Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа**

**№10**

Выполнил:   
студент группы РИС-23-1б   
Фаезов Александр

Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

Пермь, 2024 г.

**Условие:**

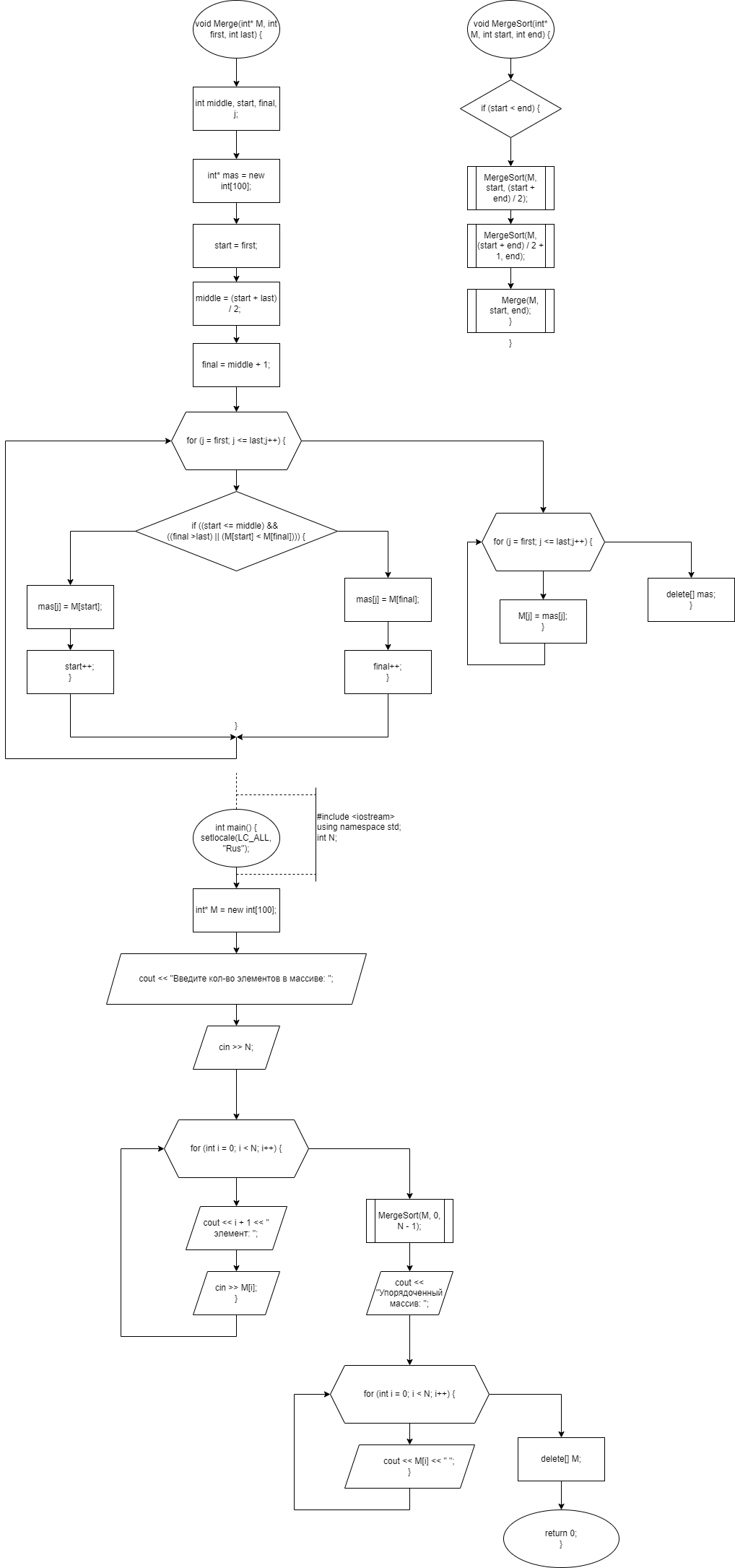
Разобрать все методы сортировки.

**Анализ задачи:**

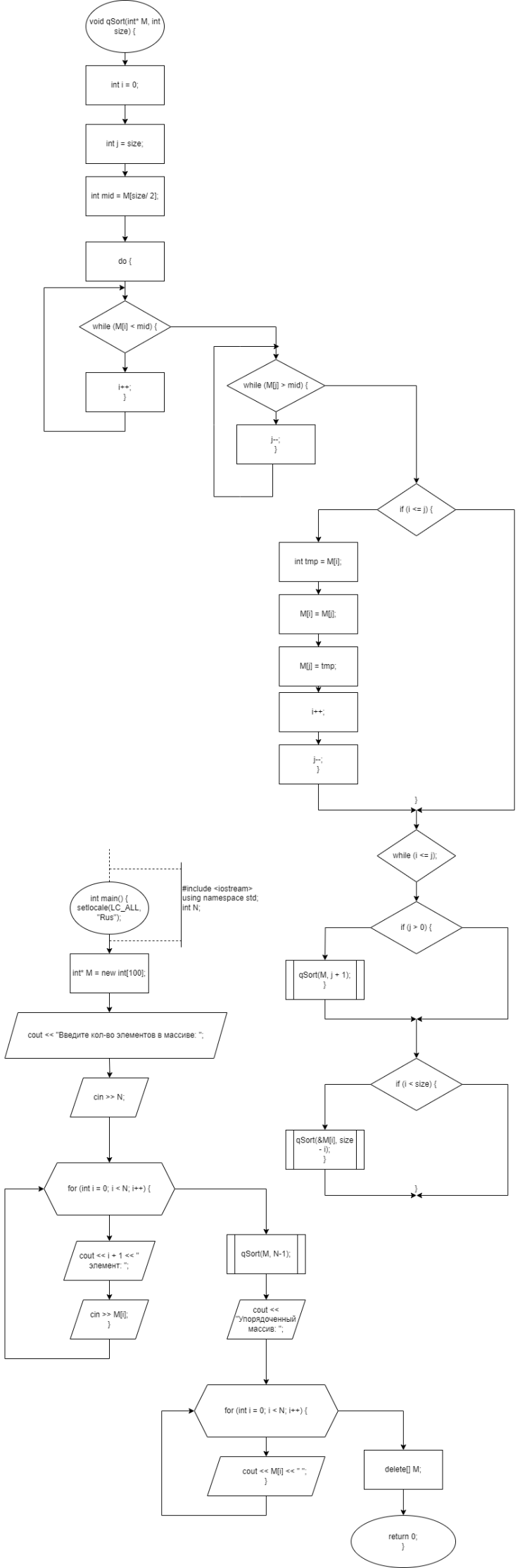
1. Изучить определённый метод сортировки
2. Выполнить его реализацию на коде

