КАФЕДРА №

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

АНАЛИЗ СЛОЖНОСТИ АЛГОРИТМОВ

по курсу: Структуры и алгоритмы обработки данных

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

	подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

1.1 Цель работы:

Целью работы является изучение методов и получение практических навыков анализа сложности алгоритмов.

1.2 Задание на лабораторную работу:

Используя память, пропорциональную n, хранить массив целых чисел A, содержащий n элементов. Элементы массива A могут принимать случайные значения от -((n div 2) - 1) до (n div 2). То есть, если в массиве хранится 10 элементов, то эти элементы должны быть в диапазоне от -4 до 5. Разработать алгоритм, который осуществляет заполнение массива A случайными значениями, и по выбору пользователя выполняет одну из двух функций.

4	Подсчитать сумму всех элементов, имеющих положительные значения	$O(n)$
	Подсчитать количество элементов с четными значениями	$O(1)$

1.3 Листинг программы:

```
/*
 4 вариант ( (26 % 03) + 1 )

  Используя память, пропорциональную n, хранить массив целых чисел
A, содержащий n элементов.
  Элементы массива A могут принимать случайные значения от
-((n div 2) - 1) до (n div 2). То есть, если в массиве хранится 10 элементов, то
эти элементы должны быть в диапазоне от -4 до 5.
  Разработать алгоритм, который осуществляет заполнение массива A
случайными значениями, и по выбору пользователя выполняет одну из двух
функций.

- Подсчитать сумму всех элементов, имеющих положительные значения
- Подсчитать количество элементов с четными значениями

*/

#include <iostream>
using namespace std;
```

```
20.
21.#include <cmath>
22.#include <time.h>
23.
24.#include "functions.h"
25.// Подсчитать сумму всех элементов, имеющих положительные значения
26.int get_sum_pos(int* arr, int size) {
27.int sum = 0;
28.for (int i = 0; i < size; i++) {
29.if (arr[i] > 0)
30.sum += arr[i];
31.}
32.return sum;
33.}
34.// Подсчитать количество элементов с чётными значениями
35.void get_count_pol(int count) {
36.if (count > 0) {
37.cout << "Количество чётных элементов: " << count << endl;
38.} else
39.cout << "В массиве нет положительных элементов." << endl;
40.}
41.int main() {
42.    // смена кодировки
43.    setlocale(LC_ALL, "Russian");
44.
45.  //рандом
46.  srand(time(NULL));
47.
48.  int size = 0;
49.
50.  // ввод размера массива
51.  while (true) {
52.    //size = read_value("Размер массива: ", true, true, false);
53.    cout << "Размер массива: ";
54.    scanf("%d", &size);
55.
56.    if (size > 0)
57.      break;
58.
59.    else
60.      cout << "Размер массива должен быть больше 0." << endl;
61.  }
```

```
62.
63.  // создаём массив
64.  int* arr = (int*)malloc(size * sizeof(int));
65.
66.  // диапазон чисел для заполнения через рандом
67.  int rand_min = - ((size / 2) - 1);
68.  int rand_max = size / 2;
69.
70.  // заполняем
71.  for (int i = 0; i < size; i++) {
72.    arr[i] = rand_min + (rand() % ( rand_max - rand_min + 1 ) );
73.  }
74.
75.  int num;
76.  bool run = true;
77.
78.  int sum;
79.  int count;
80.
81.  // вывод меню и информации о массиве
82.  cout << "Числа сгенерированны в диапазоне от " << rand_min << " до " << rand_max << endl;
83.  cout << "Массив: ";
84.
85.  for (int i = 0; i < size; i++)
86.    cout << arr[i] << " ";
87.  cout << endl;
88.
89.
90.  cout << "1 - Подсчитать сумму всех элементов, имеющих положительные значения." << endl;
91.  cout << "2 - Подсчитать количество элементов с чётными значениями." << endl << endl;
92.  cout << "0 - Выход" << endl;
93.
94.  // цикл для меню
95.  while (run) {
96.
97.    // ввод пункта меню
98.    //num = read_value(" >> ", true, true, false);
99.    cout << " >> ";
100.      scanf("%d", &num);
101.
102.      switch (num) {
103.
```

```
104.       // Подсчитать сумму всех элементов, имеющих положительные значения
105.       case (1):
106.
107.         sum = get_sum_pos(arr, size);
108.         if (sum > 0) {
109.           cout << "Сумма положительных элементов: " << sum << endl;
110.         } else
111.           cout << "В массиве нет положительных элементов." << endl;
112.
113.         break;
114.
115.       // Подсчитать количество элементов с чётными значениями.
116.       case (2):
117.
118.
119.         count = 0;
120.
121.         for (int i = 0; i < size; i++) {
122.           if (arr[i] % 2 == 0) count++;
123.         }
124.
125.         get_count_pol(count);
126.
127.
128.         break;
129.
130.       // выход
131.       case (0):
132.         run = false;
133.         break;
134.     }
135.   }
136.
137.
138.   free(arr);
139.     return 0;
140. }
```

1.4 Расчет теоретической пространственной сложности алгоритма:

В данном алгоритме содержатся, следующие переменные:

1. Один массив размерностью size;

2. 11 целочисленного типа;
3. 1 bool тип.

$$v = n * C_{\text{int}} + 11 * C_{\text{int}} + C_{bool}$$

$$V(n) = O(v) = O(\max(O(n * C_{\text{int}}), O(11 * C_{\text{int}}), O(C_{bool}))) = O(\max(O(n), O(1), O(1))) = O(n)$$

1.5 Расчет теоретической временной сложности алгоритма:

$$t_{\text{get\_sum\_pos}} = K_{32} + n * K_{34} + K_{36}$$

$$t_{\text{get\_count\_pos}} = K_{45}$$

$$t_{Alg} = K_{57} + K_{63} + K_{73} + K_{76} + K_{77} + K_{80} + n * K_{81} + K_{84} + K_{85} + K_{87} + K_{88} + K_{91} + K_{94} + n * K_{95} + K_{109}$$

$$+ K_{118} + K_{130} + n * K_{131} + t_{\text{get\_sum\_pos}} + t_{\text{get\_count\_pos}}$$

Теоретическая временная сложность функций составляет:

$$T_{\text{get\_sum\_pos}}(n) = O_{(\text{get\_sum\_pos})} = O(\max(O(K_{32}), O(n * K_{34}), O(K_{36})) = O(max(O(1),\ O(n))) = O(n)$$

$$T_{\text{get\_count\_pos}}(n) = O_{(\text{get\_count\_pos}} = O(K_{45}) = O(1)$$

1.6 Скриншот выполнения программы:

```
Размер массива: 20
Числа сгенерированны в диапазоне от -9 до 10
Массив: -7 6 7 -9 -3 -7 3 8 8 -3 4 5 -5 3 4 -9 -8 -1 2 6
1 - Подсчитать сумму всех элементов, имеющих положительные значения.
2 - Подсчитать количество элементов с чётными значениями.

0 - Выход
 >> 1
Сумма положительных элементов: 56
 >> 2
Количество чётных элементов: 8
 >> 0
Program ended with exit code: 0
```

1.7 Вывод:

На основе этих расчетов можно сделать вывод, что был разработан алгоритм, характеристики которого соответствуют поставленному заданию