Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №4**

Дисциплина: основы алгоритмизации и программирования

Тема: работа с одномерными массивами

Вариант 23

Выполнил:

Студент группы ИВТ-20-2б

Брейкин Алексей Дмитриевич

Проверил:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2020**

**Постановка задачи**

1) Реализовать с использованием массива двунаправленное кольцо (просмотр возможен в обе стороны, от последнего элемента можно перейти к первому).

2) Распечатать полученный массив, начиная с К-ого элемента и до К-1 (по кольцу влево).

3) Удалить из кольца все элементы, совпадающие с его максимальным значением.

4) Распечатать полученный массив, начиная с К-ого элемента (и до К+1 по кольцу вправо).

**Анализ задачи**

1. Просим пользователя ввести размер массива, учитывая ограничение размера > 1
2. Создаём динамический массив размером, который ввёл пользователь

2.1) Если флаг USER\_INPUT равен истине, то просим пользователя заполнить массив самостоятельно

2.2) В противном случае заполняем массив случайными числами с помощью датчика случайных чисел

3) Запрашиваем индекс, с которого нужно начать вывод кольца влево с помощью цикла

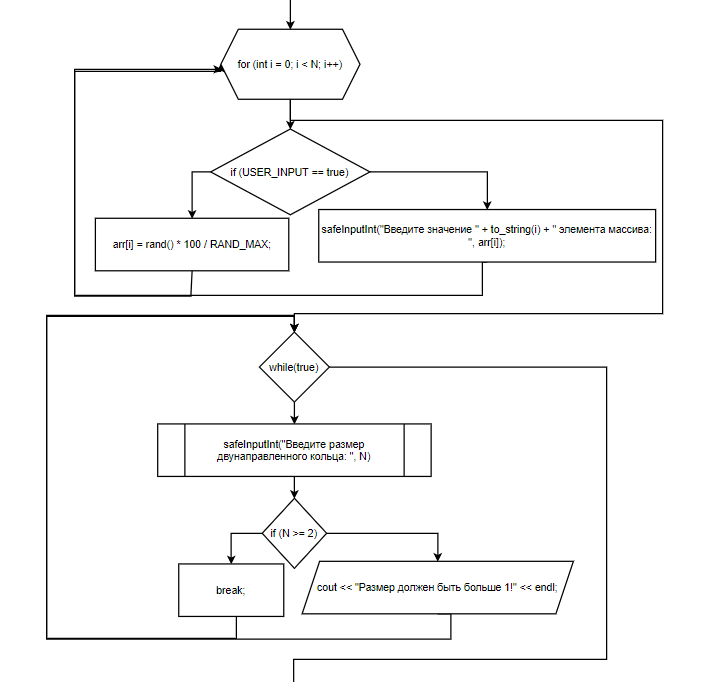
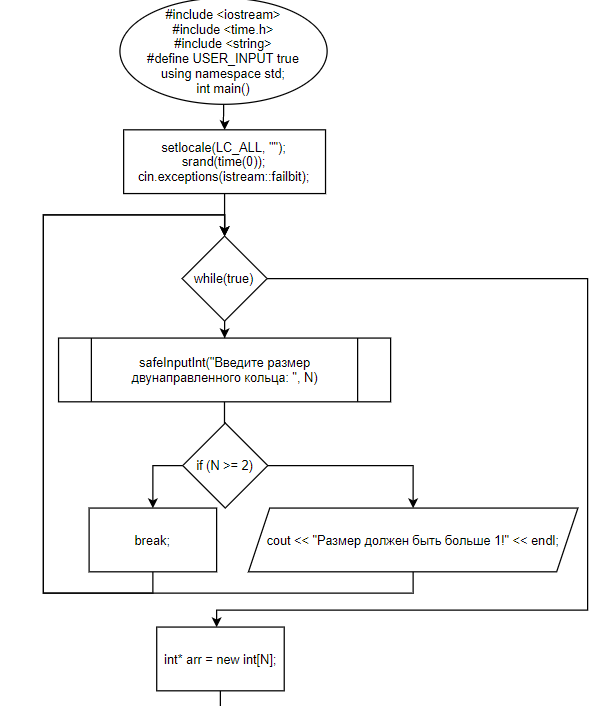
4) С помощью цикла ищем максимальное значение в массиве, выводим найденный элемент