**Блок-схема:**

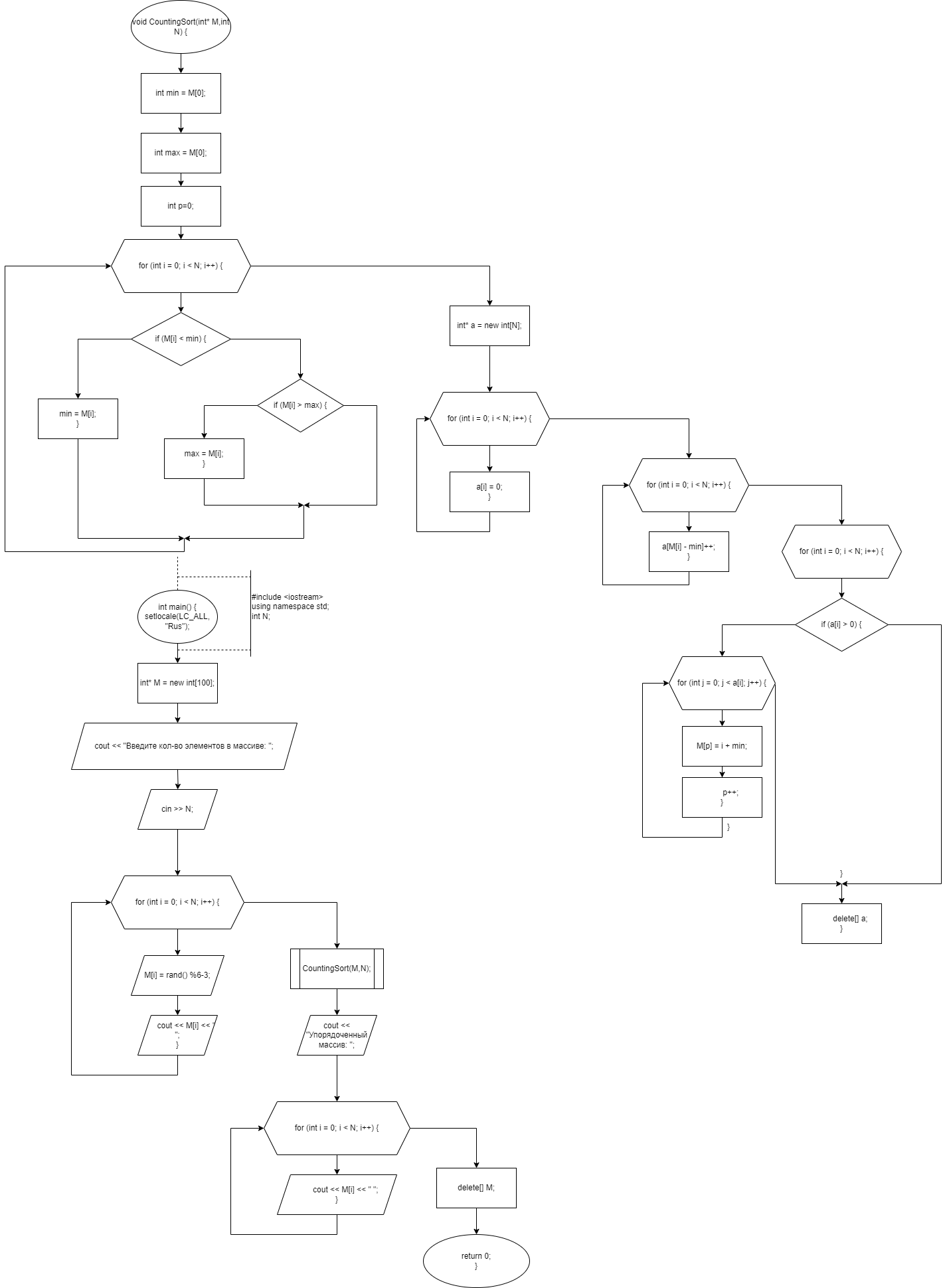
1) Сортировка слиянием

****

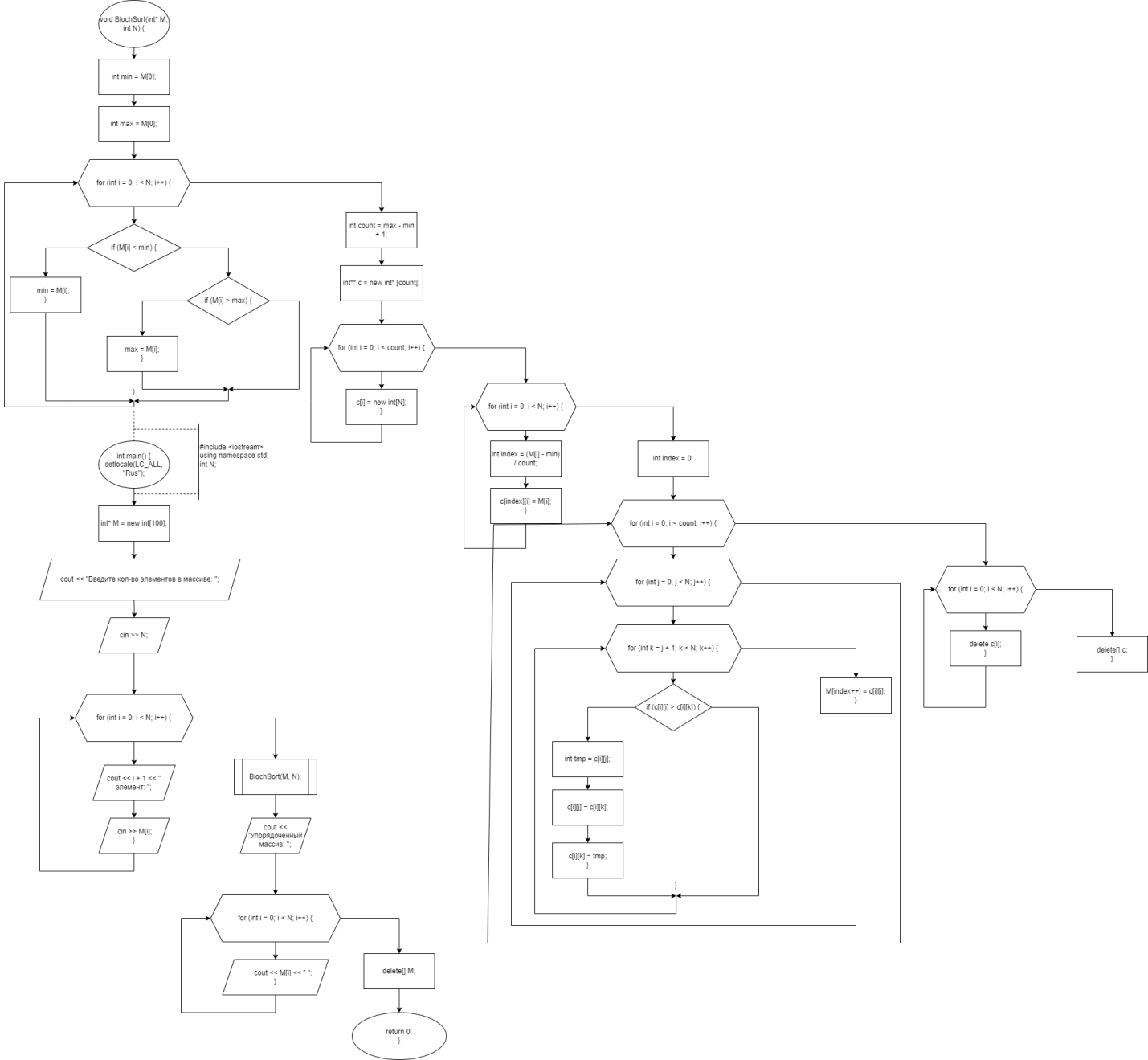
2) Быстрая сортировка

****

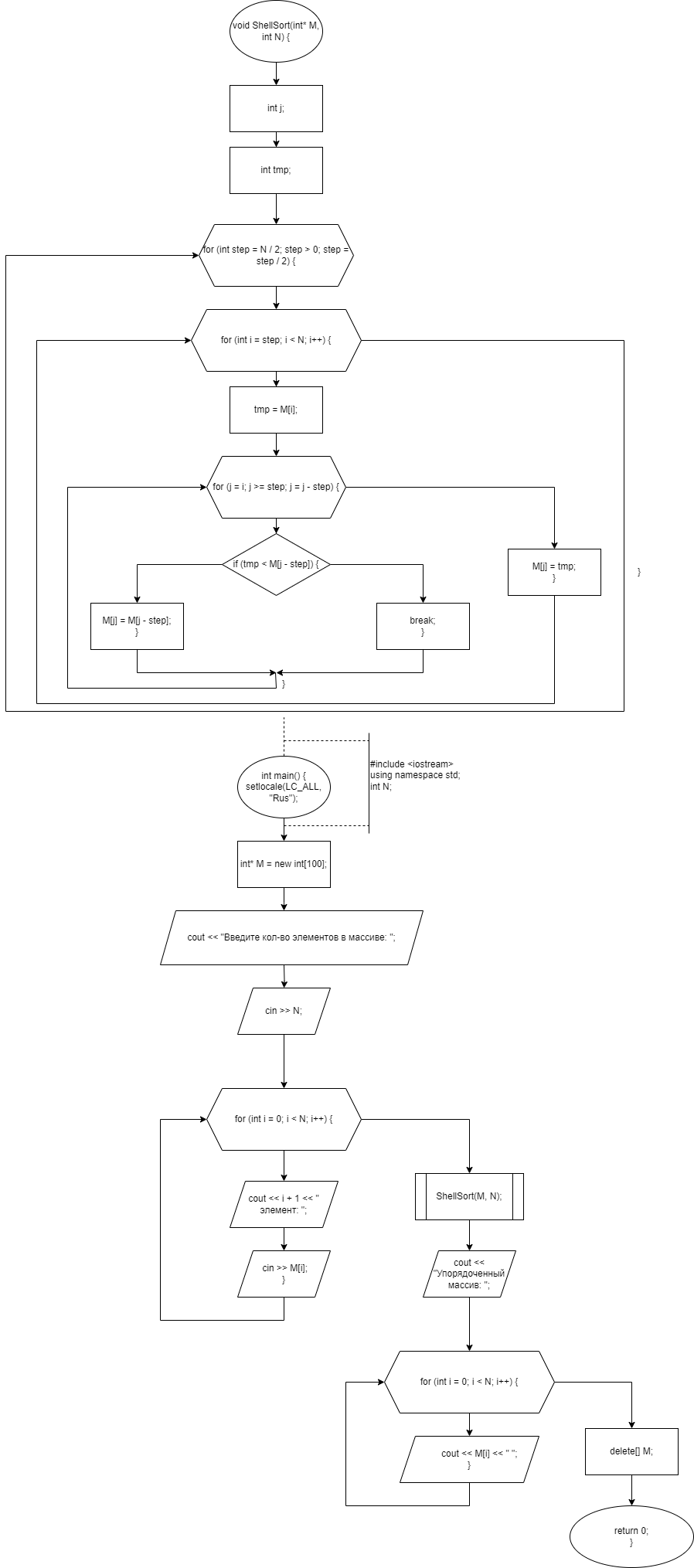
3) Сортировка подсчётом

****

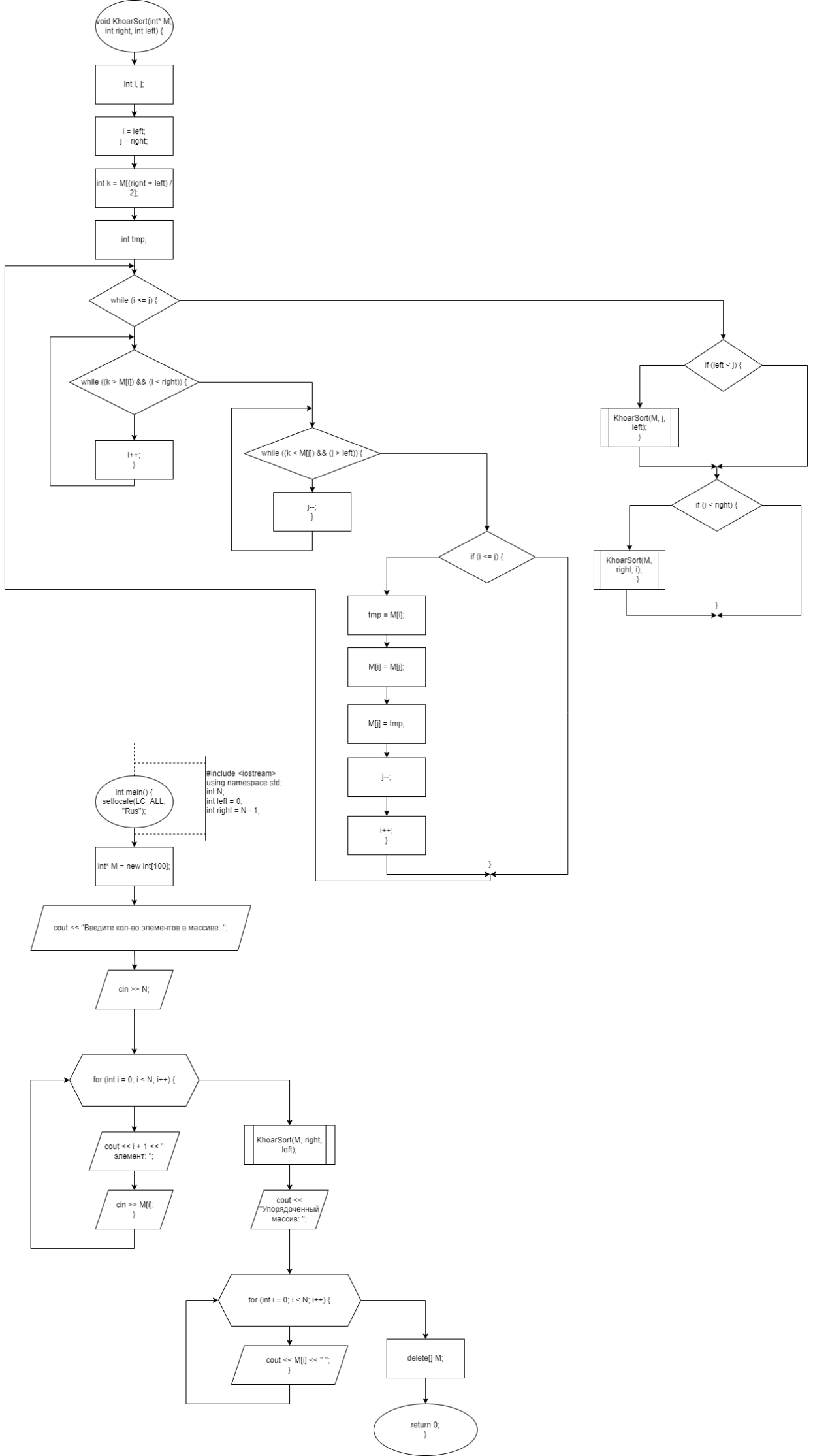
4) Блочная сортировка

****

5) Сортировка Шелла

****

6) Сортировка Хоара

****

**Код на языке C++:**

1) Сортировка слиянием

#include <iostream>

using namespace std;

void Merge(int\* M, int first, int last) {

int middle, start, final, j;

int\* mas = new int[100];

start = first;

middle = (start + last) / 2;

final = middle + 1;

for (j = first; j <= last;j++) {

if ((start <= middle) && ((final >last) || (M[start] < M[final]))) {

mas[j] = M[start];

start++;

}

else {

mas[j] = M[final];

final++;

}

}

for (j = first; j <= last;j++) {

M[j] = mas[j];

}

delete[] mas;

}

void MergeSort(int\* M, int start, int end) {

if (start < end) {

MergeSort(M, start, (start + end) / 2);

MergeSort(M, (start + end) / 2 + 1, end);

Merge(M, start, end);

}

}

int main() {

setlocale(LC\_ALL, "Rus");

int N;

int\* M = new int[100];

cout << "Введите кол-во элементов в массиве: ";

cin >> N;

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

cout << i + 1 << " элемент: ";

cin >> M[i];

}

MergeSort(M, 0, N - 1);

cout << "Упорядоченный массив: ";

