

**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(МИИТ)**

|  |
| --- |
| **ИНСТИТУТ ТРАНСПОРТНОЙ ТЕХНИКИ И СИСТЕМ УПРАВЛЕНИЯ**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Кафедра «Управление и защита информации»

**Лабораторная работа №5**

**«Работа с подмассивами»**

по дисциплине

**«Методы программирования»**

**Выполнил:** ст. гр. ТКИ-311,

Виноградова В.Л.

**Проверил:**

к.т.н., доц. Логинова Л.Н.,

к.т.н., доц. Сафронов А.И.

**Москва – 2022 г.**

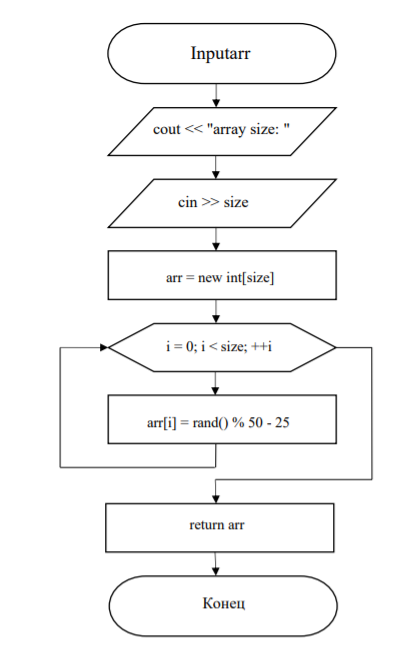
**Цель работы:**

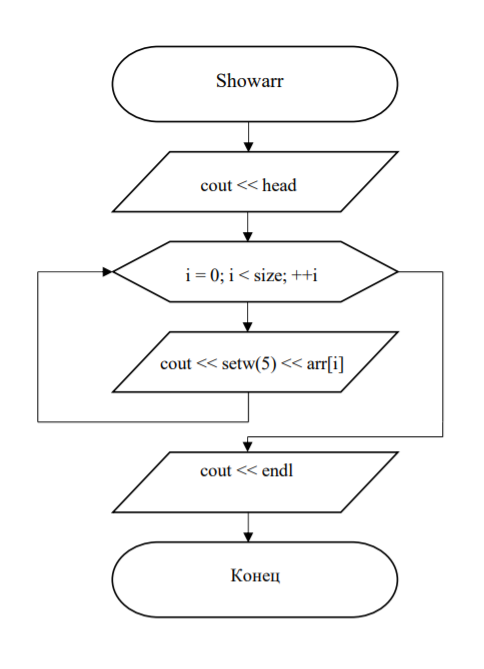
Изучение алгоритма поиска наибольшего подмассива.

