**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Основы программирования»

Отчет по лабораторной работе № 4

«Нахождение корней нелинейного уравнения»

| Выполнил: Чернев Николай Андреевич |  | Проверил: |
| --- | --- | --- |
| студент группы ИУ5-14Б |  | преподаватель каф. ИУ5 |
|  |  |  |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2023 г.

Задание

Отсортировать числовой массив методом выбора максимального (минимального) элемента и методом пузырькового всплытия. По окончании сортировки вывести отсортированный массив и количество сделанных сравнений и перестановок элементов.

Сравнить быстродействие алгоритмов, которое определяется числом сравнений и перестановок, для исходного не отсортированного массива и для исходного массива, отсортированного в прямом и обратном порядке.

Исследовать зависимость быстродействия от размера массива. Возможность изменения длины массива реализуйте с помощью динамического массива, а для его инициализации используйте датчик случайных чисел (см. Приложение 1). Результаты исследования выведите в виде отформатированной таблицы.

Разработка алгоритма

Описание переменных

Main:

* int n - длина первого массива
* int \*arr - первый массив (для сортировки выбором)
* int \*arr2 - копия первого массива (для сортировки пузырьком)
* int i - счетчик цикла для заполнения массивов
* int \*arr5, \*arr5\_2, \*arr50, \*arr50\_2, \*arr500, \*arr500\_2 - массивы размера 5, 50, 50 для сортировки выбором (без приписки) и сортировки пузырьком (приписка \_2)

Сортировка методом выбора мин/макс:

* int \*a - сортируемый массив
* int N - длина сортируемого массива
* int k - 1 если надо отсортировать по неубыванию, -1 если по невозрастанию
* int count\_perm - кол-во перестановок
* int count\_comp - кол-во сравнений
* int min\_ind - индекс мин элемента
* int i, j - счетчики циклов

Сортировка пузырьком:

* int \*a - сортируемый массив
* int N - длина сортируемого массива
* int k: 1 если надо отсортировать по неубыванию, -1 если по невозрастанию
* int count\_perm - кол-во перестановок
* int count\_comp - кол-во сравнений
* int cur\_cnt - кол-во сравнений на текущем проходе
* int i, j - счетчики циклов

iRadom:

* int a, b - левая/правая границы диапазона рандомного числа

print:

* int \*a - массив
* int N - длина массива

Описание функций

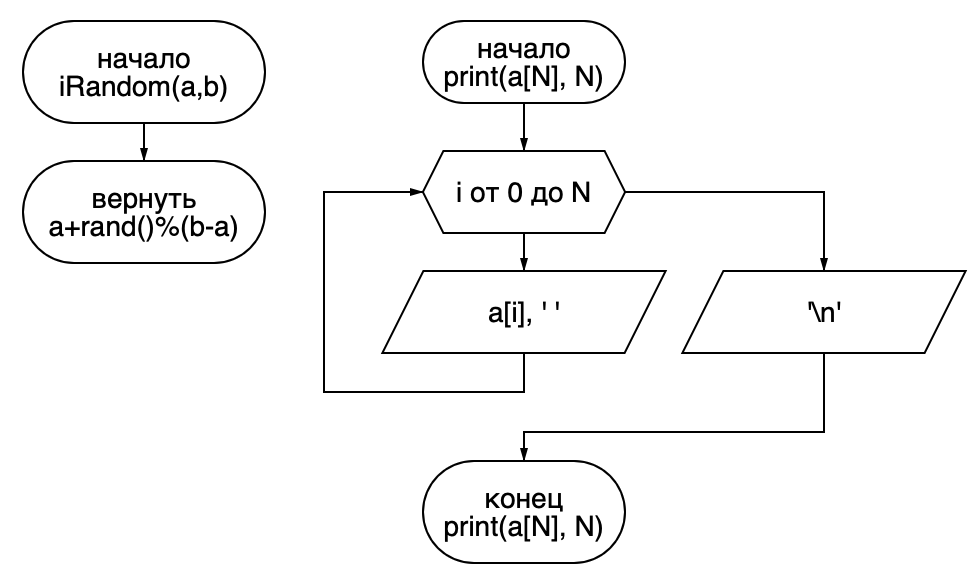
int iRandom(int a, int b) - принимает границы a и b и возвращает рандомное число из промежутка от a до b

