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

ПНИПУ

Лабораторная работа  
«Методы Сортировки»

Выполнил:   
студент группы РИС-23-1б   
Мокрушин Никита Дмитриевич

Проверила:   
доцент кафедры ИТАС   
О.А. Полякова

Пермь 2024 г.

**Постановка задачи:**

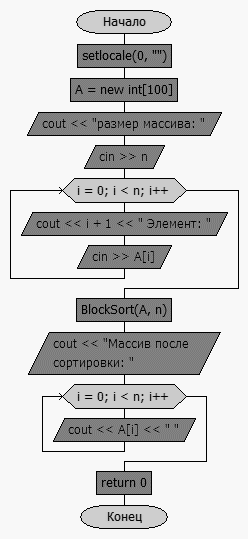
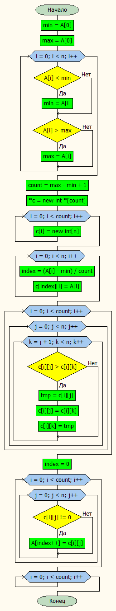
Отсортировать массив, состоящий из 25-ти элементов, используя методы: «Блочная сортировка», «Быстрая сортировка», «Метод выбора», «Метод подсчета», «Метод слияния», «Метод Хоара», «Метод Шелла». Для каждого метода подобрать подходящий под метод массив.

**Анализ задачи:**

* **Используя каждый из перечисленных методов, отсортируем массив из 25-ти элементов**
* **Для метода «Блочная сортировка» содержание массива подберем исходя от метода сортировки используемом в нем**
* **Для метода «Метод слияния» можно использовать любое содержание массива**
* **Для метода «Быстрая сортировка» можно использовать любое содержание массива**
* **Для метода «Сортировка подсчетом» содержание массива должно сожержать:** 
  + - **a) целые числа**
    - **б) небольшой диапазон значений**
    - **в) множество повторяющихся элементов**
    - **г) небольшой диапазон разброс элементов**
* **Для метода «Метод Шелла» можно использовать любое содержание массива**
* **Для метода «Метод Хоара» можно использовать любое содержание массива**
* **Для метода «Сортировка выбором» можно использовать любое содержание массива**

**Блочная сортировка**

**Блок-схема:**

****

**Код на языке с++:**

#include <iostream>

using namespace std;

void BlockSort(int\* A, int n)

{

int min = A[0];

int max = A[0];

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

{

if (A[i]<min)

{

min = A[i];

}

if (A[i] > max)

{

max = A[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 = (A[i] - min) / count;

c[index][i] = A[i];

}

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;

}

}

}

}

int index = 0;

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

{

for (int j = 0; j < n; j++)

{

if (c[i][j] != 0)

{

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

}

}

}

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

{

delete[] c[i];

}

delete c;

}

int main()

{

setlocale(0, "");

int n;

int\* A = new int[100];

cout << "размер массива: ";

cin >> n;

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

{

cout << i + 1 << " Элемент: ";

cin >> A[i];

}

BlockSort(A, n);

cout << "Массив после сортировки: ";

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

{

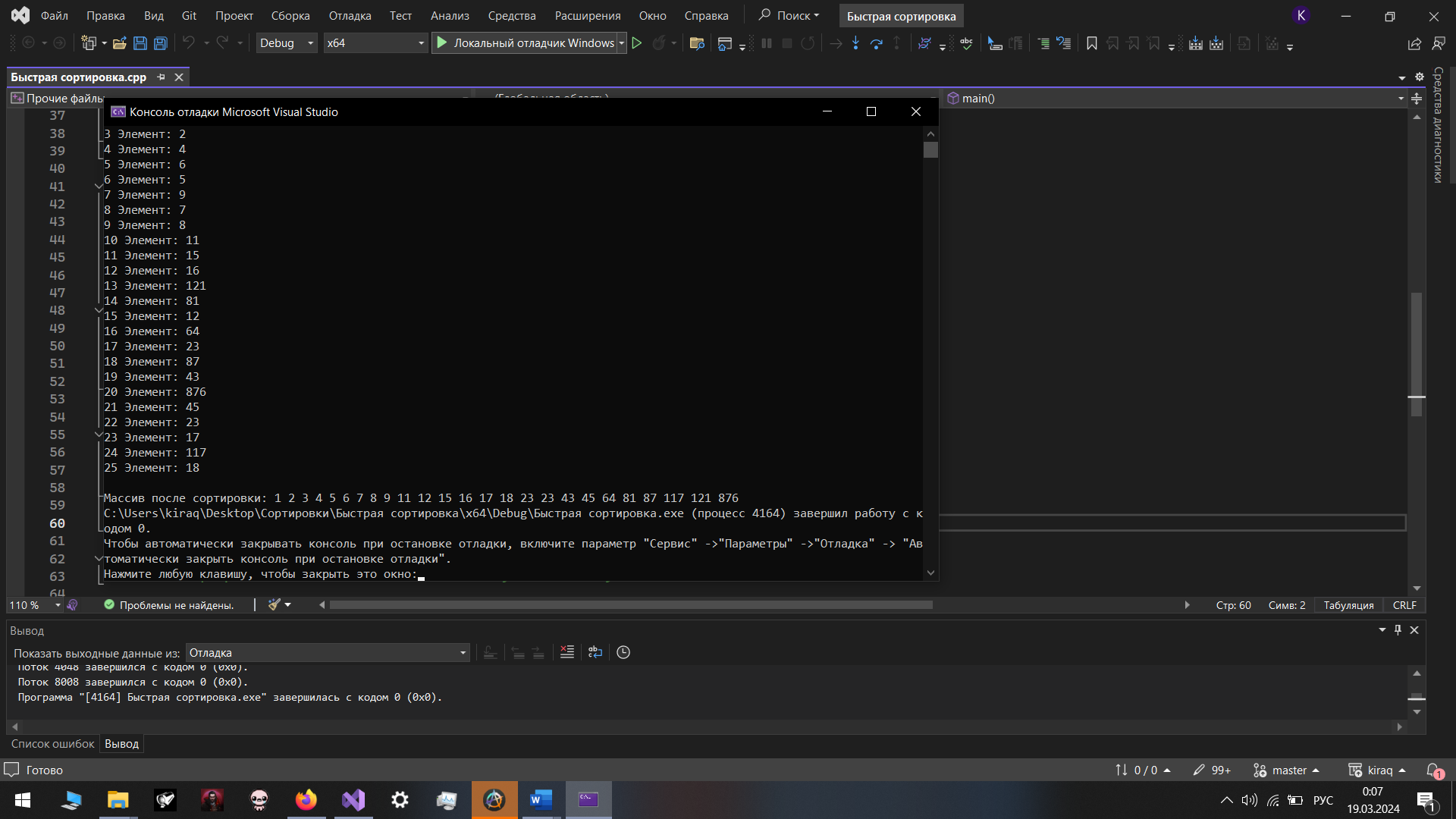
cout << A[i] << " ";