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

cout << M[i] << " ";

}

delete[] M;

return 0;

}

2) Быстрая сортировка

#include <iostream>

using namespace std;

void qSort(int\* M, int size) {

int i = 0;

int j = size;

int mid = M[size/ 2];

do {

while (M[i] < mid) {

i++;

}

while (M[j] > mid) {

j--;

}

if (i <= j) {

int tmp = M[i];

M[i] = M[j];

M[j] = tmp;

i++;

j--;

}

} while (i <= j);

if (j > 0) {

qSort(M, j + 1);

}

if (i < size) {

qSort(&M[i], size - i);

}

}

int main() {

setlocale(LC\_ALL, "Rus");

int N;

int\* M = new int[100];

cout << "Введите кол-во элементов в массиве: ";

cin >> N;

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

cout << i + 1 << " элемент: ";

cin >> M[i];

}

qSort(M, N-1);

cout << "Упорядоченный массив: ";

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

cout << M[i] << " ";

}

delete[] M;

return 0;

}

3) Сортировка подсчётом

#include <iostream>

using namespace std;

void CountingSort(int\* M,int N) {

int min = M[0];

int max = M[0];

int p=0;

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

if (M[i] < min) {

min = M[i];

}

if (M[i] > max) {

max = M[i];

}

}

int\* a = new int[N];

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

a[i] = 0;

}

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

a[M[i] - min]++;

}

