Name : Shrinivas Hatyalikar

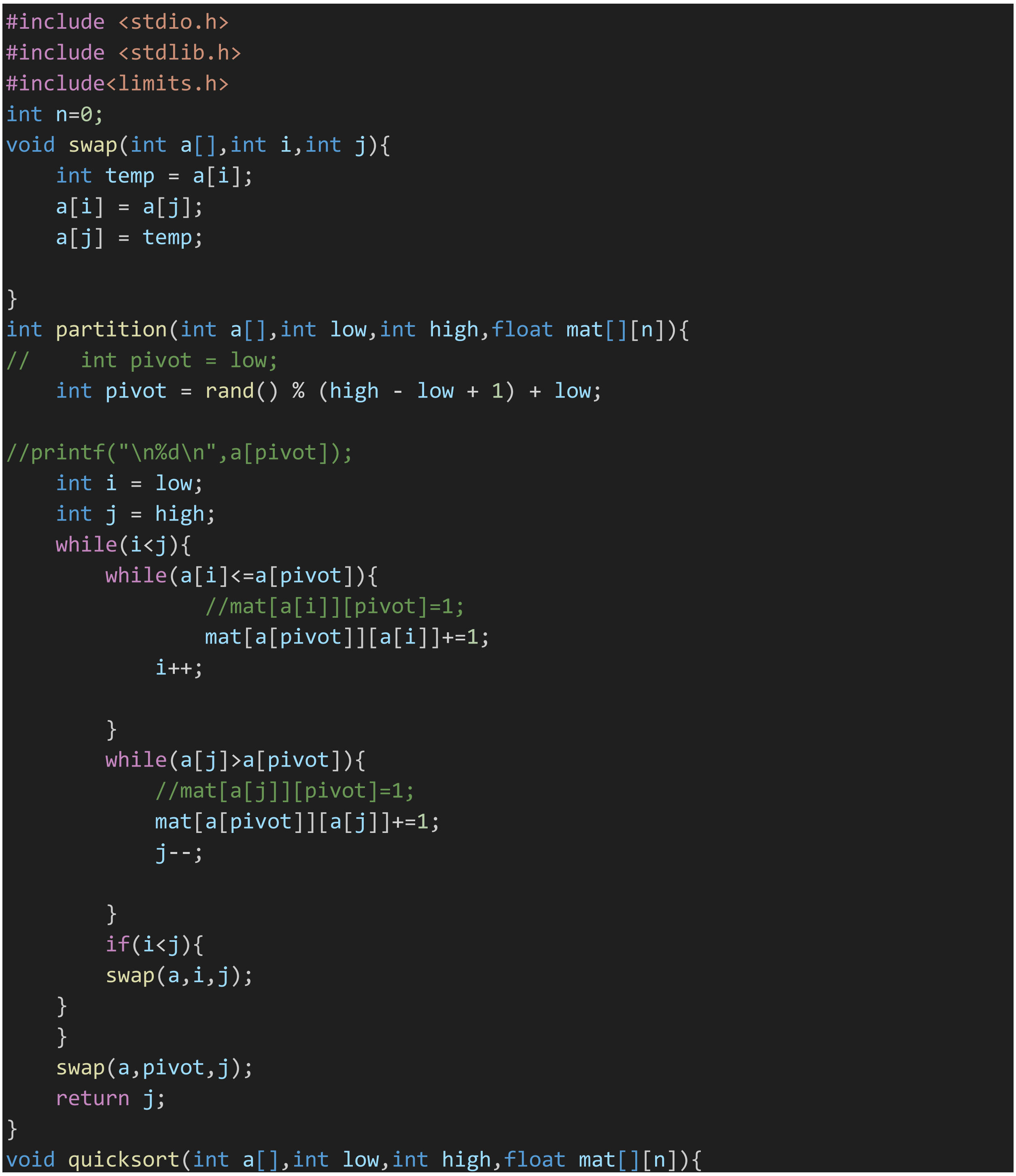
Div: TY-B (B2)

