|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | 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-09-22 | Гришин А. В. |
| Принял преподаватель | Филатов А.С. |

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

Москва 2023

# **Цель работы**

Приобретение навыков по определению одномерного массива для структуры данных задачи, приобретение навыков создания алгоритмов операций над одномерным массивом и получение навыков по реализации алгоритмов операций над массивом через аппарат функций.

1. **Постановка задачи**
2. Разработать программу для выполнения операций на статическом массиве.
3. Разработать функции для ввода и вывода значений массива.
4. Выполнить декомпозицию задач, определенных вариантом индивидуального задания.
5. Разработать алгоритмы решения задач, определенных вариантом индивидуального задания.
6. Реализовать функции, выполняющие задачи, определенные вариантом индивидуального задания.
7. Разработать программу, демонстрирующую работу всех функций.
8. Изменить разработанную программу так, чтобы все операции выполнялись над динамическим массивом.
9. Изменить разработанную программу так, чтобы все операции выполнялись над динамическим массивом, используя контейнер <vector> для его представления в программе.
10. Составить отчет, отобразив в нем описание выполнения всех этапов разработки, тестирования и код всей программы со скриншотами результатов тестирования.

Персональный вариант – 4.

Таблица 1. Задачи варианта.

|  |
| --- |
| 1. Вставить новое значение после значения в заданной позиции. 2. Определить, сколько раз входит в массив максимальное значение массива (одним алгоритмом).   3Удалить все числа массива, которые являются совершенными числами (число равно сумме своих делителей кроме самого числа: 6, 28). |

# **Решение**

Массив – это линейная, однородная структура данных, состоящая из конечного числа элементов. Все элементы массива принадлежат одному типу данных. Доступ к элементу массива осуществляется по индексу. В языке программирования C++ можно выделить 3 типа хранилищ данных: статический массив, динамический массив и контейнер <vector>

**Используемые функции:**

**3.1 Функция для вывода значения массива**

Ниже представлены функции для вывода значений на статическом динамическом массивах и для вывода значений вектора соответственно

|  |
| --- |
| int printArray(int arrayToPrint[], int size)  {  cout << "Массив: ";  for (int i = 0; i < size; i++)  cout << arrayToPrint[i] << " ";  cout << endl;  return 0;  }  int printDynArray(int arrayToPrint[], int size)  {  cout << "Массив: ";  for (int i = 0; i < size; i++)  cout << arrayToPrint[i] << " ";  cout << endl;  return 0;  }  void printVec(vector<int> sVector) {  cout << "Вектор: ";  for (int i = 0; i < sVector.size(); i++)  cout << sVector[i] << " ";  cout << endl;  } |

Самая простая реализация данной функции представлена у вектора, так как для данного контейнера не требуется значения размера. Функция для двух массивов между собой не различается

**3.2 Функция для вывода значения массива**

Ниже представлены функции для ввода значений на статическом динамическом массивах и для ввода значений вектора соответственно

|  |
| --- |
| int addToArray(int staticArray[], int& size, int newNumber)  {  if (size >= maxStaticSize) return -1;  staticArray[size] = newNumber;  size++;  return 0;  }  int addToDynArray(int\*& dynamicArray, int& size, int newNumber) {  size++;  dynamicArray = (int\*)realloc(dynamicArray, size \* sizeof(int));  dynamicArray[size-1] = newNumber;  return 0;  }  int addToVector(vector<int>& sVector, int newNumber) {  sVector.push\_back(newNumber);  return 0;  } |

Для работы с одномерным массивом требуется отслеживать его максимальный размер, чтобы не выйти за пределы изначально выделенного количества ячеек. Функция принимает массив, текущий размер массива, который после выполнения программы возвращается, и число, которое требуется вставить.

Для работы с динамическим массивом мы используем фунцию realloc(), которая перераспределяет блок памяти, который был ранее выделен, но еще не освобожден. Функция принимает массив, текущий размер массива, который после выполнения программы возвращается, и число, которое требуется вставить.