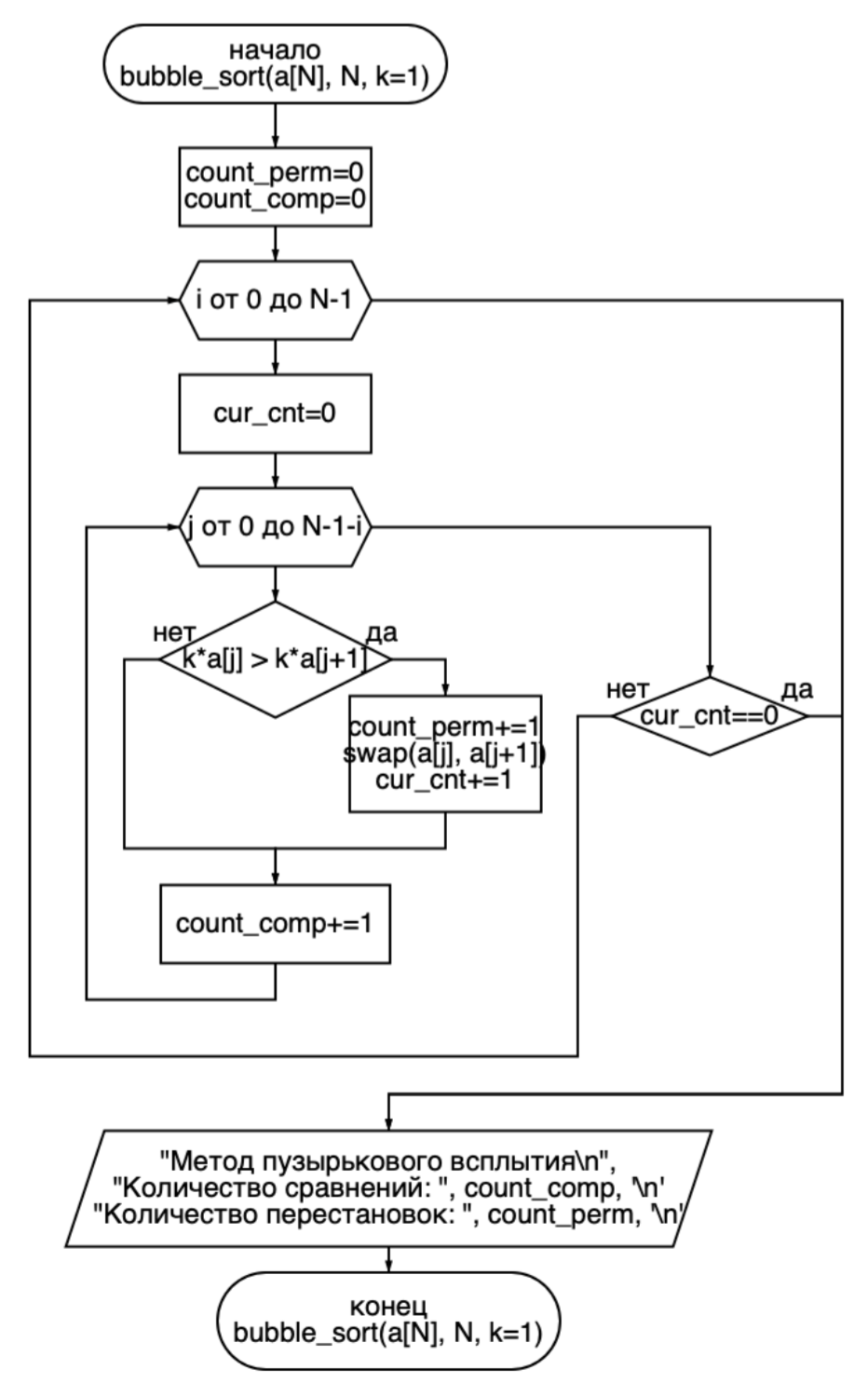
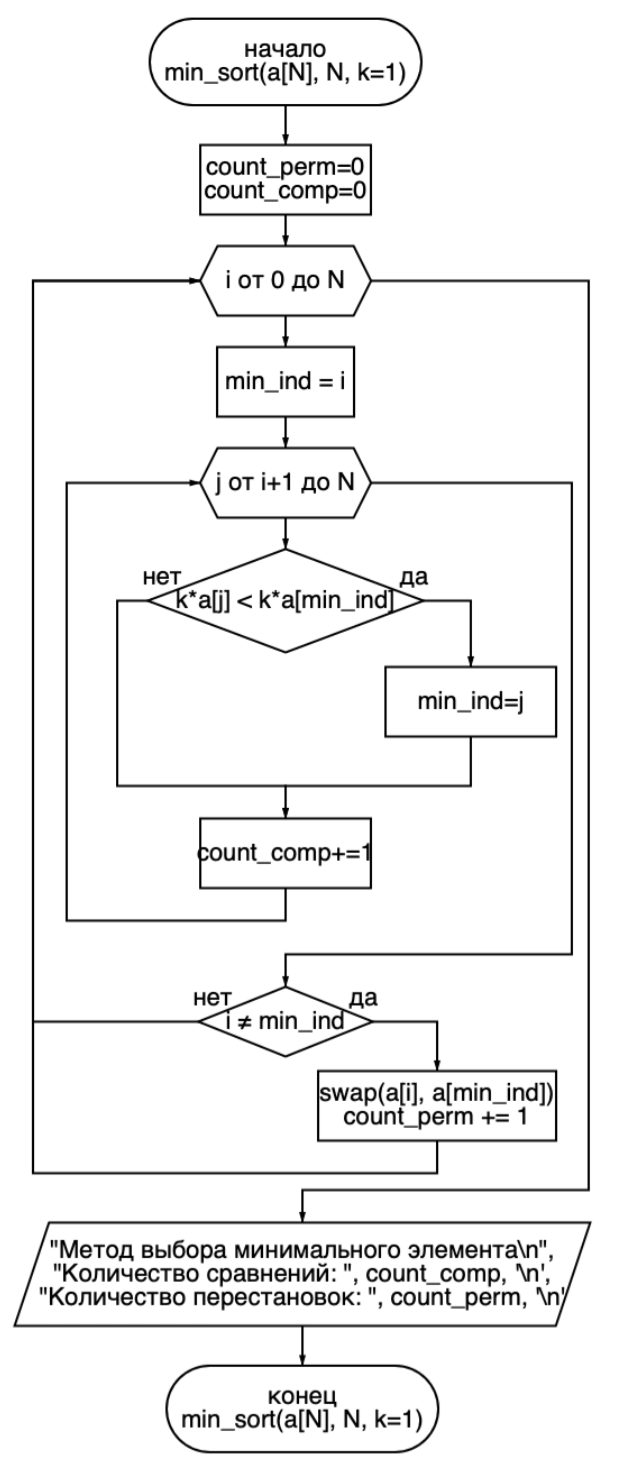
void print(int\* a, int N) - принимает массив а и его длину N и выводит этот массив на экран, ничего не возвращает