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

if (a[i] > 0) {

for (int j = 0; j < a[i]; j++) {

M[p] = i + min;

p++;

}

}

}

delete[] a;

}

int main() {

setlocale(LC\_ALL, "Rus");

int N;

int\* M = new int[100];

cout << "Введите кол-во элементов в массиве: ";

cin >> N;

cout << "Изаначальный массив: " << endl;

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

M[i] = rand() %6-3;

cout << M[i] << " ";

}

cout << endl;

CountingSort(M,N);

cout << "Упорядоченный массив: ";

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

cout << M[i] << " ";

}

delete[] M;

return 0;

}

4) Блочная сортировка

#include <iostream>

using namespace std;

void BlochSort(int\* M, int N) {

int min = M[0];

int max = M[0];

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

if (M[i] < min) {

min = M[i];

}

if (M[i] > max) {

max = M[i];

}

}

int count = max - min + 1;

int\*\* c = new int\* [count];

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

c[i] = new int[N];

}

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

int index = (M[i] - min) / count;

c[index][i] = M[i];

}

int index = 0;

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

for (int j = 0; j < N; j++) {

for (int k = j + 1; k < N; k++) {

if (c[i][j] > c[i][k]) {

int tmp = c[i][j];

c[i][j] = c[i][k];

c[i][k] = tmp;

