//Program to implementing sorting

#include <stdio.h>

// Function to perform Quick Sort

void quickSort(int arr[], int low, int high) {

    if (low < high) {

        int pivot = arr[low];

        int i = low + 1;

        int j = high;

        while (1) {

            while (i <= high && arr[i] < pivot) {

                i++;

            }

            while (arr[j] > pivot) {

                j--;

            }

            if (i < j) {

                int temp = arr[i];

                arr[i] = arr[j];

                arr[j] = temp;

            } else {

                break;

            }

        }

        arr[low] = arr[j];

        arr[j] = pivot;

        printf("Partitioned Array: ");

        for (int k = low; k <= high; k++) {

            printf("%d ", arr[k]);

        }

        printf("\n");

        quickSort(arr, low, j - 1);

        quickSort(arr, j + 1, high);

    }

}

// Function to perform Insertion Sort

void insertionSort(int arr[], int n) {

    for (int i = 1; i < n; i++) {

        int key = arr[i];

        int j = i - 1;

        while (j >= 0 && arr[j] > key) {

            arr[j + 1] = arr[j];

            j--;

        }

        arr[j + 1] = key;

        printf("After Pass %d: ", i);

        for (int k = 0; k < n; k++) {

            printf("%d ", arr[k]);

        }

        printf("\n");

    }

}

// Function to perform Heap Sort

void heapify(int arr[], int n, int i) {

    int largest = i;

    int left = 2 \* i + 1;

    int right = 2 \* i + 2;

    if (left < n && arr[left] > arr[largest]) {

        largest = left;

    }

    if (right < n && arr[right] > arr[largest]) {

        largest = right;

    }

    if (largest != i) {

        int temp = arr[i];

        arr[i] = arr[largest];

        arr[largest] = temp;

        heapify(arr, n, largest);

    }

}

void heapSort(int arr[], int n) {

    for (int i = n / 2 - 1; i >= 0; i--) {

        heapify(arr, n, i);

    }

    for (int i = n - 1; i > 0; i--) {

        int temp = arr[0];

        arr[0] = arr[i];

        arr[i] = temp;

        printf("After Pass %d: ", n - i);

        for (int k = 0; k < n; k++) {

            printf("%d ", arr[k]);

        }

        printf("\n");

        heapify(arr, i, 0);

    }

}

int main() {

    int choice, n;

    printf("Enter the number of elements: ");

    scanf("%d", &n);

    int arr[n];

    printf("Enter the elements:\n");

    for (int i = 0; i < n; i++) {

        scanf("%d", &arr[i]);

    }

    while (1) {

        printf("\nMenu:\n");

        printf("1. Quick Sort\n");

        printf("2. Insertion Sort\n");

        printf("3. Heap Sort\n");

        printf("4. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                printf("Quick Sort Steps:\n");

                quickSort(arr, 0, n - 1);

                printf("Quick Sort Result: ");

                for (int i = 0; i < n; i++) {

                    printf("%d ", arr[i]);

                }

                break;

            case 2:

                printf("Insertion Sort Steps:\n");

                insertionSort(arr, n);

                printf("Insertion Sort Result: ");

                for (int i = 0; i < n; i++) {

                    printf("%d ", arr[i]);

                }

                break;

            case 3:

                printf("Heap Sort Steps:\n");

                heapSort(arr, n);

                printf("Heap Sort Result: ");

                for (int i = 0; i < n; i++) {

                    printf("%d ", arr[i]);

                }

                break;

            case 4:

                printf("Exiting program.\n");

                return 0;

            default:

                printf("Invalid choice. Please enter a valid option.\n");

        }

    }

    return 0;

}