|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 1** | |
| **по дисциплине** | |
| **«**Структуры и алгоритмы обработки данных**»**  **Тема: «Одномерный массив»** | |
|  | |
| Выполнил студент группы ИКБО-72-23 | Шатохин Б.А. |
| Принял преподаватель | Филатов А.С. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись руководителя)* |

Москва 2024

# **Цель работы**

Получение навыков по определению одномерного массива для структуры данных задачи. Приобретение навыков создания алгоритмов работы над одномерным массивов. Получение навыков по реализации алгоритмов операций над массивом через аппарат функций.

# **Постановка задачи**

Разработать программы для выполнения операций на статическом массиве, динамическом, используя контейнер <vector>. Составить отчёт.

1. Разработать функции ввода и вывода значений массива.
2. Выполнить декомпозицию индивидуальных задача.
3. Разработать алгоритмы решения индивидуальных задача.
4. Реализовать функции необходимые для решения индивидуальных задач.
5. Реализовать программу, демонстрирующую работу всех функций.

Индивидуальное задание:

1. Сформировать новый массив из простых чисел исходного массива, вставляя каждое значение (кроме первого) так, чтобы числа образовали в результате возрастающую последовательность.
2. Удалить минимальное число в последовательности.
3. Определить у скольких чисел исходного массива количество делитель больше трёх.
4. Решение

Массив – непрерывный участок памяти, содержащий последовательность объектов одинакового типа данных, обозначаемый одним именем. Массив характеризуется следующими понятиями:

* Элемент массива – значение, хранящиеся в определённой ячейке памяти, расположенной в пределах массива.
* Адрес массива – адрес начального элемента.
* Имя массива – идентификатор, используемы для обращения к элементам массива.
* Размер массива – количество байт, занимаемых одним элементом массива.

Графически массив можно представить так.

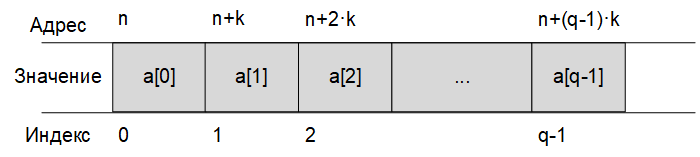


Рисунок 1. Представление массива в памяти

Так как адрес массива – это адрес первого элемента, то для выбора последующих элементов используется алгебра указателей.

Для статического массива мы используем заранее определённый, константный размер, в связи с этим, изменение размера массива в ходе выполнения программы невозможен. Для динамического массива мы используем оператор new, который выделяет ячейку памяти. В этом случае изменить размер можно, заново выделив память. Контейнер <vector> включает в себя методы для добавления и удаления элемента в массив, что облегчает работу с массивом.

Для статического массива:

Для заполнения массива была использована функция addToArray, принимающая в качестве аргументов массив, ссылку на переменную, хранящую в себе текущий размер массива (т.е. количество элементов) и число, которое нужно добавить.

int addToArray(int staticArray[], int& size, int newNumber) {

if (size >= MAX\_SIZE) return -1;

staticArray[size] = newNumber;

size++;

return 0;

}

Для вывода массива используется функция printArray, принимающая в качестве аргументов сам массив и текущий размер массива.

int printArray(int arrayToPrint[], int size) {

std::cout << "Массив: ";

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

std::cout << arrayToPrint[i] << " ";

std::cout << std::endl;

return 0;

}

Для удаления элементов массива используется функция RemoveFromArray, которая принимает в качестве аргументов сам массив, его текущий размер и индекс элемента.

int RemoveFromArray(int staticArray[], int& size, int position) {

if (size <= 0) return -1;

if (position >= size) return -2;

for (int i = position; i < size - 1; i++)

staticArray[i] = staticArray[i + 1];

size--;

return 0;

}

Для нахождения числа с определённым количеством делителей используется функция prime, которая в качестве аргумента принимает число.

int prime(int x) {

int k = 2;

for (int i = 2; i <= sqrt(x); ++i) {

if (x % i == 0) {

k += 2;

}

}

return k;

}

Для создания ещё одного массива используется функция makeSecondArray, которая принимает в качестве аргументов массив, текущий размер массива и число.

int makeSecondArray(int Array[], int& size, int newNumber) {

if (size >= MAX\_SIZE) return -1;

if (size == 0) Array[size] = newNumber;

for (int i = size - 1; i >= 0; --i) {

if (Array[i] <= newNumber) {

Array[i + 1] = newNumber;

break;

}

else {

Array[i + 1] = Array[i];

}

}

size++;

return 0;

}

Для динамического массива:

Были изменены две функции: addToArray и RemoveFromArray.