void min\_sort(int\* a, int N, int k = 1) - принимает массив a, его длину N, k (отвечает за выбор того, в каком порядке сортировать - по неубыванию или невозрастанию) - сортирует этот массив методом выбора минимума и выводит результат (кол-во сравнений и перестановок), ничего не возвращает

void bubble\_sort(int\* a, int N, int k = 1) - принимает массив a, его длину N, k (аналогично min\_sort) - сортирует этот массив методом пузырька и выводит результат (кол-во сравнений и перестановок), ничего не возвращает

Блок-схема





Код программы

#include <time.h>

#include <stdlib.h>

#include <iostream>

using namespace std;

int iRandom(int a, int b){

return a + rand() % (b - a);

}

void print(int\* a, int N){

for (int i = 0; i < N; i++) cout << a[i] << ' ';

cout << '\n';

}

void min\_sort(int\* a, int N, int k = 1){

int count\_perm = 0, count\_comp = 0, min\_ind;

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

min\_ind = i;

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

if (k \* a[j] < k \* a[min\_ind]) {

min\_ind = j;

}

count\_comp++;

}

if (i != min\_ind) {

swap(a[i], a[min\_ind]);

count\_perm++;

}

}

cout << "Метод выбора минимального элемента\n";

cout << "Количество сравнений: " << count\_comp << '\n';

cout << "Количество перестановок: " << count\_perm << '\n';

}

void bubble\_sort(int\* a, int N, int k = 1){

int count\_perm = 0, count\_comp = 0, cur\_cnt;

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

cur\_cnt = 0;

for (int j = 0; j < N - 1 - i; j++){

if (k \* a[j] > k \* a[j + 1]){

count\_perm++;

swap(a[j], a[j + 1]);

cur\_cnt++;

}

count\_comp++;

}

if (cur\_cnt == 0) break;

}

cout << "Метод пузырькового всплытия\n";

cout << "Количество сравнений: " << count\_comp << '\n';

cout << "Количество перестановок: " << count\_perm << '\n';

}

int main(){

int N;

cout << "Введите N - длину массива\n";

cin >> N;

int \*arr = new int[N];

int \*arr2 = new int[N];

srand(time(0));

rand();

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

arr[i] = iRandom(1, 100);

arr2[i] = arr[i];

}

print(arr, N);

min\_sort(arr, N);

print(arr, N);

cout << '\n';

bubble\_sort(arr2, N);

print(arr2, N);

cout << '\n';

print(arr, N);

min\_sort(arr, N);

print(arr, N);

cout << '\n';

bubble\_sort(arr2, N);

print(arr2, N);

cout << '\n';

print(arr, N);

min\_sort(arr, N, -1);

print(arr, N);

cout << '\n';

bubble\_sort(arr2, N, -1);

print(arr2, N);

cout << '\n';

delete[] arr;

delete[] arr2;

int \*arr5 = new int[5];

int \*arr5\_2 = new int[5];

int \*arr50 = new int[50];

int \*arr50\_2 = new int[50];

int \*arr500 = new int[500];

int \*arr500\_2 = new int[500];

for (int i = 0; i < 5; i++) {

arr5[i] = iRandom(1, 100);

arr5\_2[i] = arr5[i];

}

for (int i = 0; i < 50; i++) {

arr50[i] = iRandom(1, 100);

arr50\_2[i] = arr50[i];

}

for (int i = 0; i < 500; i++) {

arr500[i] = iRandom(1, 100);

arr500\_2[i] = arr500[i];

}

cout << "5 элементов: ";

min\_sort(arr5, 5);

cout << "\n5 элементов: ";

bubble\_sort(arr5\_2, 5);

cout << '\n';

cout << "50 элементов: ";

min\_sort(arr50, 50);

cout << "\n50 элементов: ";

bubble\_sort(arr50\_2, 50);

cout << '\n';

cout << "500 элементов: ";

min\_sort(arr500, 500);

cout << "\n500 элементов: ";

bubble\_sort(arr500\_2, 500);

cout << '\n';

delete[] arr5;

delete[] arr50;

delete[] arr500;

delete[] arr5\_2;

delete[] arr50\_2;

delete[] arr500\_2;

return 0;

}

Анализ результатов

