Lab 9: Implementation of Sorting Techniques.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void GetArray(int a[],int n);
void BubbleSort(int a[],int n);
void InsertionSort(int a[],int n);
void SelectionSort(int a[],int n);
int partition(int a[],int l,int r);
void quick_sort(int a[],int l,int r);
void merge(int a[],int l,int m,int r);
void merge_sort(int a[],int l,int r);
void display(int *p,int n);
int a[1000],b[1000];
int main()
{
    int n,choice;
    srand(time(0));
    do
    {
        printf("\n1. Get Array 2. Bubble Sort 3. Selection Sort
4. Insertion Sort 5. Quick Sort 6. Merge Sort 7. EXIT\n");
        printf("Enter Your Choice : ");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:
            printf("Enter n : ");
            scanf("%d",&n);
            GetArray(a,n);
            break;

        case 2:
            BubbleSort(a,n);
            display(a,n);
            break;
        case 3:
            SelectionSort(a,n);
            display(a,n);
            break;

        case 4:
            InsertionSort(a,n);
            display(a,n);
            break;

        case 5:
            quick_sort(a,0,n-1);
            display(a,n);
            break;

        case 6:
            merge_sort(a,0,n-1);
            display(a,n);
            break;
        }
    }while(choice!=7);
    return 0;
}
void GetArray(int a[],int n)
{
    int i;
    for(i=0;i<n;i++)
        a[i]=rand();
    display(a,n);
}
void display(int a[],int n)
{
    int i;
    printf("\n--------------------------------------------------------------
-----------------------------------------------------------------------
\n");
    for(i=0;i<n;i++)
        printf("%d \t",a[i]);

}

void BubbleSort(int a[],int n)
{
    int i,j,temp;
    for(i=0;i<n-1;i++)
    {
        for(j=0;j<n-i-1;j++)
        {
            if(a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
}

void InsertionSort(int a[],int n)
{
    int i,temp,j;
    for(i=0;i<n;i++)
    {
        j=i-1;
        temp=a[i];
        while(j>=0 && temp<a[j])
        {
           a[j+1]=a[j];
           j--;
        }
        a[j+1]=temp;
    }
}

void SelectionSort(int a[],int n)
{
    int i,j,least,p,temp;
    for(i=0;i<n;i++)
    {
        least=a[i];
        p=i;
        for(j=i+1;j<n;j++)
        {
            if(a[j]<least)
            {
                least=a[j];
                p=j;
            }
        }
        if(i!=p)
        {
            temp=a[p];
            a[p]=a[i];
            a[i]=temp;
        }
    }
}

void quick_sort(int a[],int l,int r)
{
    int p;
    if(l<r)
    {
        p=partition(a,l,r);
        quick_sort(a,l,p-1);
        quick_sort(a,p+1,r);
    }
}
int partition(int a[],int l,int r)
{
    int x=l;
    int y=r;
    int p=a[l];
    int temp;
    while(x<y)
    {
        while(a[x]<=p)
            x++;
        while(a[y]>p)
            y--;
        if(x<y)
        {
            temp=a[y];
            a[y]=a[x];
            a[x]=temp;
        }
    }
        a[l]=a[y];
        a[y]=p;
        return y;
}

void merge_sort(int a[],int l,int r)
{
    int m;
    if(l<r)
    {
        m=(l+r)/2;
        merge_sort(a,l,m);
        merge_sort(a,m+1,r);
        merge(a,l,m+1,r);
    }
}
void merge(int a[],int l,int m,int r)
{
    int x=l;
    int k=l;
    int y=m;
    int i;
    while(x<m && y<=r)
    {
        if (a[x]<a[y])
            b[k++]=a[x++];
        else
            b[k++]=a[y++];
    }
    for(;x<m;x++,k++)
        b[k]=a[x];
    for(;y<=r;y++,k++)
        b[k]=a[y];
    for(i=l;i<=r;i++)
        a[i]=b[i];
}
```