Для работы с вектором используется метод push\_back(),который автоматически добавляет значениие в конец вектора. Функция принимает вектор и число, которое требуется вставить

**3.3 Функция вставки нового значения после значения в заданной позиции**.

Ниже представлены функции для вставки значения после значения в заданной позиции на статическом динамическом массивах и векторе соответственно

|  |
| --- |
| int addToArrayAfterPos(int staticArray[], int& size, int newNumber, int poscion) {  if (size >= maxStaticSize) return -1;  size++;  for (int i = size - 1; i > poscion + 1; i--) {  staticArray[i] = staticArray[i - 1];  }  staticArray[poscion + 1] = newNumber;  return 0;  }  int addToArrayAfterPos(int dynamicArray[], int& size, int newNumber, int position) {  size++;  for (int i = size - 1; i > position + 1; i--) {  dynamicArray[i] = dynamicArray[i - 1];  }  dynamicArray[position] = newNumber;  return 0;  }  void addToVectorAfterPos(vector<int>& sVector, int newNumber, int poscion) {  sVector.resize(sVector.size() + 1);  for (int i = sVector.size() - 1; i > poscion + 1; i--) {  sVector[i] = sVector[i - 1];  }  sVector[poscion + 1] = newNumber;  } |

Функции массивов отличаются лишь тем, что у динамического массива мы не проверяем его на предмет переполнения. А так для обоих функций работа алгоритма одинакова:

1. Увеличиваем размер массива.

2. Переносим в новую ячейку значение из ячейки из ячейки слева до position+1

3. Вставляем число в освободившуюся ячейку по индексу position

Функции принимают статический/динамический массив, размер, который после выполнения возвращают, число и заданную позицию.

Алгоритм работы вектора ничем не отличается, за исключением того, что используется метод resize(), который изменяет размеры вектора на необходимый до необходимого размера. Функция имеет те же аргументы, что и массивы, за исключением размера.

**3.4 Функция** **определяющая, сколько раз входит в массив максимальное значение массива (одним алгоритмом).**

Ниже представлены функции определяющие, сколько раз входит в массив максимальное значение массива (одним алгоритмом) на статическом динамическом массивах и векторе соответственно

|  |
| --- |
| int countMAX(int staticArray[], int size, int& maxchislo, int& kolvo)  {  maxchislo = 0;  kolvo = 0;  for (int i = 0; i < size; i++) {  int chislo = staticArray[i];  if (maxchislo < chislo) {  kolvo = 0;  }  if (staticArray[i] >= maxchislo) {  maxchislo = staticArray[i];  kolvo++;  }  }  return 0;  }  int countMAX(int dynamicArray[], int size, int& maxchislo, int& kolvo) {  kolvo = 0;  maxchislo = 0;  for (int i = 0; i < size; i++) {  if (dynamicArray[i] > maxchislo) {  maxchislo = dynamicArray[i];  }  }  for (int i = 0; i < size; i++) {  if (dynamicArray[i] == maxchislo) {  kolvo += 1;  }  }  return 0;  }  int countMAX(vector<int> sVector, int& maxchislo, int& kolvo) {  kolvo = 0;  maxchislo = 0;  for (int i = 0; i < sVector.size(); i++) {  int chislo = sVector[i];  if (maxchislo < chislo) {  kolvo = 0;  }  if (sVector[i] > maxchislo) {  maxchislo = sVector[i];  kolvo++;  }  }  return 0;  } |

Алгоритм работы в этих программах ничем не отличается и выглядит примерно так:

1.Присваиваем переменной kolvo, отвечающей за количество повторений значение 0

2. Начинаем проходить по элементам массива/вектора

3. Инициализируем переменную, которая фиксирует только поступившее число

4. Проверяем, больше ли это число максимального и если да, то обнуляем количество

5. Если нет, то присваиваем значение числа к переменной, отвечающей за максимальное число.

Отличаются эти функции лишь тем, что у массива в аргументах есть переменная, отвечающая за размер, а у вектора - нет.