Roll No: 26

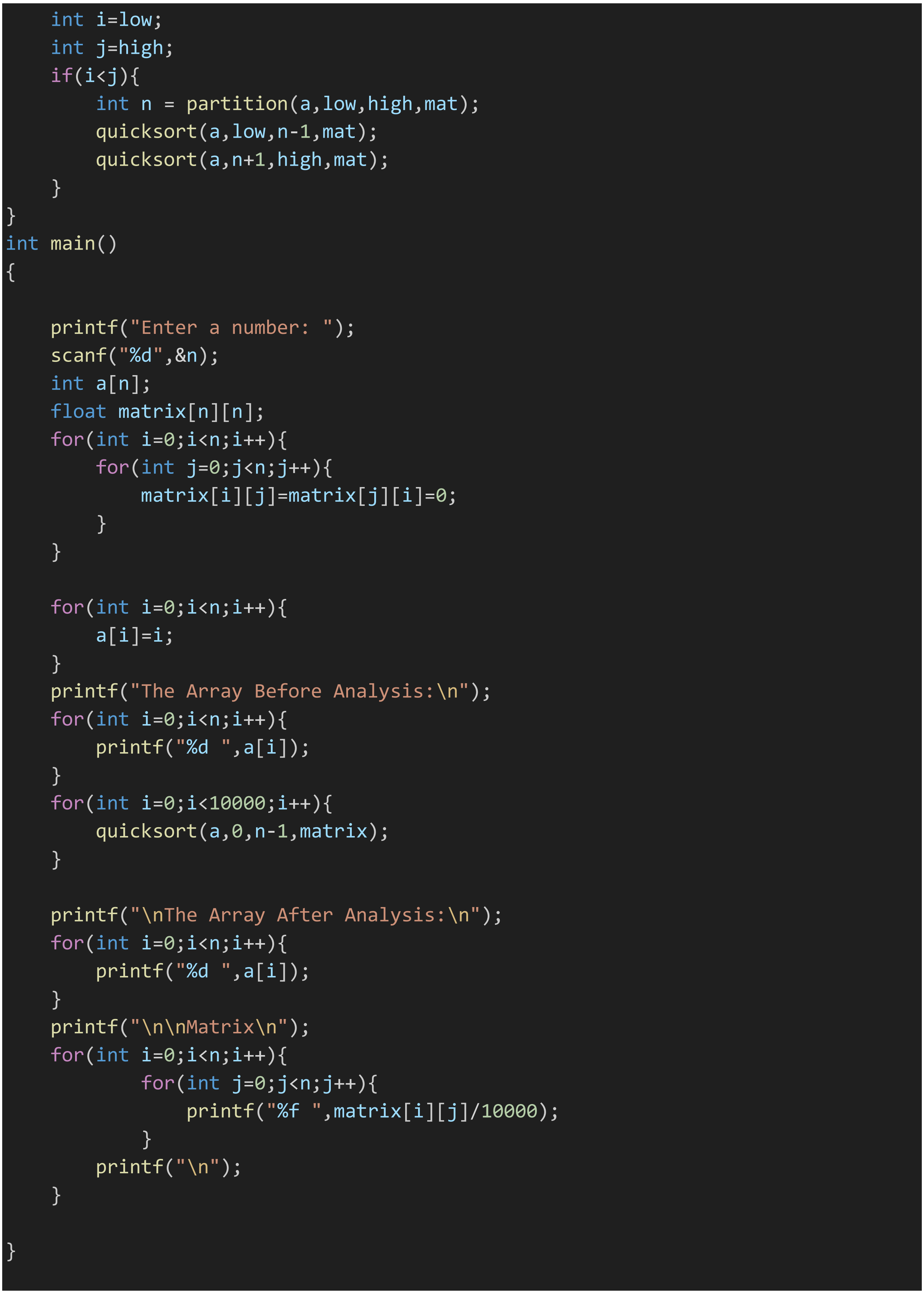
Q) Experimental analysis of Quick sort and variants

Part 1 : Taking a Random element as Pivot

Code:

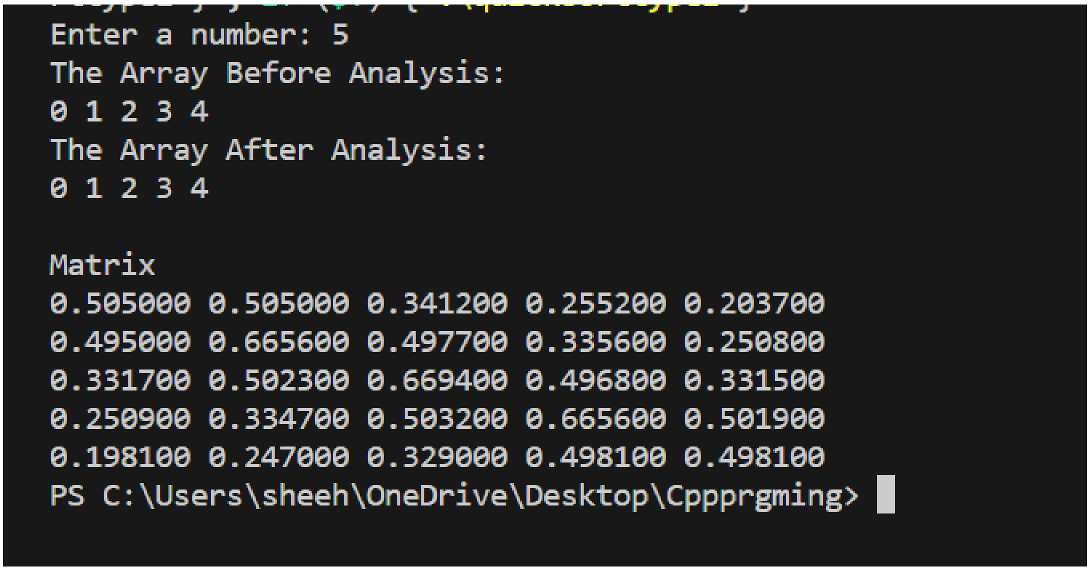


```
#include <stdio.h>
#include <stdlib.h>
#include<limits.h>
int n=0;
void swap(int a[],int i,int j){
    int temp = a[i];
    a[i] = a[j];
    a[j] = temp;

}
int partition(int a[],int low,int high,float mat[][n]){
//    int pivot = low;
    int pivot = rand() % (high - low + 1) + low;

//printf("\n%d\n",a[pivot]);
    int i = low;
    int j = high;
    while(i<j){
        while(a[i]<=a[pivot]){
                //mat[a[i]][pivot]=1;
                mat[a[pivot]][a[i]]+=1;
            i++;

        }
        while(a[j]>a[pivot]){
            //mat[a[j]][pivot]=1;
            mat[a[pivot]][a[j]]+=1;
            j--;

        }
        if(i<j){
        swap(a,i,j);
    }
    }
    swap(a,pivot,j);
    return j;
}
void quicksort(int a[],int low,int high,float mat[][n]){
```



```c
    int i=low;
    int j=high;
    if(i<j){
        int n = partition(a,low,high,mat);
        quicksort(a,low,n-1,mat);
        quicksort(a,n+1,high,mat);
    }
}
int main()
{

    printf("Enter a number: ");
    scanf("%d",&n);
    int a[n];
    float matrix[n][n];
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            matrix[i][j]=matrix[j][i]=0;
        }
    }

    for(int i=0;i<n;i++){
        a[i]=i;
    }
    printf("The Array Before Analysis:\n");
    for(int i=0;i<n;i++){
        printf("%d ",a[i]);
    }
    for(int i=0;i<10000;i++){
        quicksort(a,0,n-1,matrix);
    }

    printf("\nThe Array After Analysis:\n");
    for(int i=0;i<n;i++){
        printf("%d ",a[i]);
    }
    printf("\n\nMatrix\n");
    for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                printf("%f ",matrix[i][j]/10000);
            }
        printf("\n");
    }

}
```