}

}

M[index++] = c[i][j];

}

}

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

delete c[i];

}

delete[] c;

}

int main() {

setlocale(LC\_ALL, "Rus");

int N;

int\* M = new int[100];

cout << "Введите кол-во элементов в массиве: ";

cin >> N;

cout << "Изначальный массив: " << endl;

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

cout << i + 1 << " элемент: ";

cin >> M[i];

}

cout << endl;

BlochSort(M, N);

cout << "Упорядоченный массив: ";

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

cout << M[i] << " ";

}

delete[] M;

return 0;

}

5) Сортировка Шелла

#include <iostream>

using namespace std;

void ShellSort(int\* M, int N) {

int j;

int tmp;

for (int step = N / 2; step > 0; step = step / 2) {

for (int i = step; i < N; i++) {

tmp = M[i];

for (j = i; j >= step; j = j - step) {

if (tmp < M[j - step]) {

M[j] = M[j - step];

}

else

break;

}

M[j] = tmp;

}

}

}

int main() {

setlocale(LC\_ALL, "Rus");

int N;

int\* M = new int[100];

cout << "Введите кол-во элементов в массиве: ";

cin >> N;

cout << "Изначальный массив: " << endl;

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

cout << i + 1 << " элемент: ";

cin >> M[i];

}

cout << endl;

ShellSort(M, N);

cout << "Упорядоченный массив: ";

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

cout << M[i] << " ";

}

delete[] M;

return 0;

}

6) Сортировка Хоара

#include <iostream>

using namespace std;

void KhoarSort(int\* M, int right, int left) {

int i, j;

i = left;

j = right;

int k = M[(right + left) / 2];

int tmp;

while (i <= j) {

while ((k > M[i]) && (i < right)) {

i++;

}

while ((k < M[j]) && (j > left)) {

j--;

}

if (i <= j) {

tmp = M[i];

M[i] = M[j];

M[j] = tmp;

j--;

i++;

}

}

if (left < j) {

KhoarSort(M, j, left);

}

if (i < right) {

KhoarSort(M, right, i);

}

}

int main() {

setlocale(LC\_ALL, "Rus");

int N;

int\* M = new int[100];

cout << "Введите кол-во элементов в массиве: ";

cin >> N;

cout << "Изначальный массив: " << endl;

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

cout << i + 1 << " элемент: ";

cin >> M[i];

}

cout << endl;

int left = 0;

int right = N - 1;

KhoarSort(M, right, left);

cout << "Упорядоченный массив: ";

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

cout << M[i] << " ";

}

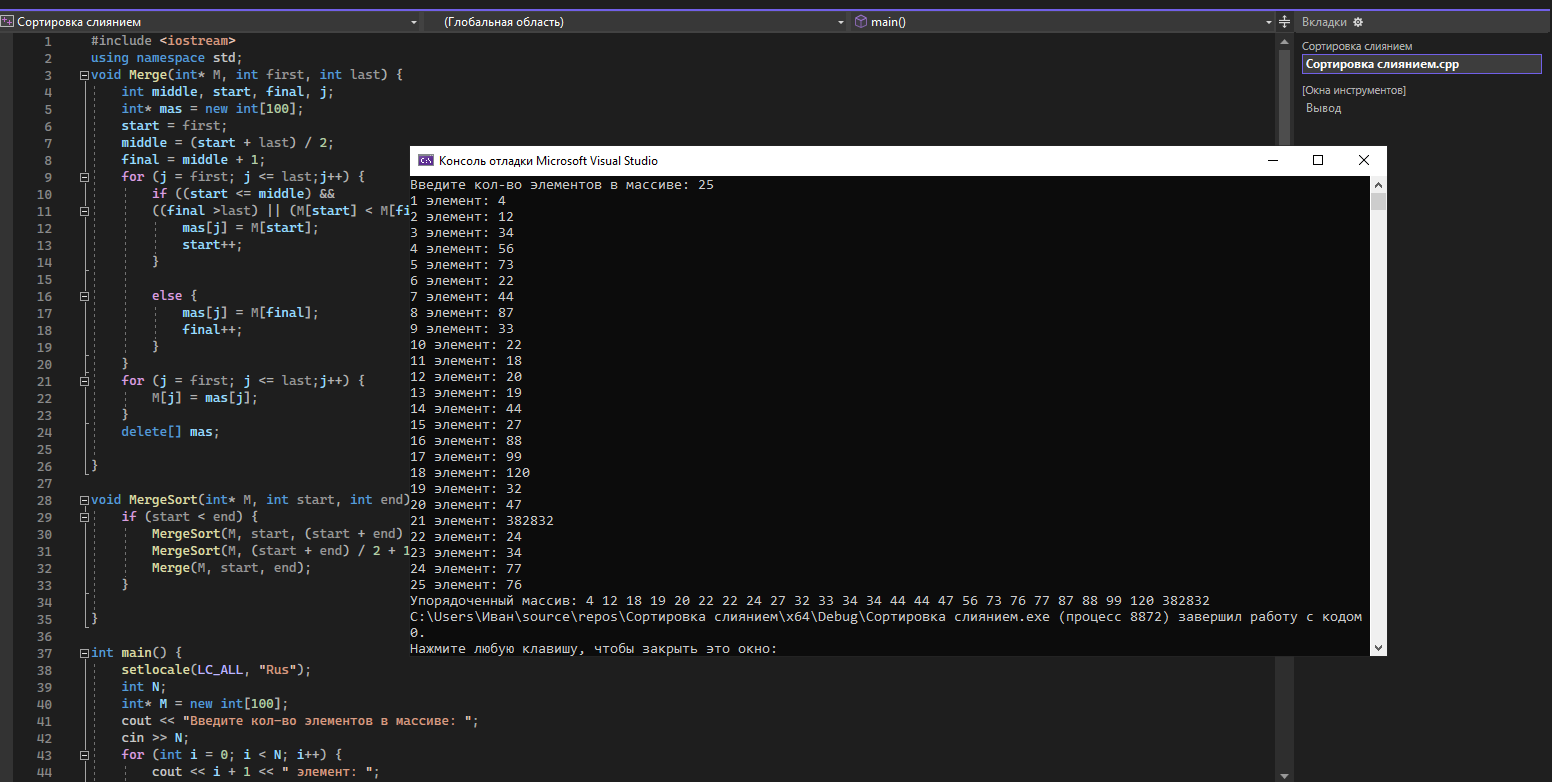
delete[] M;

return 0;