**3.4 Функция удаляющая все числа массива, которые являются совершенными числами (число равно сумме своих делителей кроме самого числа: 6, 28):**

Ниже представлены функции удаляющая все числа массива, которые являются совершенными числамидля статического и динамического массивов и вектора.

|  |
| --- |
| int dellAllAbsolute(int staticArray[], int& size) {  if (size <= 0) return -1;  for (int i = 0; i < size; i++) {  if (isAbsolute(staticArray[i]) == 1) {  for (int z = i; z < size - 1; z++) {  staticArray[z] = staticArray[z + 1];  }  size--;  i--;  }  }  return 0;  }  int dellAllAbsolute(int dynamicArray[], int& size) {  for (int i = 0; i < size; i++) {  if (isAbsolute(dynamicArray[i]) == 1) {  for (int z = i; z < size - 1; z++) {  dynamicArray[z] = dynamicArray[z + 1];  }  size--;  i--;  }  }  return 0;  }  int dellAllAbsolute(vector<int>& sVector) {  for (int i = 0; i < sVector.size(); i++) {  if (isAbsolute(sVector[i]) == 1) {  sVector.erase(sVector.begin() + i);  i--;  }  }  return 0;  } |

Алгоритм работы для функции для массивов :

1. Перебор по массиву

2. Если элемент является абсолютным число (проверяется функцией isAbsolute) то все элементы массива, начинаная с конца сдвигаются влево к элементу, который является абсолютным числом.

3. Уменьшаем размер массива

4. Уменьшаем значение итератора.

Для вектора:

1. Перебор по вектору.

2. Если элемент является абсолютным число (проверяется функцией isAbsolute), то используем метод erase(), который удаляет нужный элемент

3. Уменьшаем значение итератора.

Функция isAbsolute:

|  |
| --- |
| int isAbsolute(int n) {  int summa = 0;  for (int i = 1; i < n; i++) {  if (n % i == 0) {  summa += i;  }  }  if (summa == n) {  return 1;  }  return 0;  } |

Функция суммирует все делители искомого числа и,если сумма равна делимому, возвращает единицу

**4.Тестирование**

Протестируем работу программы для всех трёх задач на статическом, динамическом массивах и векторе по отдельности.

Рассмотрим выполнение первых двух функций на статическом массиве (рис. 1), динамическом массиве (рис. 2), векторе (рис. 3)