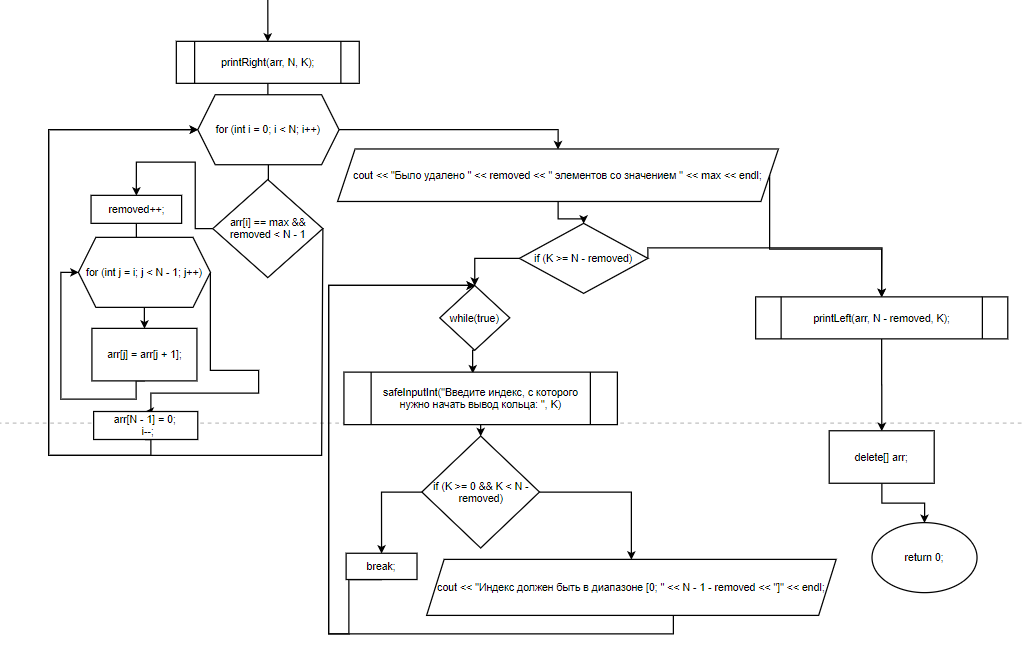
5) Используем вложенные циклы: через внешний проходимся по массиву, а через внутренний смещаем элементы, чтобы удалить найденные максимальные значения