}

return 0;

}

**Пример работы кода:**

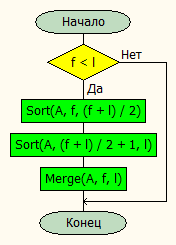
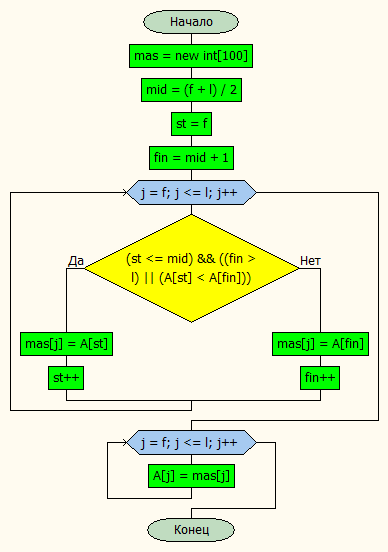
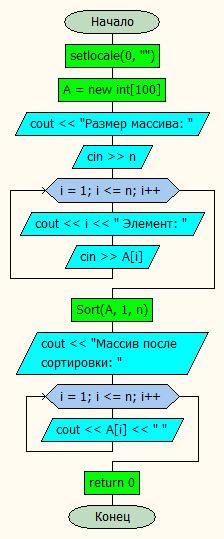


**Вывод:**

Код отработал как задуманно, алгоритм работает исправно

**Метод слияния**

**Блок-схема:**

****

**Код на языке с++:**

#include <iostream>

using namespace std;

void Merge(int\* A, int f, int l)

{

int mid, st, fin, j;

int\* mas = new int[100];

mid = (f + l) / 2;

st = f;

fin = mid + 1;

for (j = f; j <= l; j++)

{

if ((st <= mid) && ((fin > l) || (A[st] < A[fin])))

{

mas[j] = A[st];

st++;

}

else

{

mas[j] = A[fin];

fin++;

}

}

for (j = f; j <= l; j++)

{

A[j] = mas[j];

}

delete[] mas;

}

void Sort(int\* A, int f, int l)

{

if (f < l)

{

Sort(A, f, (f + l) / 2);

Sort(A, (f + l) / 2 + 1, l);

Merge(A, f, l);

}

}

int main()

{

setlocale(0, "");

int i, n;

int\* A = new int[100];

cout << "Размер массива: ";

cin >> n;

for (i = 1; i <= n; i++)

{

cout << i << " Элемент: ";

cin >> A[i];

}

Sort(A, 1, n);

cout << "Массив после сортировки: ";

for (i = 1; i <= n; i++)