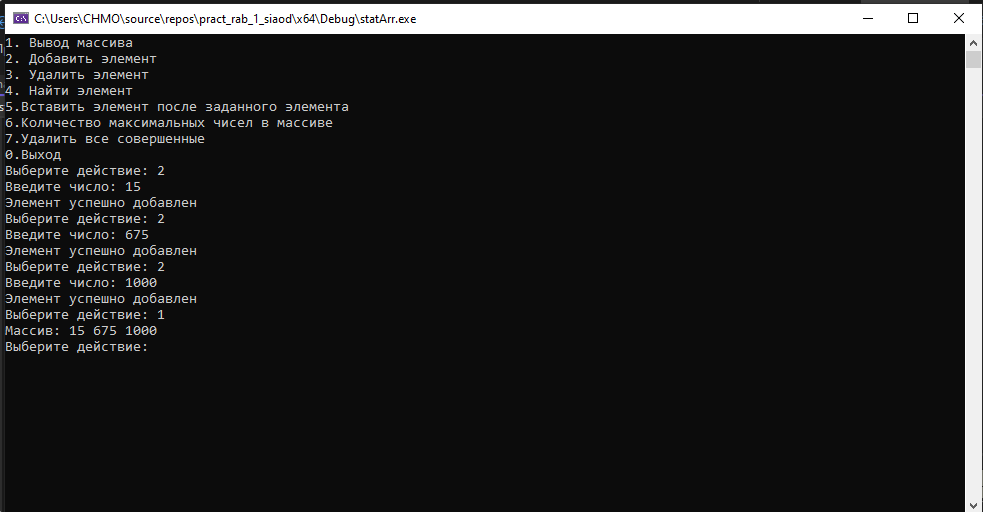


Рисунок 1. Тестирование первых двух функций на статическом массиве

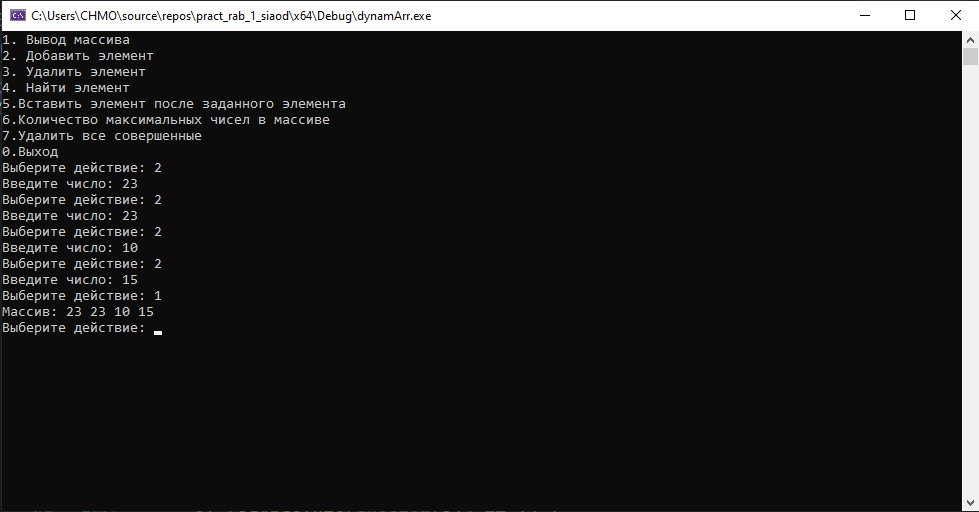


Рисунок 2. Тестирование первых двух функций на динамическом массиве

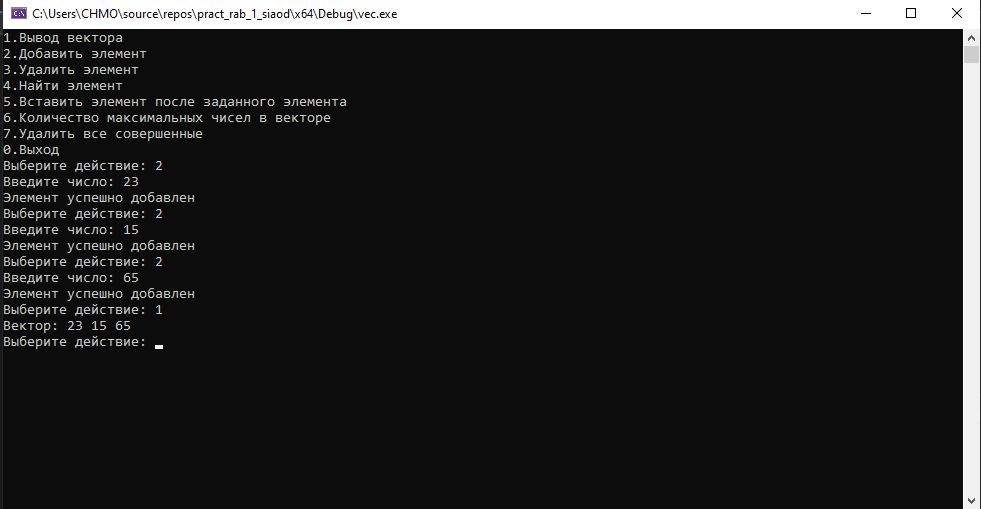


Рисунок 3. Тестирование первых двух функций на векторе

Теперь протестируем выполнение второй задачи на статическом массиве (рис. 4), динамическом массиве (рис. 5), векторе (рис. 6) и при отсутствии нужного элемента (рис. 7), индекс которого мы находим в первой задаче.