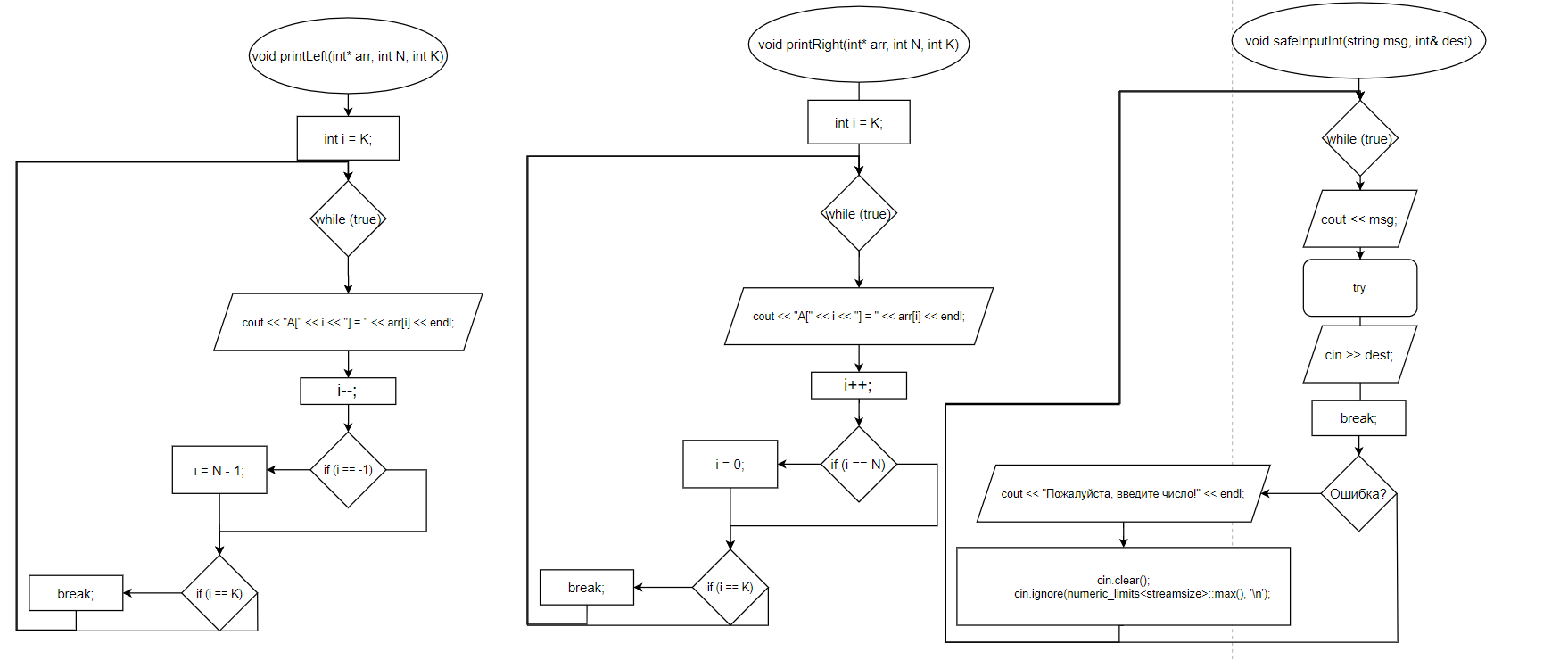
6) Используем предыдущий индекс, который ввёл пользователь на шаге (3), если элемент не был удалён. В противном случае, повторно запрашиваем индекс, учитывая новые ограничения.

7) Выводим массив с помощью цикла справа налево.

**Блок-схема**







**Исходный код**

#include <iostream>

#include <time.h>

#include <string>

using namespace std;

// set to true if you want to type all elements by yourself

constexpr auto USER\_INPUT = false;

// did you know?

// for (;;) will compile as while (true)

// so there is no any difference between them

// but for (;;) is faster than while (true)

// because it contains fewer characters

// and you can type it faster yooo

//// https://stackoverflow.com/questions/2611246/is-for-faster-than-while-true-if-not-why-do-people-use-it

void printLeft(int\* arr, int N, int K) {

int i = K;

for (;;) {

cout << "A[" << i << "] = " << arr[i] << endl;

i--;

if (i == -1) i = N - 1;

if (i == K) break;

}

}

void printRight(int\* arr, int N, int K) {

int i = K;

for (;;) {

cout << "A[" << i << "] = " << arr[i] << endl;

i++;

if (i == N) i = 0;

if (i == K) break;

}

}

void safeInputInt(string msg, int& dest) {

for (;;) {

cout << msg;

try {

cin >> dest;

break;

}

catch (std::ios\_base::failure const& ex) {

cout << "Пожалуйста, введите число!" << endl;

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

}

}

}

int main()

{

setlocale(LC\_ALL, ""); // set Russian lang

srand(time(0)); // set seed

cin.exceptions(istream::failbit); // enable cin exceptions to turn on safe input

int N;

for (;;) {

safeInputInt("Введите размер двунаправленного кольца: ", N);

if (N >= 2)

break;

cout << "Размер должен быть больше 1!" << endl;

}

int\* arr = new int[N];

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

if (!USER\_INPUT)

arr[i] = rand() \* 100 / RAND\_MAX;

else {

safeInputInt("Введите значение " + to\_string(i) + " элемента массива: ", arr[i]);

