1. **Задание**

В целочисленном массиве Х(N) после каждого положительного числа вставить ноль. Найти среднее арифметическое элементов массива до и после вставки. В программе написать функции: вставки элемента, поиска максимального элемента, определения среднего арифметического элементов массива.

1. ****Описание основных структур данных и алгоритмов, используемых в работе****

**Память для массива **A** выделяется динамически, с использованием операции new. Данные для массива **A** генерируются датчиком случайных чисел.**

**Переменные size и size\_of\_container определяют фактический размер массива и размер выделенной под массив области памяти соответственно.**

**Функция find\_median определяет среднее арифметическое всех элементов массива.**

**Функция find\_max определяет максимальный среди всех элементов массива.**

**Функция insert\_ осуществляет вставку элемента со значением val по индексу pos и осуществляет сдвиг элементов массива вправо от места вставки.**

**Функция realloc\_ осуществляет реаллокацию памяти массива, если контейнер полностью заполнен.**

1. ****Описание интерфейса пользователя программы****

**После запуска программы, на экран отображается подсказка с просьбой ввести количество элементов массиве.**

**После ввода числа, программа выполняет расчет и выводит на экран следующую информацию:**

* **Элементы массива **A** до вставок и среднее арифметическое**
* **Элементы массива **A** после вставок и среднее арифметическое**

****4. Контрольный пример и результаты тестирования****

**Введите размер массива**

**10**

**-9 17 -16 -50 19 -26 28 8 12 14**

**-0.3**

**-9 17 0 -16 -50 19 0 -26 28 0 8 0 12 0 14 0**

**-0.1875**

****4.** **Листинг программы****

#include <iostream>

#include <random>

using namespace std;

int\* realloc\_(int\*& arr, int& size\_of\_container) {

int\* extended = new int[size\_of\_container + 10];

for (int i = 0; i < size\_of\_container + 10; i++) {

extended[i] = 0;

}

for (int i = 0; i < size\_of\_container; i++) {

extended[i] = arr[i];

}

size\_of\_container += 10;

return extended;

}

void insert\_(int\*& arr, int pos, int val, int& size, int& size\_of\_container) {

// 'pos' = after which element 'val' will be inserted; [pos + 1] = index of val

if (size + 1 >= size\_of\_container)

arr = realloc\_(arr, size\_of\_container);

for (int i = size; i > pos + 1; i--) {

arr[i] = arr[i - 1];

}

arr[pos + 1] = val;

size++;

}

int find\_max(int\* arr, int size) {

int max = -100;

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

if (arr[i] > max)

max = arr[i];

}

return max;

}

double find\_median(int\* arr, int size) {

double median = 0;

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

median += arr[i];

}

median /= size;

return median;

}

int main() {

setlocale(LC\_ALL, "Russian");

cout << "Введите размер массива" << endl;

int size;

cin >> size;

int size\_of\_container = size;

int\* A = new int[size];

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

A[i] = rand() % 100 - 50;

}

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

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

}

cout << endl;

cout << find\_median(A, size) << endl;

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

if (A[i] > 0) {

insert\_(A, i, 0, size, size\_of\_container);

}

}

//cout << size << " " << size\_of\_container << endl;

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

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

}

cout << endl;

cout << find\_median(A, size) << endl;

}