Функция RemoveFromArray.

int RemoveFromArray(int\* Array, int& size, int position) {

if (size <= 0) return -1;

if (position >= size) return -2;

for (int i = position; i < size - 1; i++)

Array[i] = Array[i + 1];

size--;

Array = (int\*)realloc(Array, size \* sizeof(int));

return 0;

}

Функция addToArray.

Для контейнера <vector>:

int addToArray(int\*& Array, int& size, int newNumber) {

bool flag = false;

if (size == 0) Array[size] = newNumber;

else {

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

if (Array[i] >= newNumber) {

Array[i + 1] = Array[i];

Array[i] = newNumber;

flag = true;

break;

}

}

if (flag == false) Array[size] = newNumber;

}

size++;

Array = (int\*)realloc(Array, size \* sizeof(int));

return 0;

}

Для контейнера функции addToArray и RemoveFromArray были вырезаны, так как удобнее использовать встроенные методы push\_back() и pop().

Так же в программе было реализовано меню.

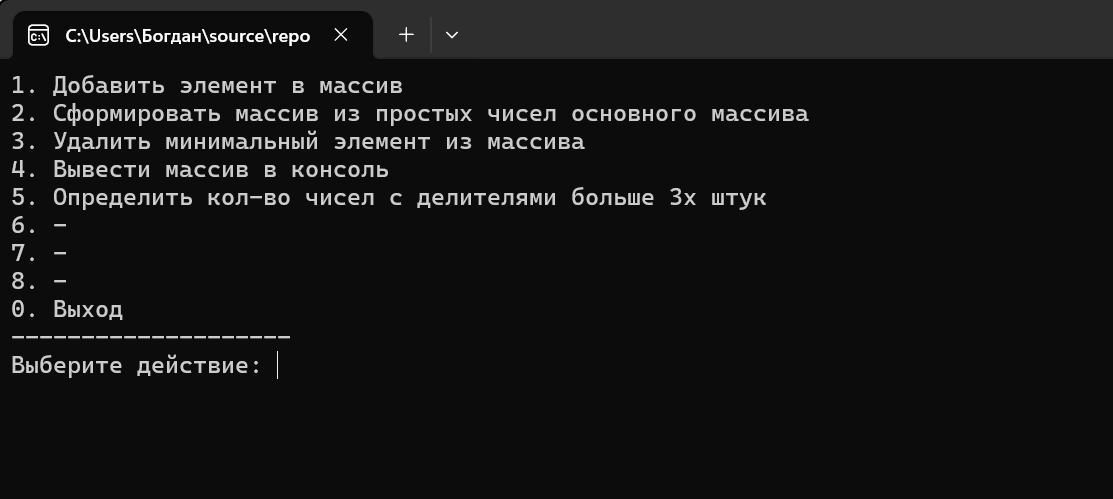


Рисунок 2. Интерфейс программы

1. Тестирование

Для статического массива:

Введём в массив числа – 2, 3, 5, 6 и выведем его.

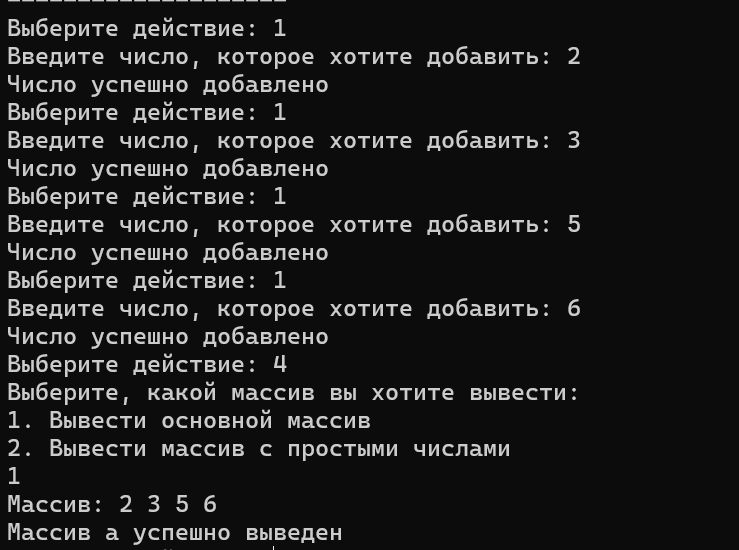


Рисунок 3. Демонстрация заполнения массива

Используем вторую функцию и сформируем второй массив из простых чисел. После, выведем этот массив в консоль.