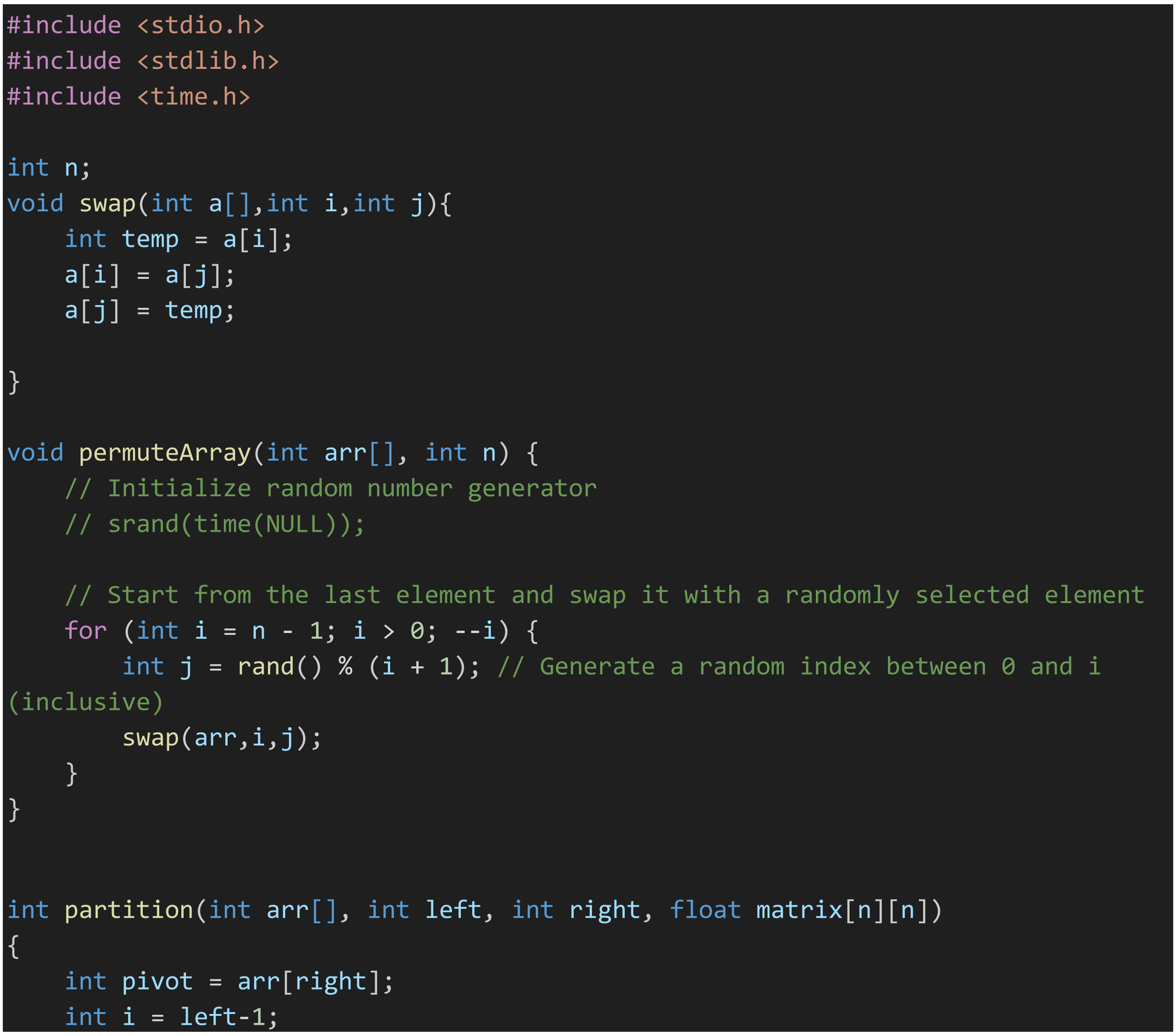
Output:



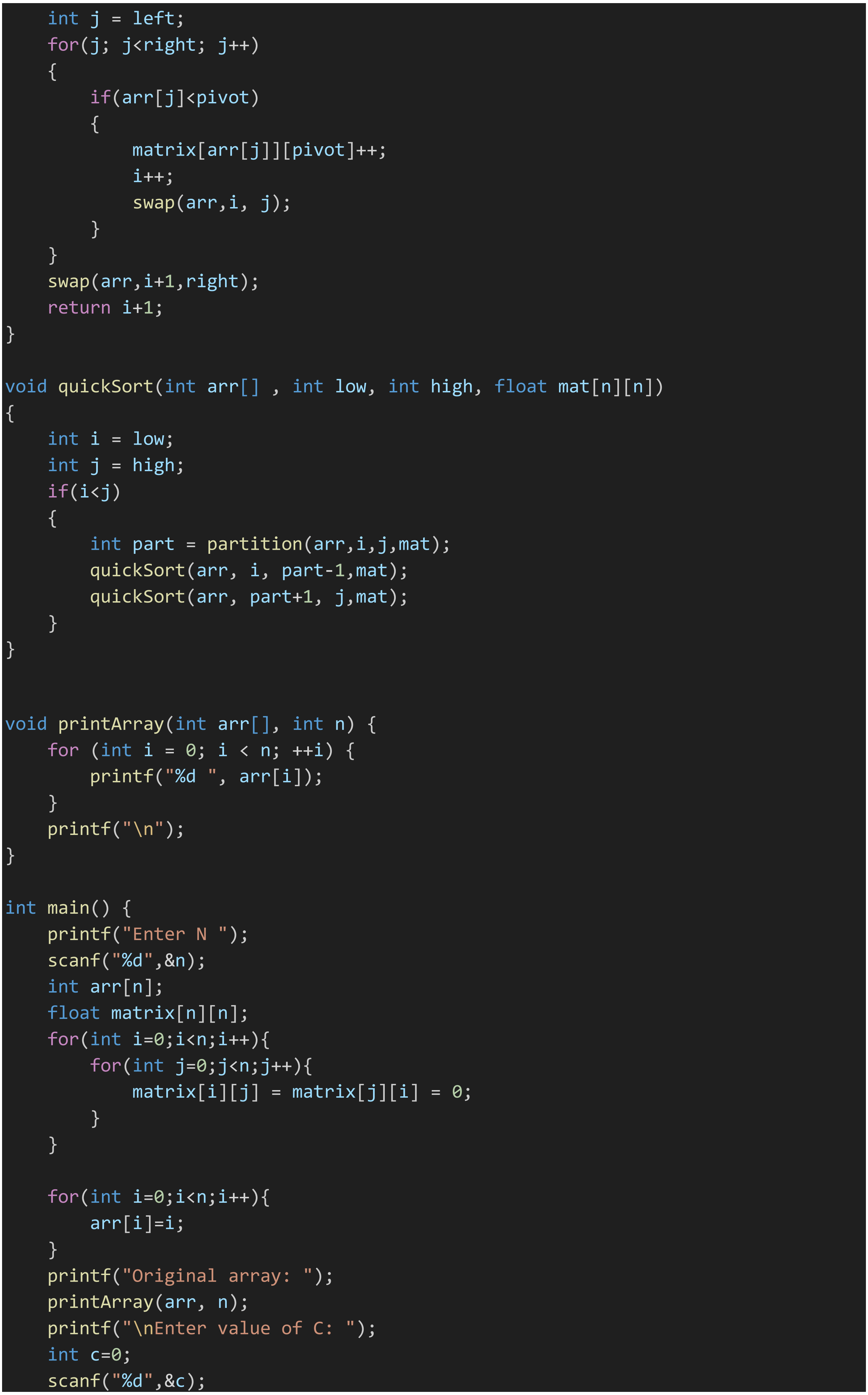
```
Enter a number: 5
The Array Before Analysis:
0 1 2 3 4
The Array After Analysis:
0 1 2 3 4

Matrix
0.505000 0.505000 0.341200 0.255200 0.203700
0.495000 0.665600 0.497700 0.335600 0.250800
0.331700 0.502300 0.669400 0.496800 0.331500
0.250900 0.334700 0.503200 0.665600 0.501900
0.198100 0.247000 0.329000 0.498100 0.498100
PS C:\Users\sheeh\OneDrive\Desktop\Cppprgming>
```