}

}

int K;

for (;;) {

safeInputInt("Введите индекс, с которого нужно начать вывод кольца: ", K);

if (K >= 0 && K < N) break;

cout << "Индекс должен быть в диапазоне [0; " << N - 1 << "]" << endl;

}

printRight(arr, N, K);

// find max

int max = 0;

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

if (arr[i] > max) max = arr[i];

}

cout << "Максимальное значение элемента в массиве: " << max << endl;

int removed = 0;

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

if (arr[i] == max && removed < N - 1) {

removed++;

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

arr[j] = arr[j + 1];

}

arr[N - 1] = 0;

i--;

}

}

cout << "Было удалено " << removed << " элементов со значением " << max << endl;

if (K >= N - removed) {

for (;;) {

safeInputInt("Введите индекс, с которого нужно начать вывод кольца: ", K);

if (K >= 0 && K < N - removed) break;

cout << "Индекс должен быть в диапазоне [0; " << N - 1 - removed << "]" << endl;

}

}

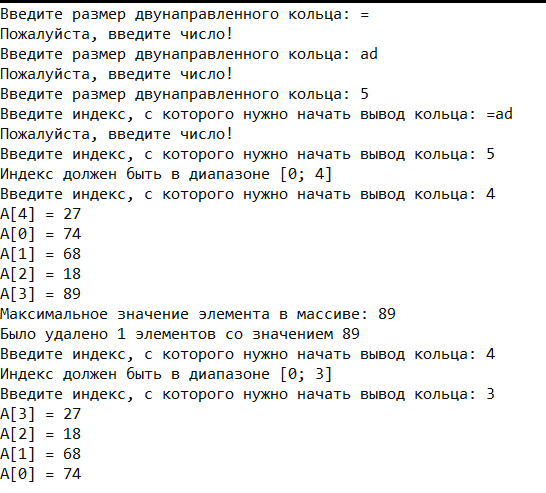
printLeft(arr, N - removed, K);

delete[] arr; // free memory after use :)

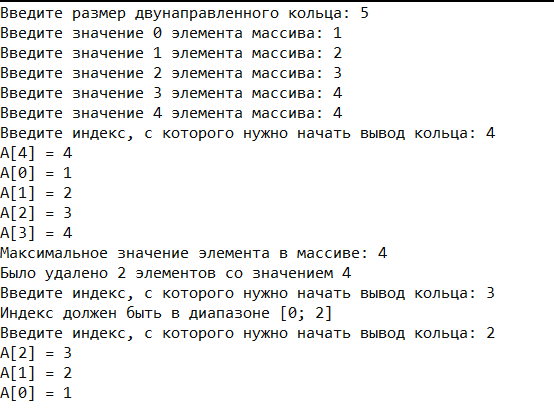
return 0;

}

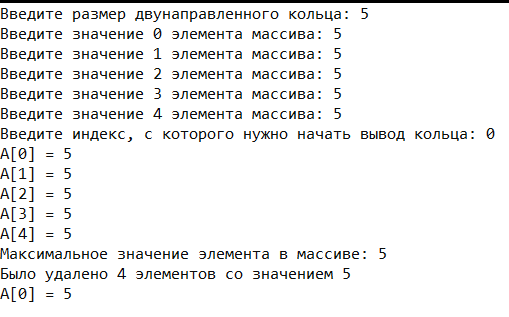
**Скриншоты работы программы**

 [1]

В первом примере показаны все возможные случаи обработки ошибок. Начиная с того, что пользователь вводит набор символов/букв вместо цифр и заканчивая тем, что программа повторно запрашивает индекс К, с которого нужно вывести кольцо после удаления максимального элемента(-ов).

[2]

Во втором примере показана работа программы, если скомпилировать её с флагом *USER\_INPUT = true.* Данный флаг позволяет пользователю самостоятельно вводить элементы массива, вместо того, чтобы они генерировались автоматически с помощью функции rand. Также здесь рассмотрено удаление нескольких элементов массива.

[3]

И в последнем, третьем примере, показано, что будет, если все элементы в массиве – максимальные. А в этом случае будут удалены все элементы, кроме одного.