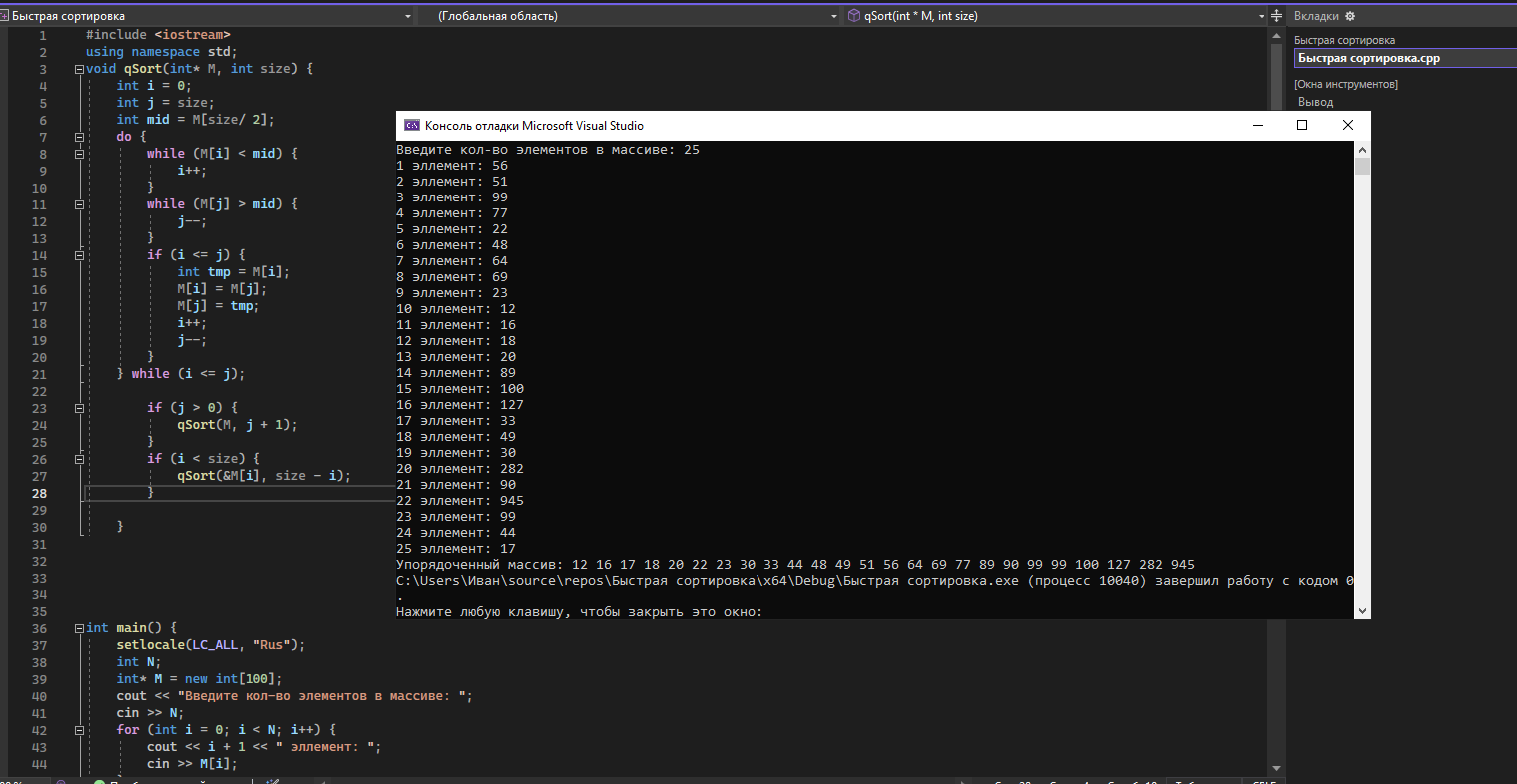
}

**Работа программы:**

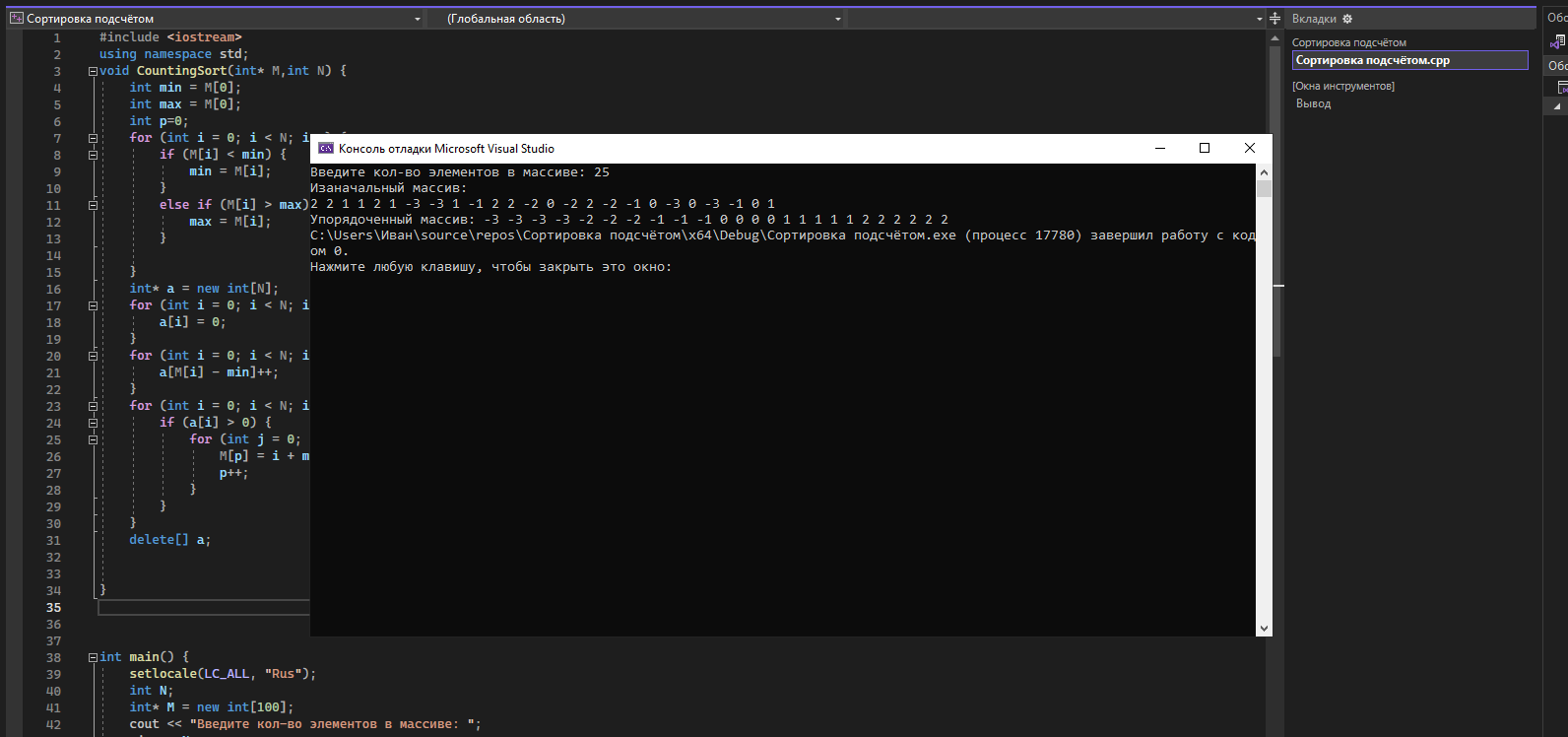
1) Сортировка слиянием



