PRACTICAL-6

Aim:-Randomized-Quick Sort

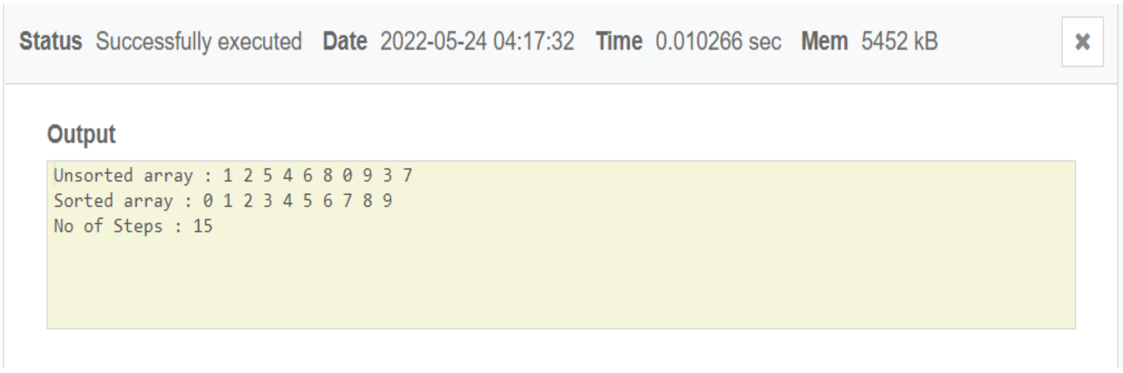
Code:

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 10
int step = 0;
void random_shuffle(int
arr[])
{
srand(time(NULL)); //
int i, j, temp;
for (i = MAX - 1; i > 0; i--)
{
j = rand()%(i + 1);
temp = arr[i]; arr[i]
= arr[j]; arr[j] =
temp;
}
}
void swap(int *a, int *b)
{  int
temp;
temp =*a;
*a = *b;
*b = temp;
}
int partion(int arr[], int p, int r)
{ int pivotIndex = p + rand()%(r - p + 1); //generates a random number as
a
pivot int pivot; int i = p - 1; int j;
pivot = arr[pivotIndex];
swap(&arr[pivotIndex], &arr[r]);
for (j = p; j < r; j++)
{
if (arr[j] < pivot)
{
i++;
swap(
&arr[i
],
&arr[j
]);
} }
swap(&arr[i+1],
&arr[r]);  return i +
1; }
void quick_sort(int arr[], int p, int q)
{ int
j;
step++;
if (p < q) {
j = partion(arr, p, q);
quick_sort(arr, p, j-1);
quick_sort(arr, j+1, q);
}
}


int main()
{
int i;
int arr[MAX];
for (i = 0; i < MAX; i++) arr[i] = i;
random_shuffle(arr); //To randomize the array
printf("Unsorted array : ");  for(int
i=0 ;i<MAX ;i++)
printf("%d ", arr[i]);

quick_sort(arr, 0, MAX-1); //function to sort the elements array
printf("\nSorted array : ");
for (i = 0; i < MAX; i++)
printf("%d ", arr[i]);
printf("\nNo of Steps :
%d",step);  return 0; }
```

Output-



Status Successfully executed Date 2022-05-24 04:17:32 Time 0.010266 sec Mem 5452 kB

Output

```
Unsorted array : 1 2 5 4 6 8 0 9 3 7
Sorted array : 0 1 2 3 4 5 6 7 8 9
No of Steps : 15
```