Insertion sort

Java

```java
import java.util.Scanner;
class Main {
    public static void main (String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();
        int[] ar = new int[n];

        for (int i=0; i<n; i++)
            ar[i] = scan.nextInt();

        ar = insertion_sort(ar, n);

        for (int i=0; i<n; i++) {
            System.out.print(ar[i] + " ");
        }
        System.out.println();
    }
    public static int[] insertion_sort (int[] ar, int n) {
        for (int i=1; i<n; i++) {
            int j = i;
            while (j>0 && ar[j-1] > ar[j]) {
                ar[j-1] = ar[j-1] + ar[j];
                ar[j] = ar[j-1] - ar[j];
                ar[j-1] = ar[j-1] - ar[j];

                j--;
            }
        }
        return ar;
    }
}
```

Python

```python
def insertion_sort(ar):
    for ii in range(1, len(ar)):
        j = ii
        while j > 0 and ar[j-1] > ar[j]:
            ar[j-1], ar[j] = ar[j], ar[j-1]
            j -= 1
    return ar


n = int(input())
arr = []
for i in range(0, n):
    elem = int(input())

    arr.append(elem)

arr = insertion_sort(arr)

print(arr)
```

C++

```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int n; cin >> n;

    vector <int> v(n);

    for (int &i : v)
        cin >> i;

    for (int i=1; i<n; i++) {
        int j=i;

        while (j > 0 and v[j-1] > v[j]) {
            swap (v[j-1], v[j]);

            j--;
        }
    }

    for (int &i : v)
        cout << i << " ";
    cout << endl;

    return 0;
}
```

Selection sort

Java

```java
import java.util.Scanner;
class Main {
    public static void main (String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();
        int[] ar = new int[n];

        for (int i=0; i<n; i++)
            ar[i] = scan.nextInt();

        ar = selection_sort(ar, n);

        for (int i=0; i<n; i++) {
            System.out.print(ar[i] + " ");
        }
        System.out.println();
    }
    public static int[] selection_sort (int[] ar, int n) {
        for (int i=0; i<n-1; i++) {
            int min_index = i;

            for (int j=i+1; j<n; j++) {
                if (ar[j] < ar[min_index]) {
                    min_index = j;
                }
            }

            int temp = ar[min_index];
            ar[min_index] = ar[i];
            ar[i] = temp;
        }
        return ar;
    }
}
```

Python

```python
def selection_sort (a):
    n = len(a)

    for i in range(n-1):
        min_ind = i

        for k in range(i+1, n):
            if a[k] < a[min_ind]:
                min_ind = k

        a[i], a[min_ind] = a[min_ind], a[i]
    return a


num = int(input())
arr = []
for j in range(0, num):
    elem = int(input())

    arr.append(elem)

arr = selection_sort(arr)
print(arr)
```

C++

```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int n; cin >> n;

    vector <int> v(n);

    for (int &i : v)
        cin >> i;

    for (int i=0; i<n-1; i++) {
        int min_ind = i;

        for (int j=i+1; j<n; j++)
            if (v[j] < v[min_ind])
                min_ind = j;

        swap (v[i], v[min_ind]);
    }

    for (int &i : v)
        cout << i << " ";
    cout << endl;

    return 0;
}
```

Merge Sort

C++

```cpp
#include <bits/stdc++.h>
using namespace std;
#define vi vector<int>

vi merge (vi left, vi right);

vi merge_sort (vi v)
{
    if (v.size() <= 1)
        return v;

    int middle = v.size()/2;
    vi left, right;

    for (int i=0; i<middle; i++)
        left.push_back(v[i]);

    for (int i=middle; i<v.size(); i++)
        right.push_back(v[i]);

    left = merge_sort (left);
    right = merge_sort (right);

    return merge (left, right);
}

vi merge (vi left, vi right) {
    int i=0, j=0;

    vi merged;

    while (i < left.size() and j < right.size()) {
        if (left[i] <= right[j]) {
            merged.push_back(left[i]);
            i++;
        }
        else {
```

```
            merged.push_back(right[j]);
            j++;
        }
    }

    while (i < left.size()) {
        merged.push_back(left[i]);
        i++;
    }

    while (j < right.size()) {
        merged.push_back(right[j]);
        j++;
    }

    return merged;
}

int main()
{
    int n; cin >> n;

    vector <int> v(n);

    for (int &i : v)
        cin >> i;

    v = merge_sort (v);

    for (int &i : v)
        cout << i << " ";
    cout << endl;

    return 0;
}
```

