exp_11/a.c

```c
#include <stdio.h>

void insertionSort(int arr[], int n) {
    for (int i = 1; i < n; i++) {
        int key = arr[i], j = i - 1;
        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j--;
        }
        arr[j + 1] = key;
    }
}

void selectionSort(int arr[], int n) {
    for (int i = 0; i < n - 1; i++) {
        int min_idx = i;
        for (int j = i + 1; j < n; j++)
            if (arr[j] < arr[min_idx])
                min_idx = j;
        int temp = arr[i];
        arr[i] = arr[min_idx];
        arr[min_idx] = temp;
    }
}

void quickSort(int arr[], int low, int high) {
    if (low < high) {
        int pivot = arr[high], i = low - 1;
        for (int j = low; j < high; j++) {
            if (arr[j] < pivot) {
                i++;
                int tmp = arr[i];
                arr[i] = arr[j];
                arr[j] = tmp;
            }
        }
        int tmp = arr[i+1];
        arr[i+1] = arr[high];
        arr[high] = tmp;
        int pi = i+1;
        quickSort(arr, low, pi-1);
        quickSort(arr, pi+1, high);
    }
}

void merge(int arr[], int l, int m, int r) {
    int n1 = m - l + 1, n2 = r - m;
    int L[100], R[100];
```

```c
    for (int i = 0; i < n1; i++) L[i] = arr[l + i];
    for (int j = 0; j < n2; j++) R[j] = arr[m + 1 + j];
    int i = 0, j = 0, k = l;
    while (i < n1 && j < n2)
        arr[k++] = (L[i] <= R[j]) ? L[i++] : R[j++];
    while (i < n1) arr[k++] = L[i++];
    while (j < n2) arr[k++] = R[j++];
}

void mergeSort(int arr[], int l, int r) {
    if (l < r) {
        int m = l + (r-l)/2;
        mergeSort(arr, l, m);
        mergeSort(arr, m+1, r);
        merge(arr, l, m, r);
    }
}

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

int main() {
    int arr[100], n, choice;
    printf("Enter number of elements: ");
    scanf("%d", &n);
    printf("Enter elements: ");
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);
    while (1) {
        printf("1. Insertion Sort\n2. Selection Sort\n3. Quick Sort\n4. Merge Sort\n0. Exit\nEnter choice: ");
        scanf("%d", &choice);
        if (choice == 0) break;
        switch (choice) {
            case 1:
                insertionSort(arr, n);
                printf("Sorted array: ");
                displayArray(arr, n);
                break;
            case 2:
                selectionSort(arr, n);
                printf("Sorted array: ");
                displayArray(arr, n);
                break;
            case 3:
                quickSort(arr, 0, n-1);
                printf("Sorted array: ");
```

```
                displayArray(arr, n);
                break;
            case 4:
                mergeSort(arr, 0, n-1);
                printf("Sorted array: ");
                displayArray(arr, n);
                break;
            default:
                printf("Invalid choice\n");
        }
    }
    return 0;
}

```