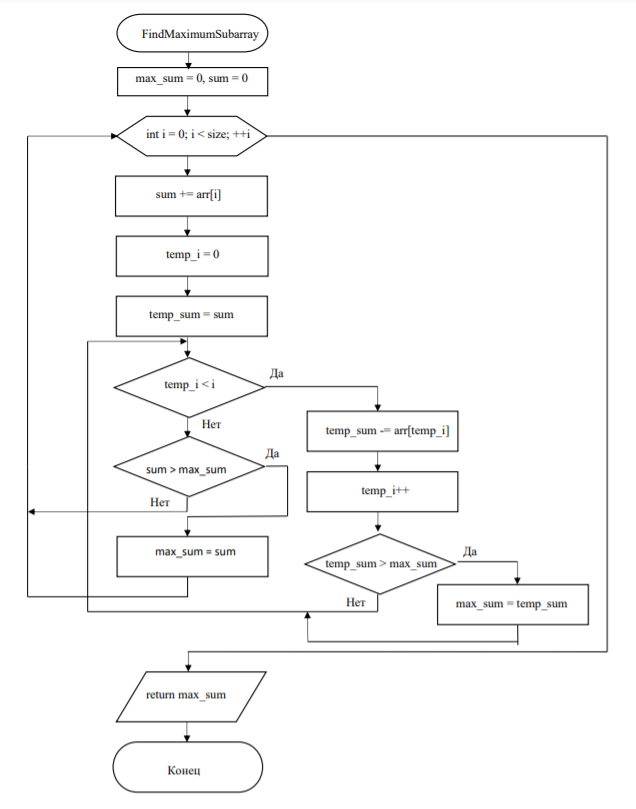
**Задача:**

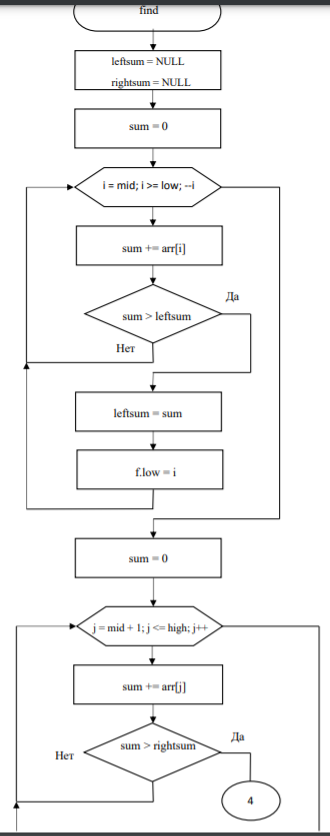
Реализовать программу, выполняющую алгоритм поиска наибольшего подмассива.