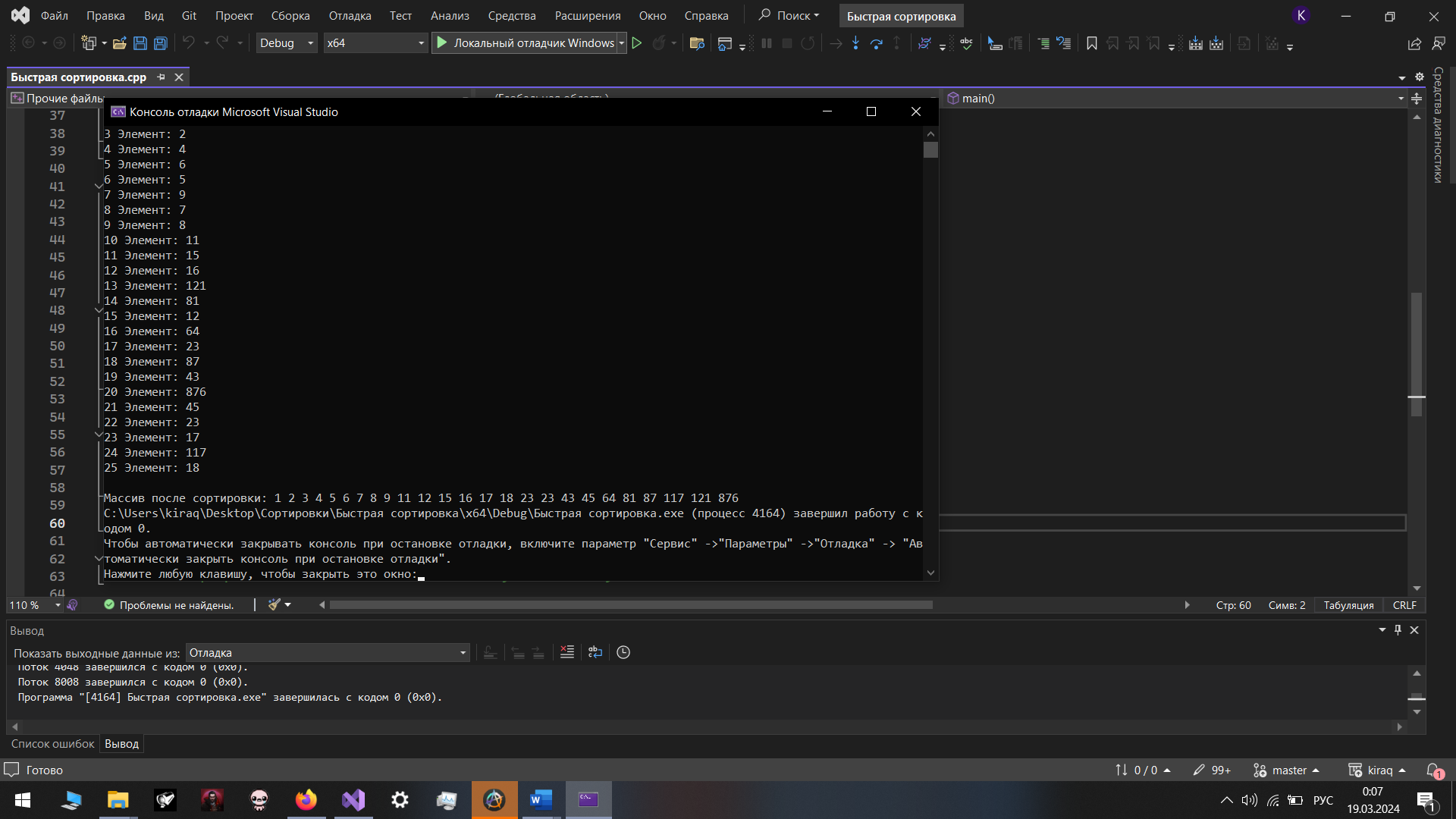
{

cout << A[i] << " ";

}

return 0;

}**Пример работы кода:**

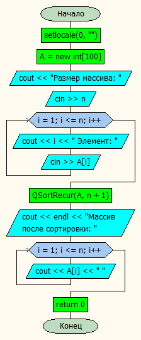
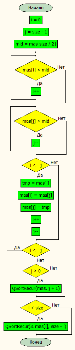


**Вывод:**

Код отработал как задуманно, алгоритм работает исправно

**Быстрая сортировка**

**Блок-схема:**

****

**Код на языке с++:**

#include <iostream>

using namespace std;

void QSortRecur(int\* mas, int size)

{

int i = 0;

int j = size - 1;

int mid = mas[size / 2];

do

{

while (mas[i]<mid)

{

i++;

}

while (mas[j]>mid)

{

j--;

}

if (i <= j)

{

int tmp = mas[i];

mas[i] = mas[j];

mas[j] = tmp;

i++;

j--;

}

} while (i <= j);

if (j > 0)

{

QSortRecur(mas, j + 1);

}

if (i < size)

{

QSortRecur(&mas[i], size - i);

}

}

int main()

{

setlocale(0, "");

int i, n;

int\* A = new int[100];

cout << "Размер массива: ";

cin >> n;

for (i = 1; i <= n; i++)

{

cout << i << " Элемент: ";

cin >> A[i];

}

QSortRecur(A, n + 1);

cout << endl << "Массив после сортировки: ";

for (i = 1; i <= n; i++)

