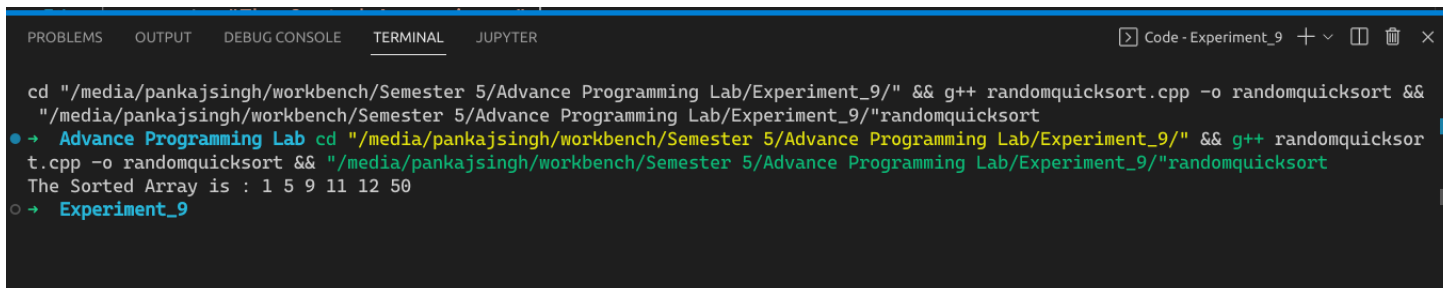
  return partition(arr, low, high);
}

void quickSort(int arr[], int low, int high)
{
    if (low < high) {
        int pi = partition_r(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

void printArray(int arr[], int size)
{
    cout<<"The Sorted Array is : ";
    int i;
    for (i = 0; i < size; i++)
        cout<<arr[i]<<" ";
    cout<<endl;
}

int main()
{
    int arr[] = { 11, 5, 9, 1, 12, 50 };
    int n = sizeof(arr) / sizeof(arr[0]);
    quickSort(arr, 0, n - 1);
    printArray(arr, n);
    return 0;
}
```

## 6. Output



```

cd "/media/pankajsingh/workbench/Semester 5/Advance Programming Lab/Experiment_9/" && g++ randomquicksort.cpp -o randomquicksort &&
"/media/pankajsingh/workbench/Semester 5/Advance Programming Lab/Experiment_9/"randomquicksort
• → Advance Programming Lab cd "/media/pankajsingh/workbench/Semester 5/Advance Programming Lab/Experiment_9/" && g++ randomquicksor
t.cpp -o randomquicksort && "/media/pankajsingh/workbench/Semester 5/Advance Programming Lab/Experiment_9/"randomquicksort
The Sorted Array is : 1 5 9 11 12 50
○ → Experiment_9
  
```

## 7. Learning Outcomes:

1. Learned the concepts of Quick sort.
2. Learned the concepts of random pivoting.
3. Learned to write a program for the above problem.
4. Learned to use vs code effectively.

**Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):**

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			