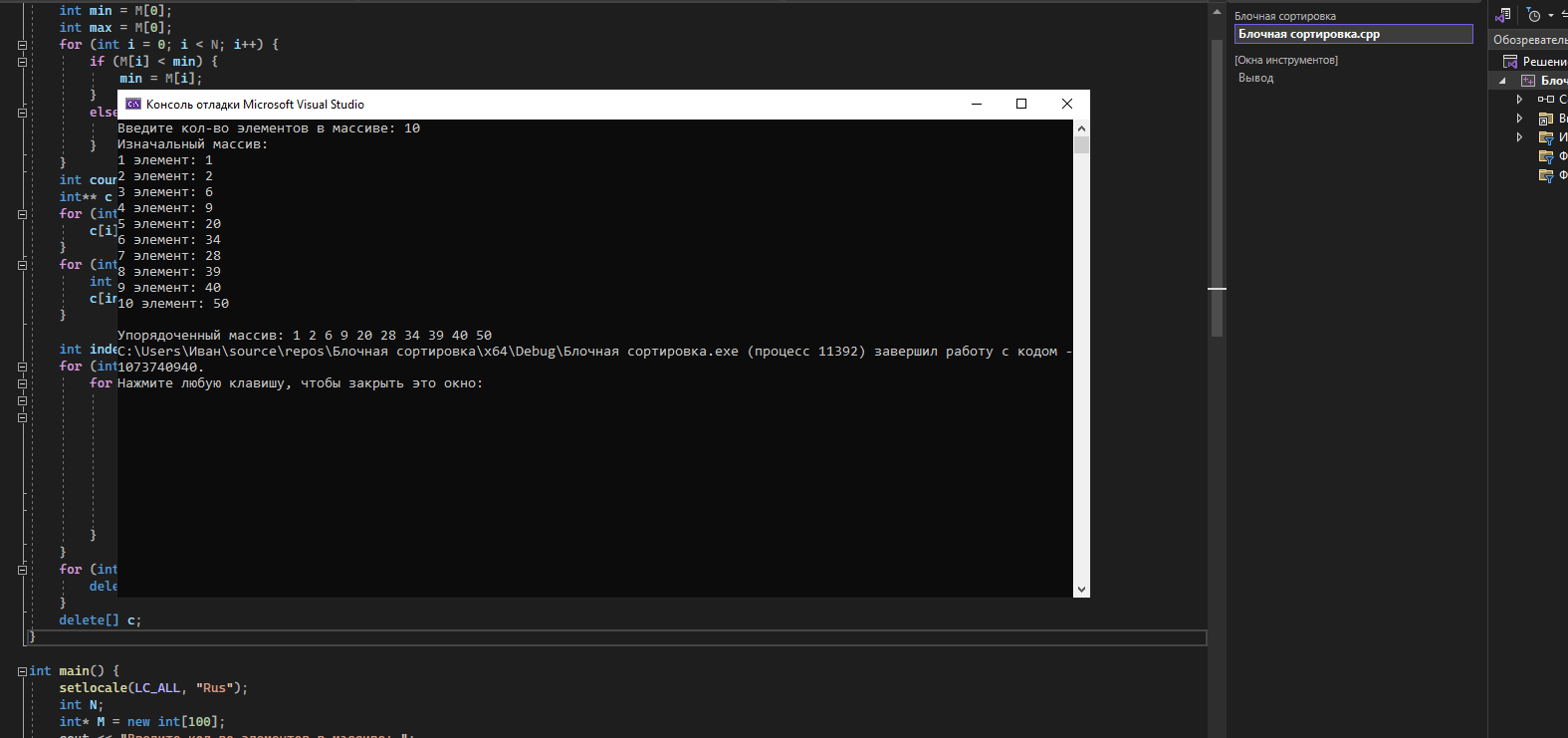
2) Быстрая сортировка



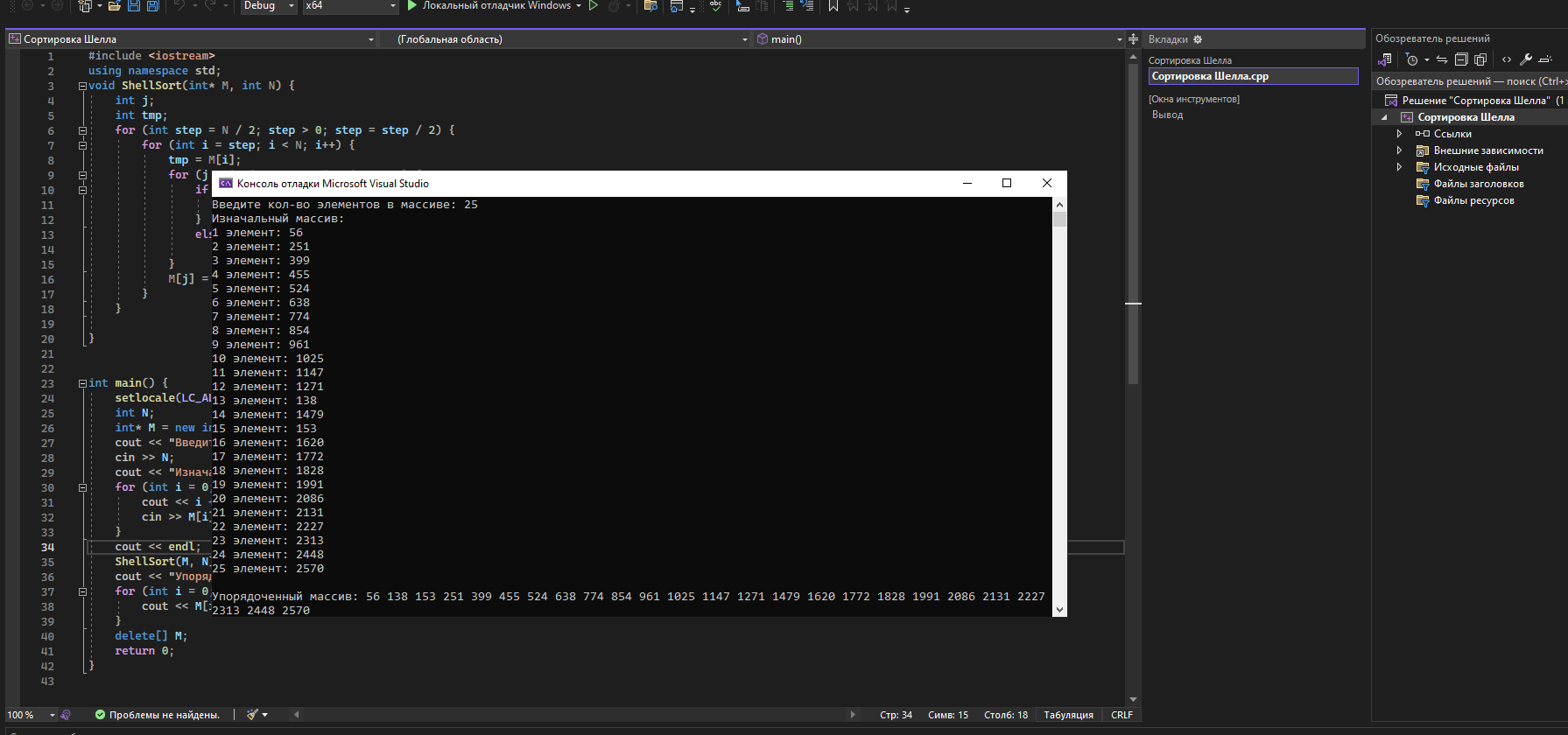
3) Сортировка подсчётом