{

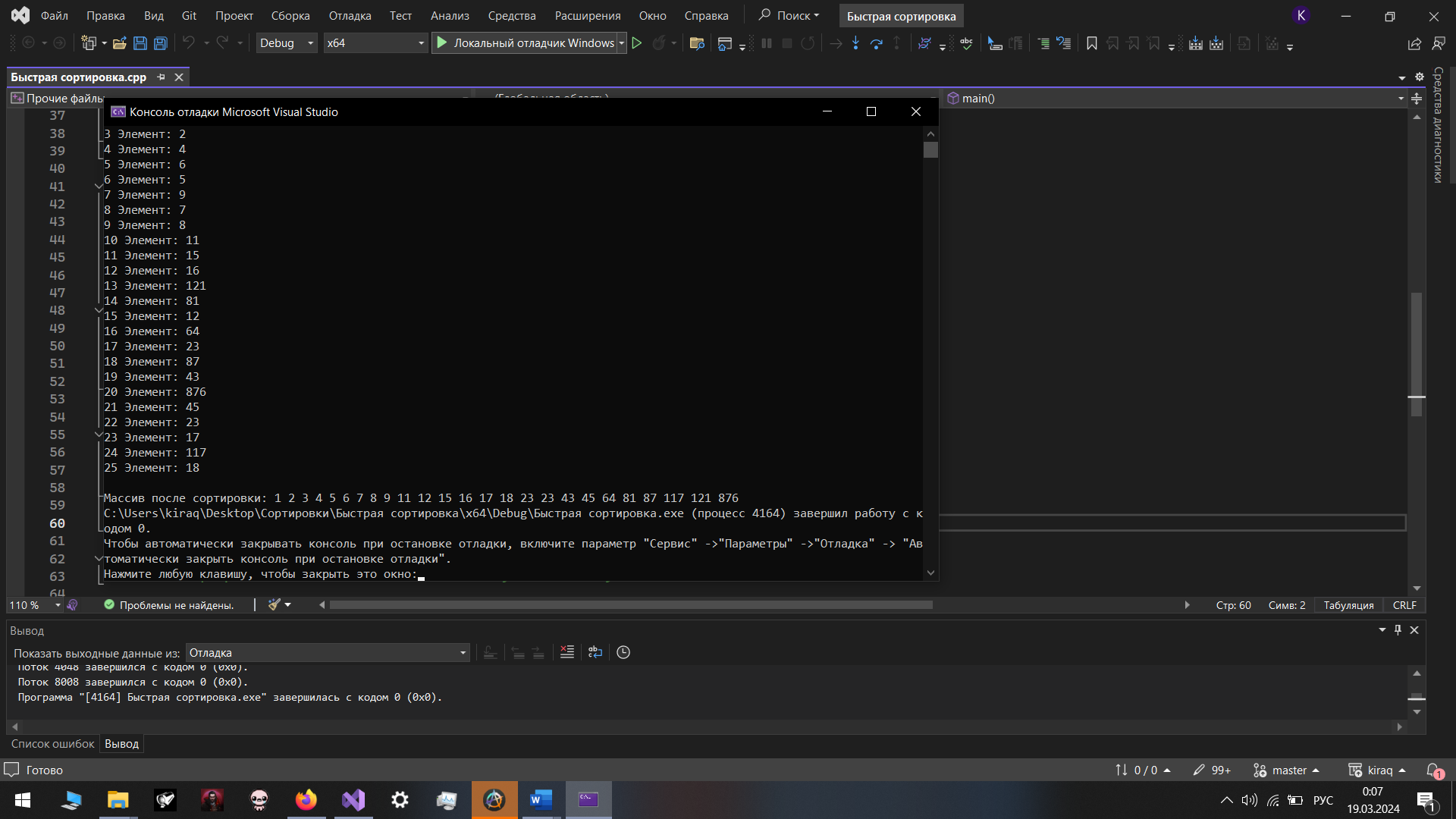
cout << A[i] << " ";

}

return 0;

}

**Пример работы кода:**

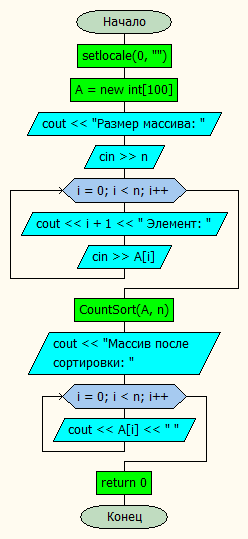
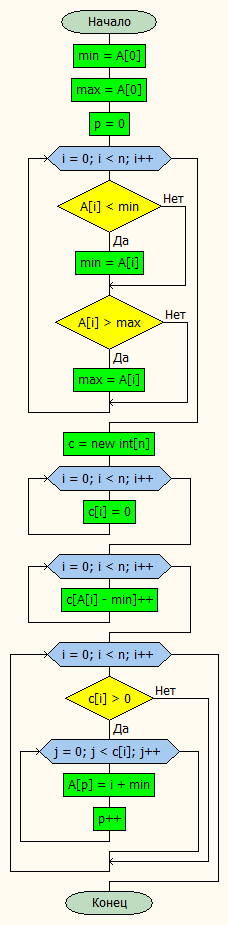


**Вывод:**

Код отработал как задуманно, алгоритм работает исправно

**Сортировка Подсчетом**

**Блок-схема:**

****

**Код на языке с++:**

#include <iostream>

using namespace std;

void CountSort(int\* A, int n)

{

int min = A[0];

int max = A[0];

int p = 0;

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

{

if (A[i] < min)

{

min = A[i];

}

if (A[i] > max)

{

max = A[i];

}

}

int\* c = new int[n];

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

{

c[i] = 0;

}

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

{

c[A[i] - min]++;

}

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

{

if (c[i]>0)

{

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

{

A[p] = i + min;

p++;

}

}

}

delete[] c;

}

int main()

{

int n;

setlocale(0, "");

int\* A = new int[100];

cout << "Размер массива: ";

cin >> n;

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

{

cout << i + 1 << " Элемент: ";

cin >> A[i];

}

CountSort(A, n);

cout << "Массив после сортировки: ";