Part 2: Permutation of Array and sorting using deterministic method (ie. Pivot as last element)

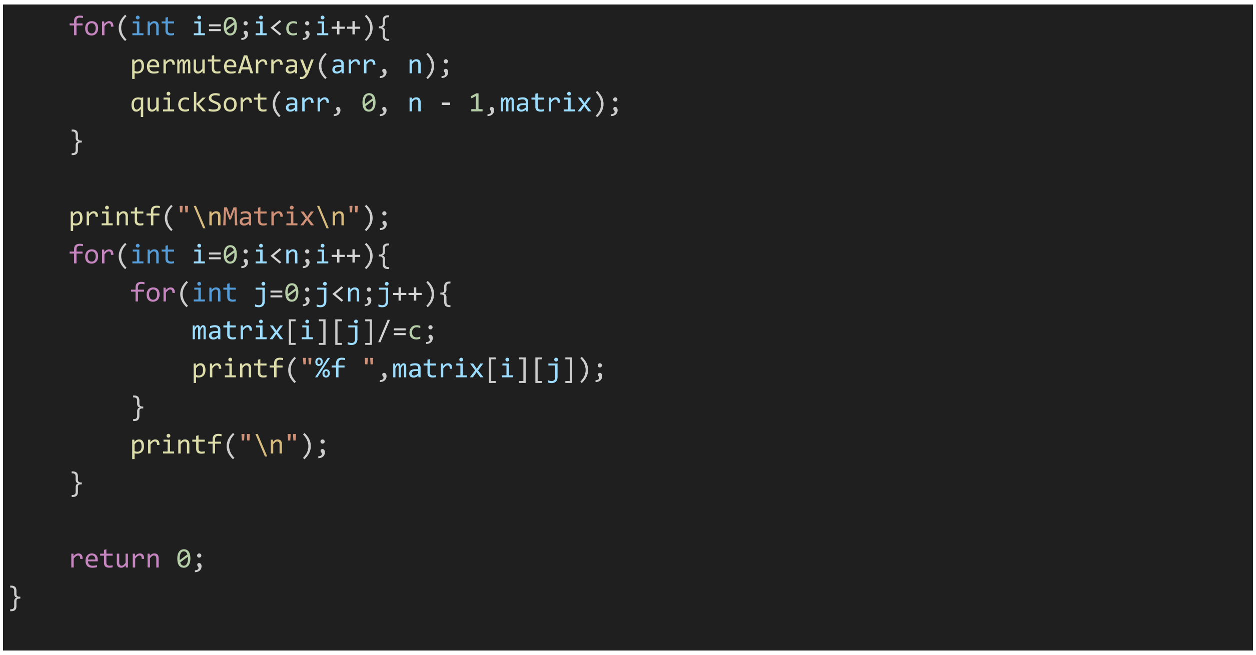
Code:



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

int n;
void swap(int a[],int i,int j){
    int temp = a[i];
    a[i] = a[j];
    a[j] = temp;

}

void permuteArray(int arr[], int n) {
    // Initialize random number generator
    // srand(time(NULL));

    // Start from the last element and swap it with a randomly selected element
    for (int i = n - 1; i > 0; --i) {
        int j = rand() % (i + 1); // Generate a random index between 0 and i
(inclusive)
        swap(arr,i,j);
    }
}