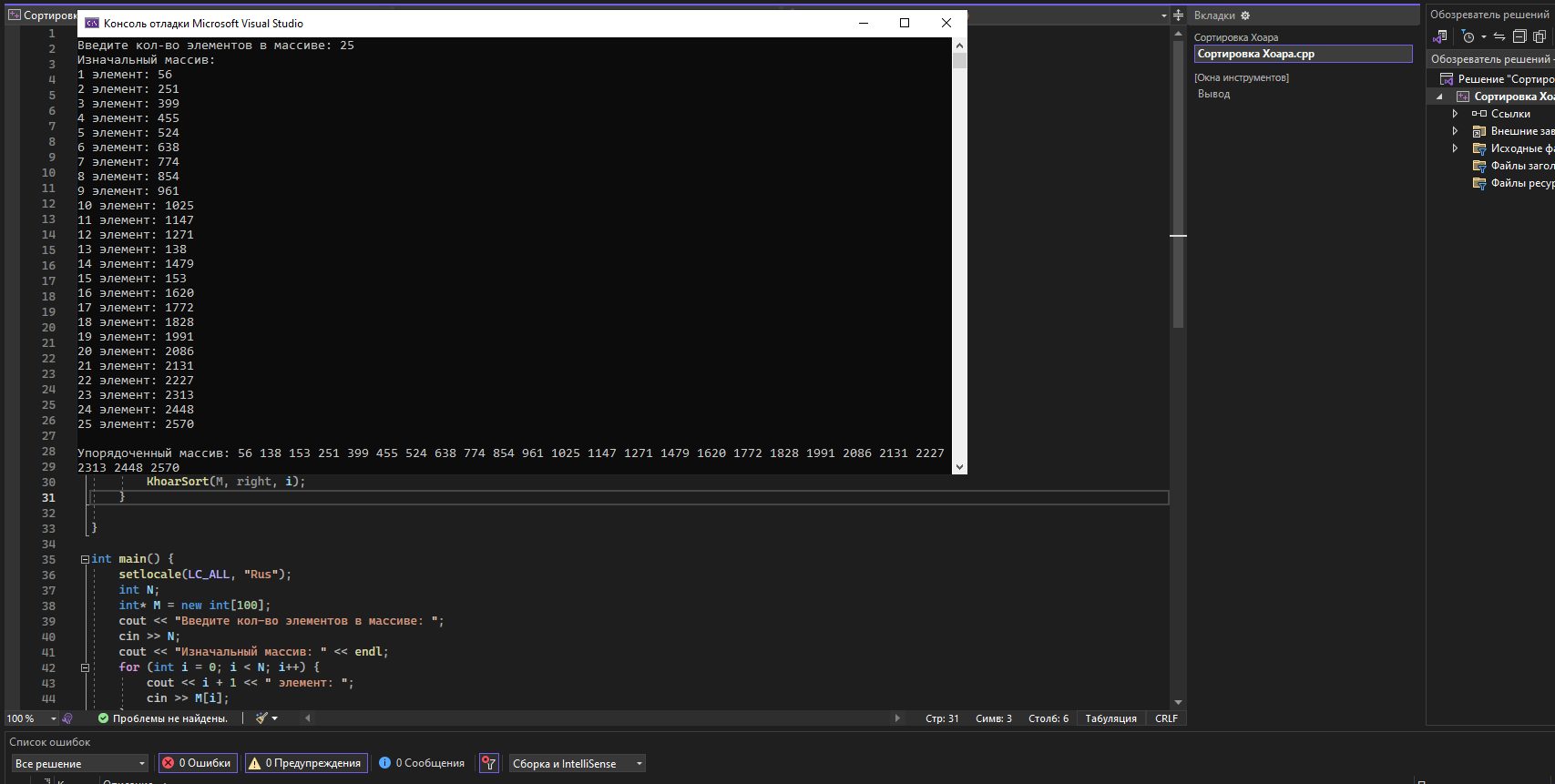
4) Блочная сортировка



5) Сортировка Шелла



6) Сортировка Хоара



**Вывод:** Задача была выполнена. Всё получилось.