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

{

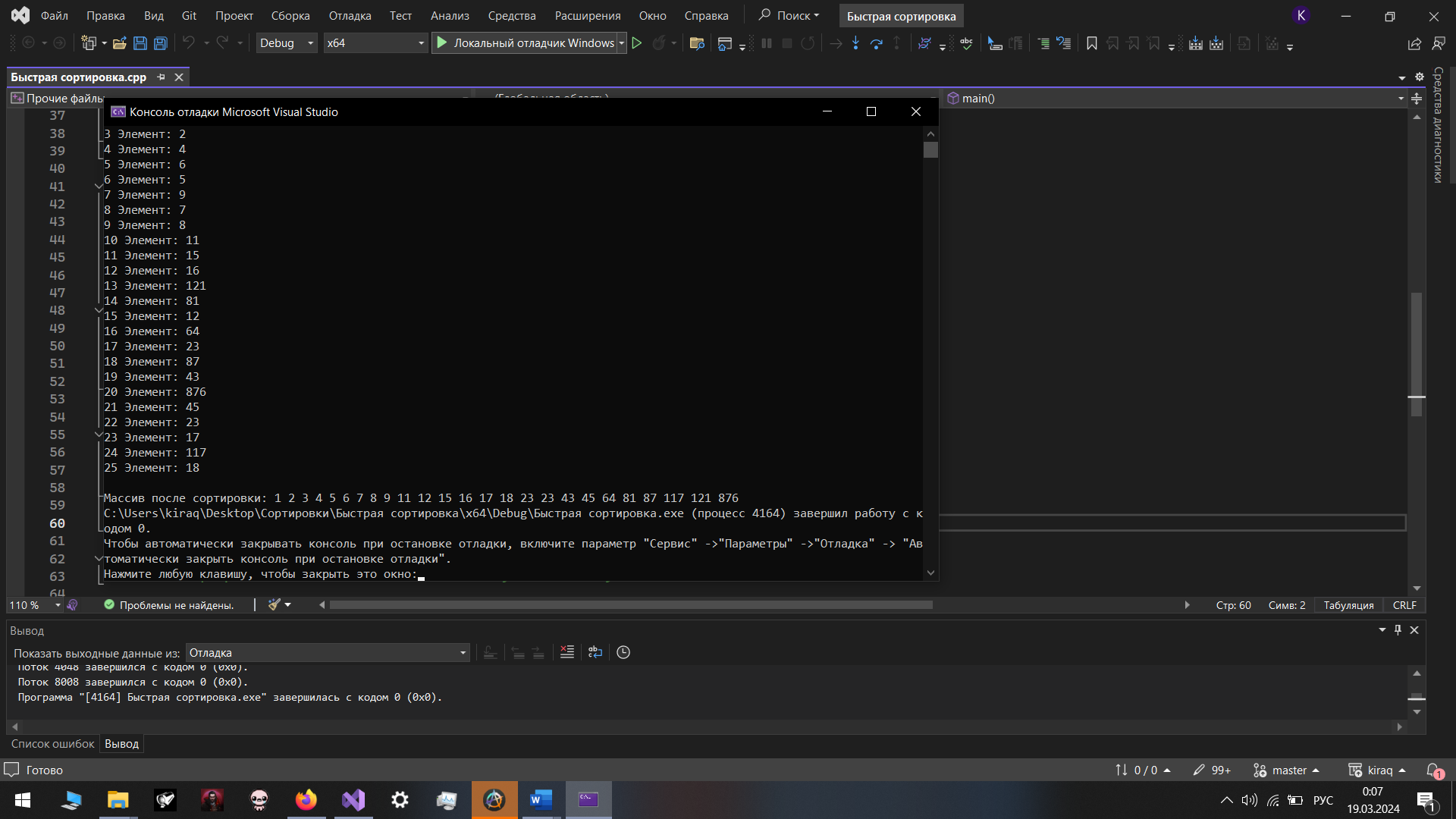
cout << A[i] << " ";

}

return 0;

}

**Пример работы кода:**

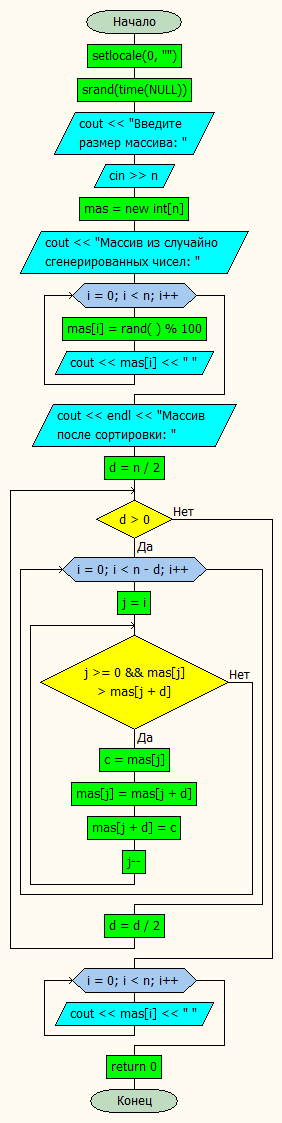


**Вывод:**

Код отработал как задуманно, алгоритм работает исправно

**Метод Шелла**

**Блок-схема:**



**Код на языке с++:**

#include <iostream>

#include <ctime>

using namespace std;

int main()