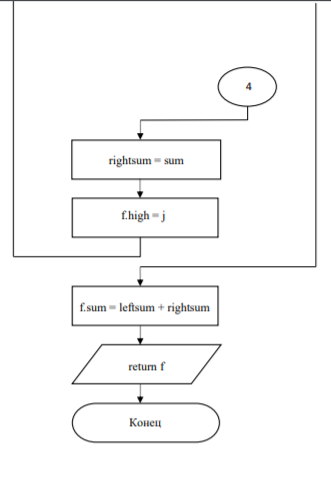
**Блок-схемы:**

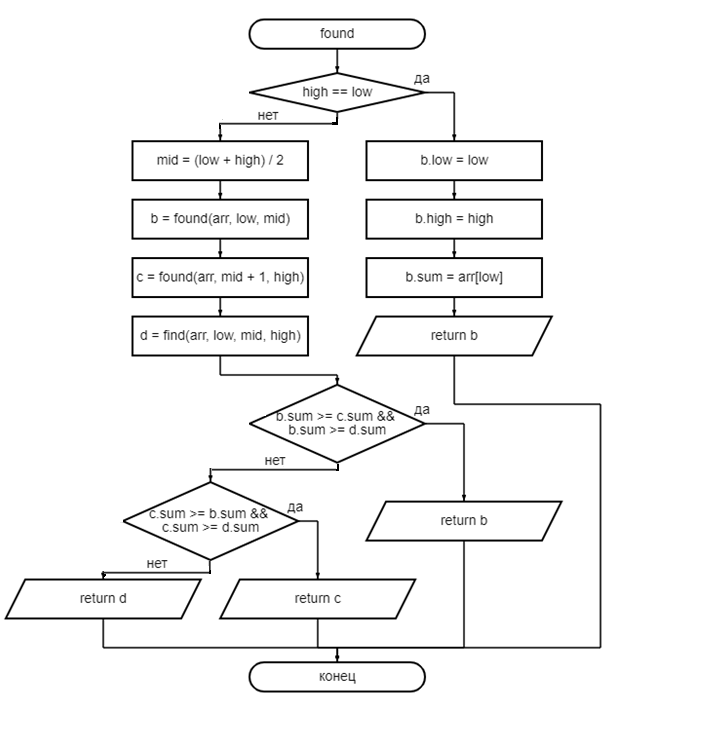


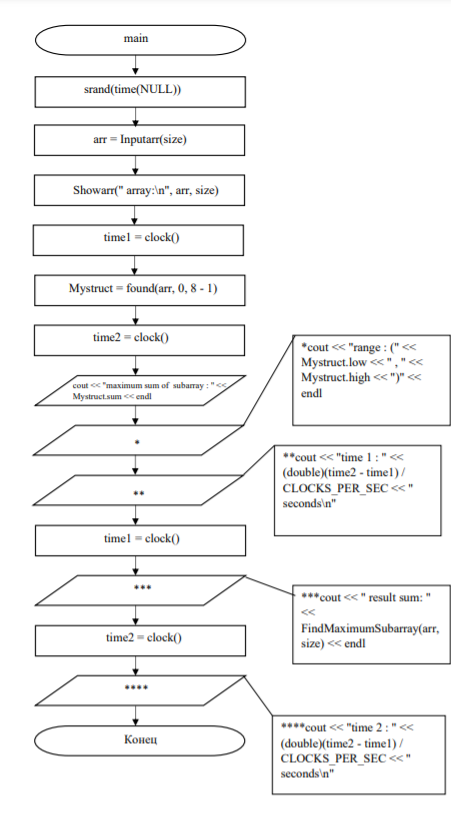






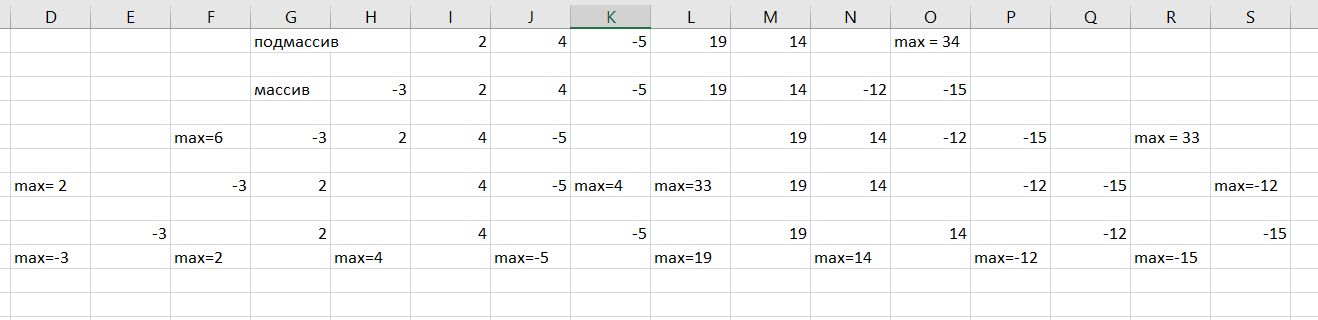


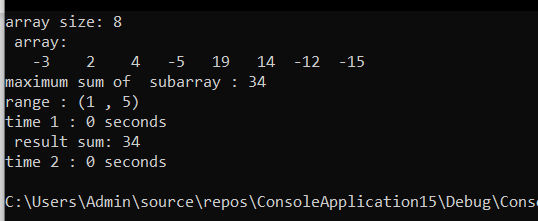




**Тестовые примеры:**

Возьмем массив длиною 8 элементов:





**Код программы:**

#include <iostream>

#include <ctime>

#include <iomanip>

using namespace std;

int\* Inputarr(int& size)

{

int i, \* arr;

cout << "array size: ";

cin >> size;

arr = new int[size];

for (i = 0; i < size; ++i)

arr[i] = rand() % 50 - 25;

return arr;

}

void Showarr(const char\* head, int\* arr, int size)

{

int i;

cout << head;

for (i = 0; i < size; ++i)

cout << setw(5) << arr[i];

cout << endl;

}

struct myfind1

{

int sum;

int low;

int high;

};

struct myfind1 find(int arr[], int low, int mid, int high)

{

struct myfind1 f;

int leftsum, rightsum, sum, i, j;

leftsum = NULL;

rightsum = NULL;

sum = 0;

for (i = mid; i >= low; --i)

{

sum += arr[i];

if (sum > leftsum)

{

leftsum = sum;

f.low = i;

}

}

sum = 0;

for (j = mid + 1; j <= high; j++)

{

sum += arr[j];

if (sum > rightsum)

{

rightsum = sum;

f.high = j;

}

}

f.sum = leftsum + rightsum;

return f;

}

struct myfind1 found(int arr[], int low, int high)

{

struct myfind1 b, c, d;

int mid;

if (high == low)

{

b.low = low;

b.high = high;

b.sum = arr[low];

return b;

}

else

{

mid = (low + high) / 2;

b = found(arr, low, mid);

c = found(arr, mid + 1, high);

d = find(arr, low, mid, high);

}

if (b.sum >= c.sum && b.sum >= d.sum)

return b;

else if (c.sum >= b.sum && c.sum >= d.sum)

return c;

else

return d;

}

double FindMaximumSubarray(int arr[], int size)