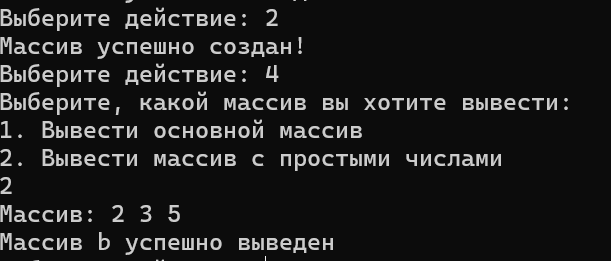


Рисунок 4. Демонстрация создания массива простых чисел

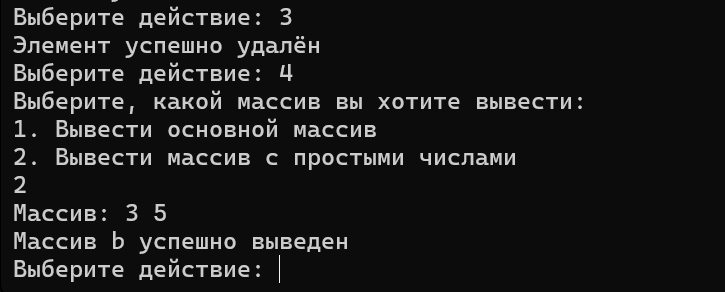


Рисунок 5. Демонстрация возможности удаления минимального числа

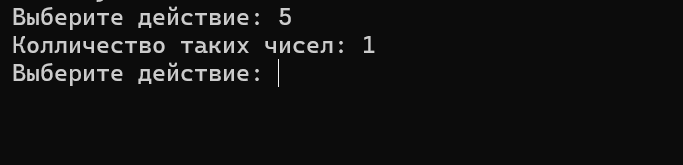


Рисунок 6. Демонстрация работы функции по нахождению чисел

1. Вывод

В результате выполнения работы, были получены навыки по созданию и обработке одномерного массива. Были разработаны алгоритмы согласно индивидуальным заданиям, а так же были получены навыки по работе с динамической памятью.

1. **Исходный код**

int main() {

std::setlocale(0, "ru");

char menu;

std::vector <int> a;

std::vector <int> b;

int k = 0;

int x;

std::cout

<< "1. Добавить элемент в вектор\n"

<< "2. Сформировать вектор из простых чисел основного вектора\n"

<< "3. Удалить минимальный элемент из вектора\n"

<< "4. Вывести вектор в консоль\n"

<< "5. Определить кол-во чисел с делителями больше 3х штук\n"

<< "6. -\n"

<< "7. -\n"

<< "8. -\n"

<< "0. Выход\n"

<< "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

while (true) {

std::cout << "Выберите действие\n";

std::cin >> menu;

switch (menu) {

case '1':

std::cout << "Введите число, которое хотите добавить:\n";

std::cin >> x;

a.push\_back(x);

std::cout << "Число успешно добавлено\n";

break;

while (true) {

std::cout << "Выберите действие\n";

std::cin >> menu;

switch (menu) {

case '1':

std::cout << "Введите число, которое хотите добавить:\n";

std::cin >> x;

a.push\_back(x);

std::cout << "Число успешно добавлено\n";

break;

case '2':

sort(begin(a), end(a));

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

if (prime(a[i]) == 2) {

b.push\_back(a[i]);

}

}

if (b.empty()) {

std::cout << "Простых числе нет!\n";

}

else {

std::cout << "Вектор создан!\n";

}

break;

case '3':

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

if (b[i] != b[i + 1]) {

std::erase(b, b[i]);

break;

}

else std::erase(b, b[i]);

}

break;

case '4':

char menu2;

std::cout << "Выберите, какой вектор вы хотите вывести:\n"

<< "a. Вывести основной вектор\n"

<< "b. Вывести вектор с простыми числами\n";

std::cin >> menu2;

switch (menu2) {

case 'a':

printVector(a);

std::cout << "Вектор a успешно выведен\n";

break;

case 'b':

printVector(b);

std::cout << "Вектор b успешно выведен\n";

break;

default:

std::cout << "Неверно выбран массив!";

break;

}

break;

case '5':

k = 0;

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

if (prime(a[i] >= 3)) {

k += 1;

}

}

std::cout << "Колличество таких чисел: " << k << "\n";

break;

std::cout << "Колличество таких чисел: " << k << "\n";

break;

case '6':

break;

case '7':

break;

case '8':

break;

case '0':

return false;

break;

default:

std::cout << "Неверный ввод!\n";

std::cout

<< "1. Добавить элемент в вектор\n"

<< "2. Сформировать вектор из простых чисел основного вектора\n"

<< "3. Удалить минимальный элемент из вектора\n"

<< "4. Вывести вектор в консоль\n"

<< "5. Определить кол-во чисел с делителями больше 3х штук\n"

<< "6. -\n"

<< "7. -\n"

<< "8. -\n"

<< "9. -\n"

<< "0. Выход";

break;

}

}

}