{

int i, j, n, d, c;

setlocale(0, "");

srand(time(NULL));

cout << "Введите размер массива: ";

cin >> n;

int\* mas = new int[n];

cout << "Массив из случайно сгенерированных чисел: ";

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

{

mas[i] = rand() % 100;

cout << mas[i] << " ";

}

cout << endl << "Массив после сортировки: ";

d = n / 2;

while (d > 0)

{

for (int i = 0; i < n-d; i++)

{

j = i;

while (j >= 0 && mas[j] > mas[j+d])

{

c = mas[j];

mas[j] = mas[j + d];

mas[j + d] = c;

j--;

}

}

d = d / 2;

}

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

{

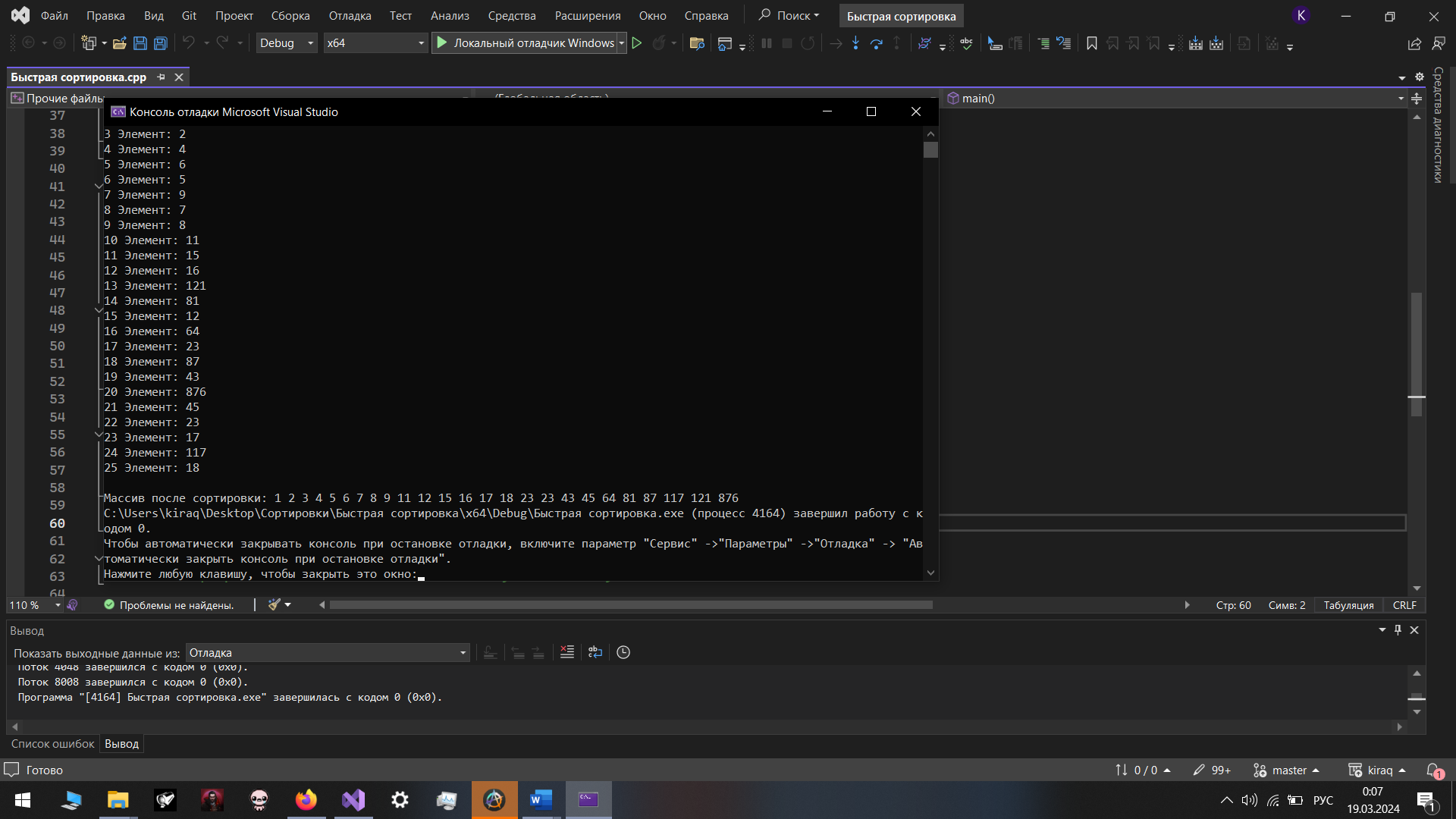
cout << mas[i] << " ";

}

return 0;

}

**Пример работы кода:**

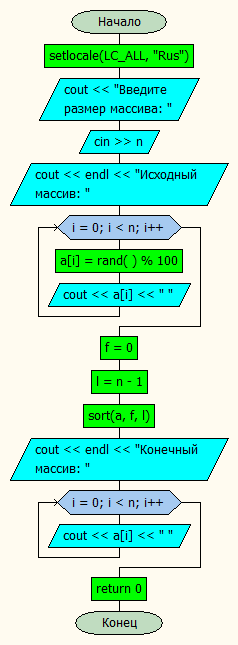
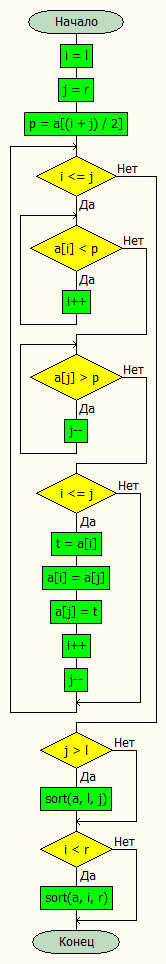