{

int max\_sum = 0, sum = 0;

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

{

sum += arr[i];

int temp\_i = 0;

int temp\_sum = sum;

while (temp\_i < i)

{

temp\_sum -= arr[temp\_i];

temp\_i++;

if (temp\_sum > max\_sum)

{

max\_sum = temp\_sum;

}

}

if (sum > max\_sum)

{

max\_sum = sum;

}

}

return max\_sum;

}

void main()

{

int\* arr, size;

clock\_t time1, time2;

struct myfind1 Mystruct;

srand(time(NULL));

arr = Inputarr(size);

Showarr(" array:\n", arr, size);

time1 = clock();

Mystruct = found(arr, 0, 8 - 1);

time2 = clock();

cout << "maximum sum of subarray : " << Mystruct.sum << endl;

cout << "range : (" << Mystruct.low << " , " << Mystruct.high << ")" << endl;

cout << "time 1 : " << (double)(time2 - time1) / CLOCKS\_PER\_SEC << " seconds\n";

time1 = clock();

cout << " result sum: " << FindMaximumSubarray(arr, size) << endl;

time2 = clock();

cout << "time 2 : " << (double)(time2 - time1) / CLOCKS\_PER\_SEC << " seconds\n";

}

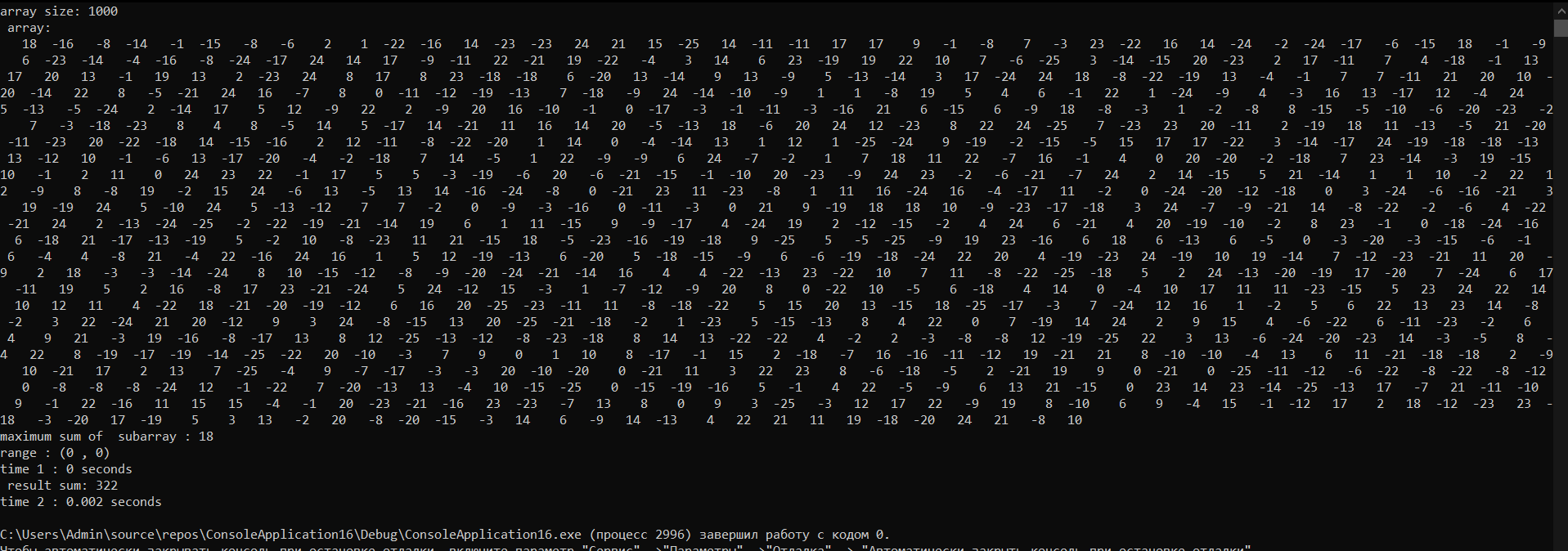
**Результаты программы:**

100 элементов:

Изображение выглядит как текст, внутренний, электроника, снимок экрана

Автоматически созданное описание

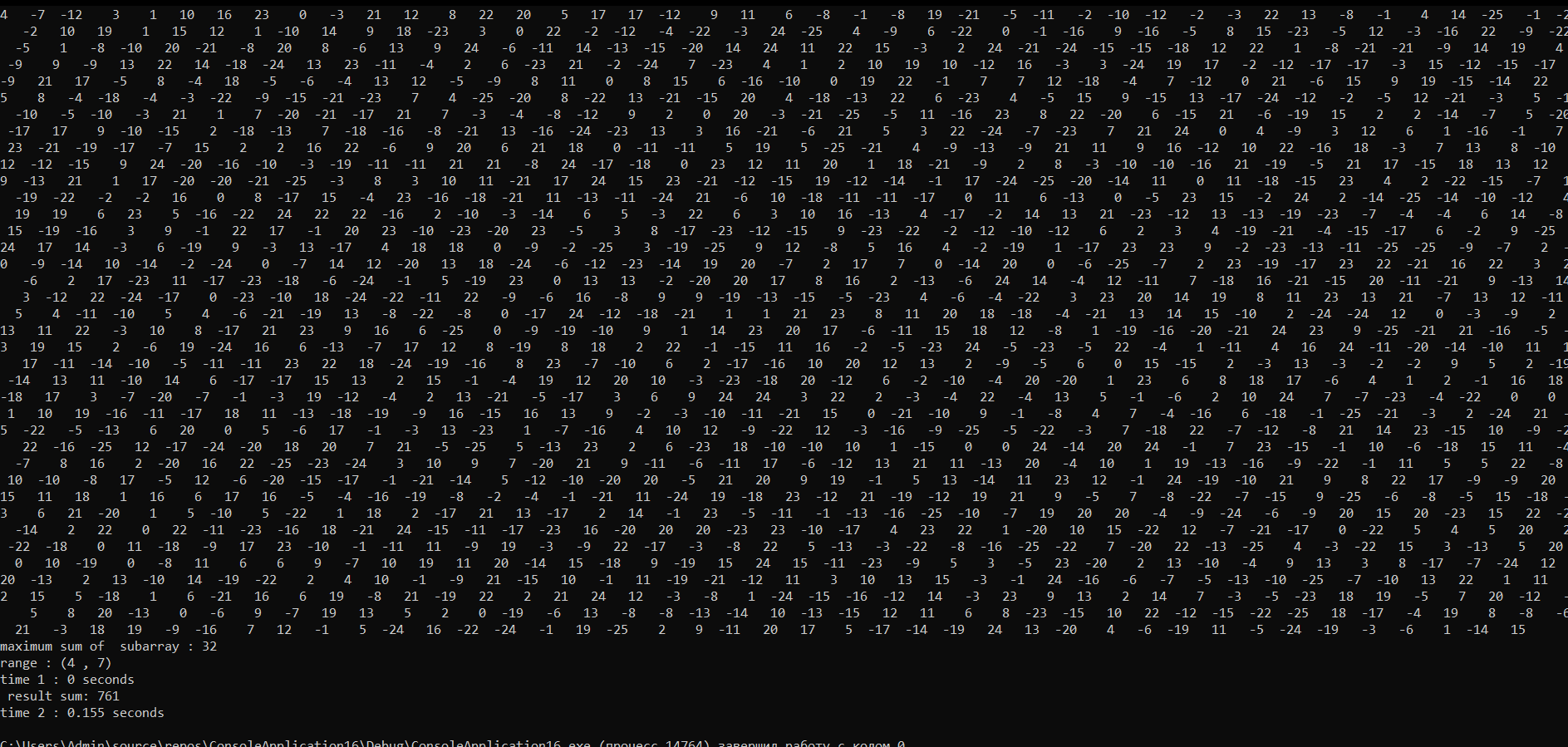
1000 элементов:



10 000 элементов:

Изображение выглядит как внешний

Автоматически созданное описание



**Вывод:**

В данной практической работе был реализован алгоритм поиска наибольшего подмассива. При реализации кода, проблем не возникло. Код работает корректно. Данный алгоритм сравнивался с алгоритмом, где напрямую использовался перебор. Как и предполагалось данный алгоритм оказался быстрее, чем грубый перебор.