int partition(int arr[], int left, int right, float matrix[n][n])
{
    int pivot = arr[right];
    int i = left-1;
```

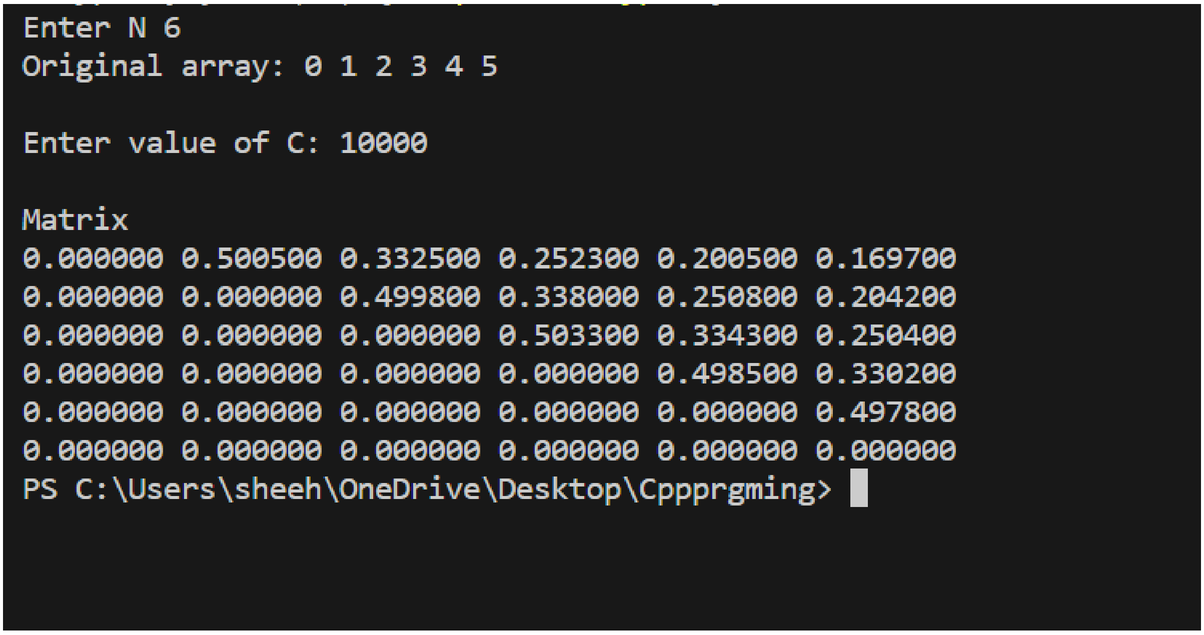


```c
    int j = left;
    for(j; j<right; j++)
    {
        if(arr[j]<pivot)
        {
            matrix[arr[j]][pivot]++;
            i++;
            swap(arr,i, j);
        }
    }
    swap(arr,i+1,right);
    return i+1;
}

void quickSort(int arr[] , int low, int high, float mat[n][n])
{
    int i = low;
    int j = high;
    if(i<j)
    {
        int part = partition(arr,i,j,mat);
        quickSort(arr, i, part-1,mat);
        quickSort(arr, part+1, j,mat);
    }
}


void printArray(int arr[], int n) {
    for (int i = 0; i < n; ++i) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main() {
    printf("Enter N ");
    scanf("%d",&n);
    int arr[n];
    float matrix[n][n];
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            matrix[i][j] = matrix[j][i] = 0;
        }
    }

    for(int i=0;i<n;i++){
        arr[i]=i;
    }
    printf("Original array: ");
    printArray(arr, n);
    printf("\nEnter value of C: ");
    int c=0;
    scanf("%d",&c);
```



```
    for(int i=0;i<c;i++){
        permuteArray(arr, n);
        quickSort(arr, 0, n - 1,matrix);
    }

    printf("\nMatrix\n");
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            matrix[i][j]/=c;
            printf("%f ",matrix[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Output:



```
Enter N 6
Original array: 0 1 2 3 4 5

Enter value of C: 10000

Matrix
0.000000 0.500500 0.332500 0.252300 0.200500 0.169700
0.000000 0.000000 0.499800 0.338000 0.250800 0.204200
0.000000 0.000000 0.000000 0.503300 0.334300 0.250400
0.000000 0.000000 0.000000 0.000000 0.498500 0.330200
0.000000 0.000000 0.000000 0.000000 0.000000 0.497800
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
PS C:\Users\sheeh\OneDrive\Desktop\Cppprgming> 
```