**Вывод:**

Код отработал как задуманно, алгоритм работает исправно

**Метод Хоара**

**Блок-схема:**

****

**Код на языке с++:**

#include <iostream>

using namespace std;

void sort(int\* a, int l, int r)

{

int i, j, p, t;

i = l;

j = r;

p = a[(i + j) / 2];

while (i <= j)

{

while (a[i] < p)

i++;

while (a[j] > p)

j--;

if (i <= j)

{

t = a[i];

a[i] = a[j];

a[j] = t;

i++;

j--;

}

}

if (j > l)

sort(a, l, j);

if (i < r)

sort(a, i, r);

}

int main()

{

int f, l, a[100], n;

setlocale(LC\_ALL, "Rus");

cout << "Введите размер массива: ";

cin >> n;

cout << endl << "Исходный массив: ";

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

{

a[i] = rand() % 100;

cout << a[i] << " ";

}

f = 0;

l = n - 1;

sort(a, f, l);

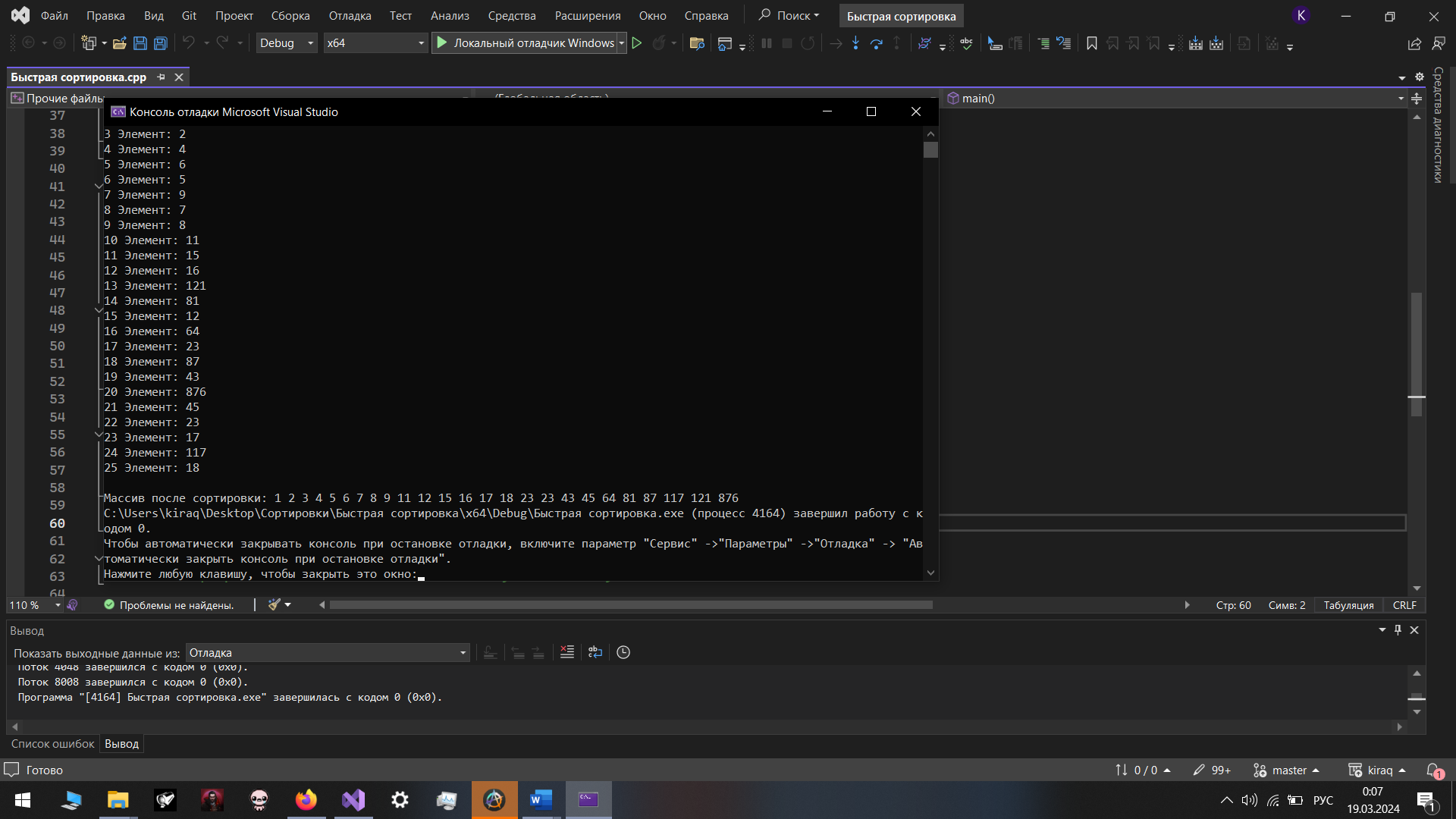
cout << endl << "Конечный массив: ";