Python

```python
def merge_sort(a):
    if len(a) <= 1:
        return a

    mid = len(a)//2
    left_half = merge_sort(a[:mid])
    right_half = merge_sort(a[mid:])

    return merge (left_half, right_half)


def merge(left, right):
    i = j = 0

    merged = []

    while i < len(left) and j < len(right):
        if left[i] <= right[j]:
            merged.append(left[i])
            i += 1
        else:
            merged.append(right[j])
            j += 1

    while i < len(left):
        merged.append(left[i])
        i += 1

    while j < len(right):
        merged.append(right[j])
        j += 1

    return merged


num = int(input())
arr = []
for j in range(0, num):
    elem = int(input())
```

```
    arr.append(elem)

arr = merge_sort(arr)
print(arr)
```

Java

```
import java.util.Scanner;
class Main {
    public static void main (String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();
        int[] ar = new int[n];

        for (int i=0; i<n; i++)
            ar[i] = scan.nextInt();

        merge_sort(ar, 0, n-1);

        for (int i=0; i<n; i++) {
            System.out.print(ar[i] + " ");
        }
        System.out.println();
    }
    public static void merge_sort (int[] a, int l, int r) {
        if (l >= r)
            return;
        int mid = (l+r)/ 2;

        merge_sort(a, l, mid);
        merge_sort(a, mid+1, r);

        merge (a, l, mid, r);
    }
    public static void merge (int[] a, int l, int m, int r) {
        int n1 = m - l + 1;
        int n2 = r - m;
```

```
        int[] left = new int[n1];
        int[] right = new int[n2];

        for (int i=0; i<n1; i++)
            left[i] = a[l+i];

        for (int j=0; j<n2; j++)
            right[j] = a[m+1+j];

        int i=0, j=0;
        int k = l;

        while (i < n1 && j < n2) {
            if (left[i] <= right[j]) {
                a[k] = left[i];
                i++;
            }
            else {
                a[k] = right[j];
                j++;
            }
            k++;
        }

        while (i < n1) {
            a[k] = left[i];
            i++; k++;
        }

        while (j < n2) {
            a[k] = right[j];
            j++; k++;
        }
    }
}
```

Quick Sort

Python

```python
def quicksort(a, low, high):
    if low < high:
        p = hoare_partition(a, low, high)
        quicksort(a, low, p)
        quicksort(a, p+1, high)


def hoare_partition(a, low, high):
    pivot = a[low]
    i = low - 1
    j = high + 1

    while True:
        i += 1
        while a[i] < pivot:
            i += 1

        j -= 1
        while a[j] > pivot:
            j -= 1

        if i >= j:
            return j
        a[i], a[j] = a[j], a[i]


num = int(input())
arr = []
for j in range(0, num):
    elem = int(input())

    arr.append(elem)

quicksort(arr, 0, num-1)
print(arr)
```

C++

```cpp
#include <bits/stdc++.h>
using namespace std;
#define vi vector<int>

int hoare_partition (int a[], int l, int h)
{
    int pivot = a[l];
    int i = l-1;
    int j = h+1;

    while (1) {
        i++;
        while (a[i] < pivot)
            i++;

        j--;
        while (a[j] > pivot)
            j--;

        if (i >= j)
            return j;

        swap (a[i], a[j]);
    }
}

void quicksort (int a[], int l, int h)
{
    if (l < h) {
        int p = hoare_partition (a, l, h);

        quicksort (a, l, p);
        quicksort (a, p+1, h);
    }
    return;
}

int main()
{
    int n; cin >> n;
```

```
    int a[n];

    for (int i=0; i<n; i++)
        cin >> a[i];

    quicksort (a, 0, n-1);

    for (int i=0; i<n; i++)
        cout << a[i] << " ";
    cout << endl;

    return 0;
}
```

Java

```java
import java.util.Scanner;
class Main {
    public static void main (String[] args) {
        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();
        int[] ar = new int[n];

        for (int i=0; i<n; i++)
            ar[i] = scan.nextInt();

        quicksort(ar, 0, n-1);

        for (int i=0; i<n; i++) {
            System.out.print(ar[i] + " ");
        }
        System.out.println();
    }
    public static int hoare_partition (int[] a, int l, int h) {
        int pivot = a[l];
        int i = l-1;
        int j = h+1;

        while (true) {
            i++;
            while (a[i] < pivot)
                i++;

            j--;
            while (a[j] > pivot)
                j--;

            if (i >= j)
                return j;

            int temp = a[i];
            a[i] = a[j];
            a[j] = temp;
        }
    }
```

```
    public static void quicksort (int[] a, int l ,int h) {
        if (l < h) {
            int p = hoare_partition(a, l, h);
            quicksort(a, l, p);
            quicksort(a, p+1, h);
        }
    }
}
```