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

cout << a[i] << " ";

return 0;

}**Пример работы кода:**

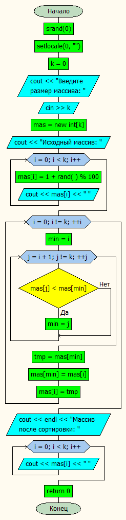


**Вывод:**

Код отработал как задуманно, алгоритм работает исправно

**Сортировка выбором**

**Блок-схема:**



**Код на языке с++:**

#include <iostream>

using namespace std;

int main()

{

srand(0);

setlocale(0, "");

int k = 0;

cout << "Введите размер массива: ";

cin >> k;

int\* mas = new int[k];

cout << "Исходный массив: ";

for (int i = 0; i < k; i++)

{

mas[i] = 1 + rand() % 100;

cout << mas[i] << " ";

}

for (int i = 0; i != k; ++i)

{

int min = i;

for (int j = i + 1; j != k; ++j)

{

if (mas[j] < mas[min])

{

min = j;

}

}

int tmp = mas[min];

mas[min] = mas[i];

mas[i] = tmp;

}

cout << endl << "Массив после сортировки: ";

for (int i = 0; i < k; i++)

{

cout << mas[i] << " ";

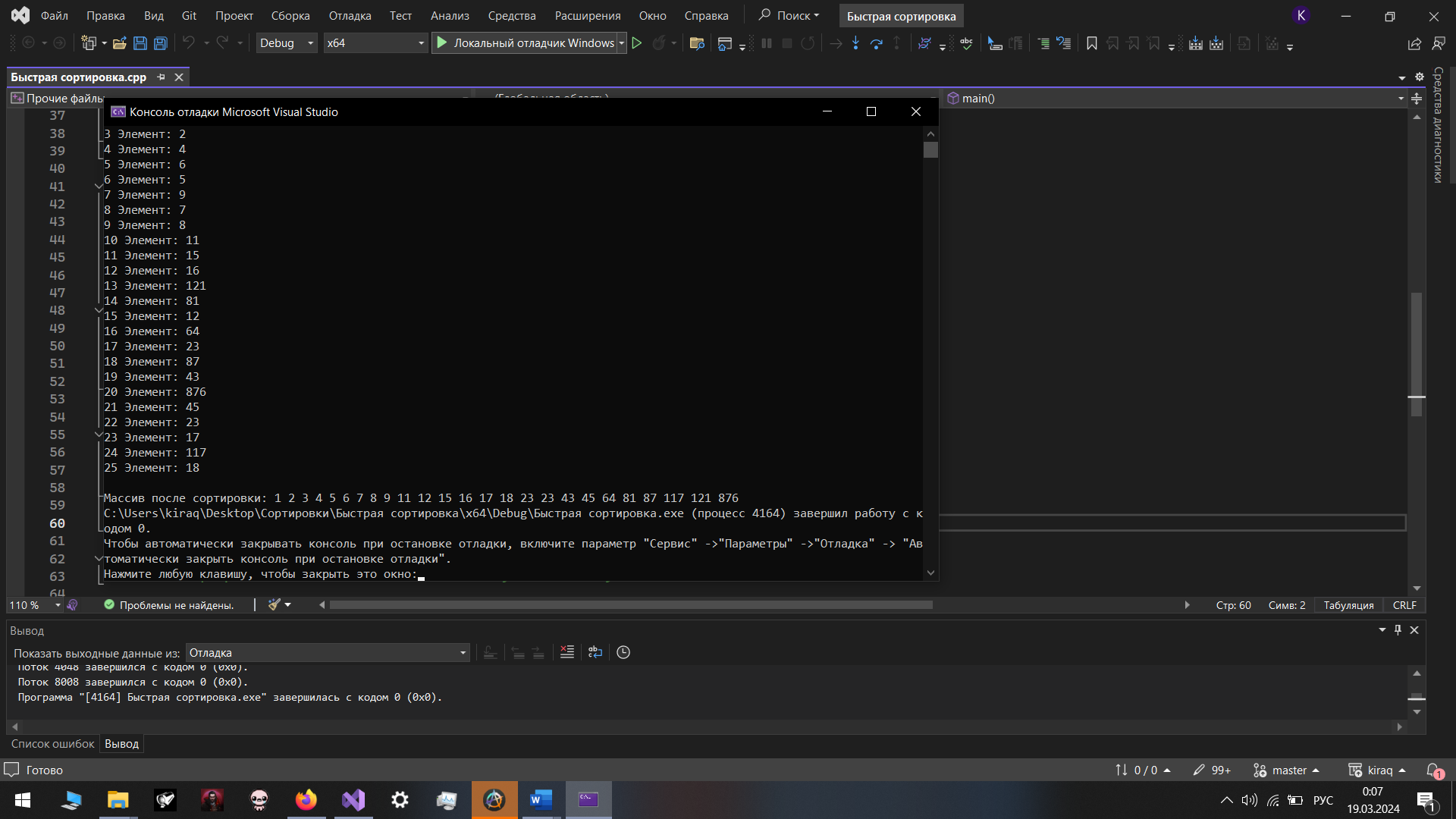
}

delete[] mas;

return 0;

}

**Пример работы кода:**



**Вывод:**

Код отработал как задуманно, алгоритм работает исправно