Рисунок 4. Тестирование второй задачи на статическом массиве

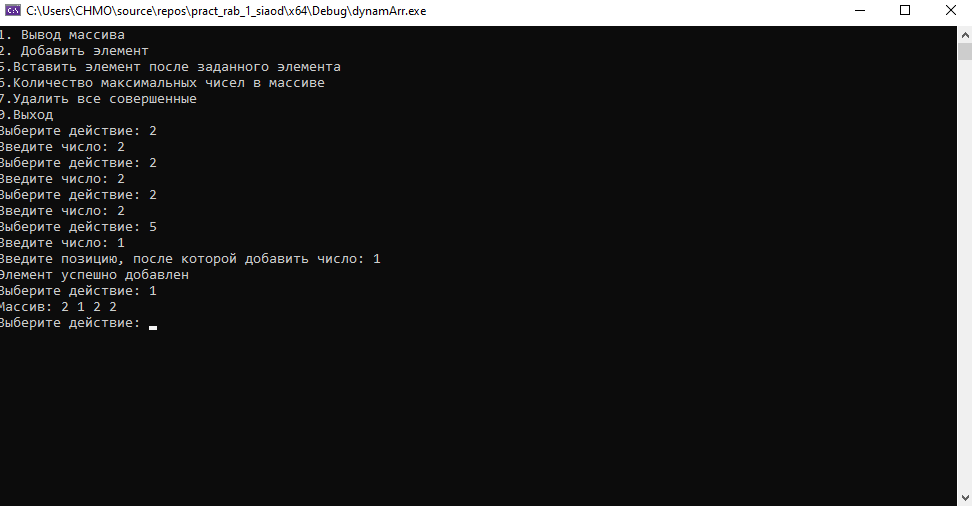


Рисунок 5. Тестирование второй задачи на динамическом массиве



Рисунок 6. Тестирование третьей функции на векторе

Далее пройдет тест третьего задания на тех же статическом (рис. 7), динамическом (рис. 8) массивах, векторе (рис. 9)

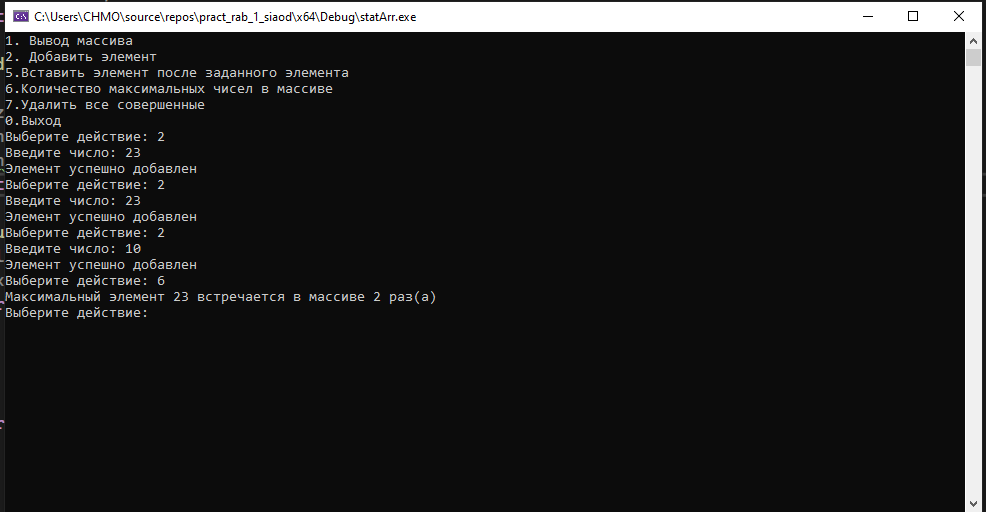


Рисунок 7. Тестирование третьей задачи на статическом массиве



Рисунок 8. Тестирование третьей задачи на динамическом массиве



Рисунок 9. Тестирование третьей задачи на векторе

Последним пройдет тестирование четвертого задания на тех же статическом (рис. 10), динамическом (рис. 11) массивах, векторе (рис. 12)

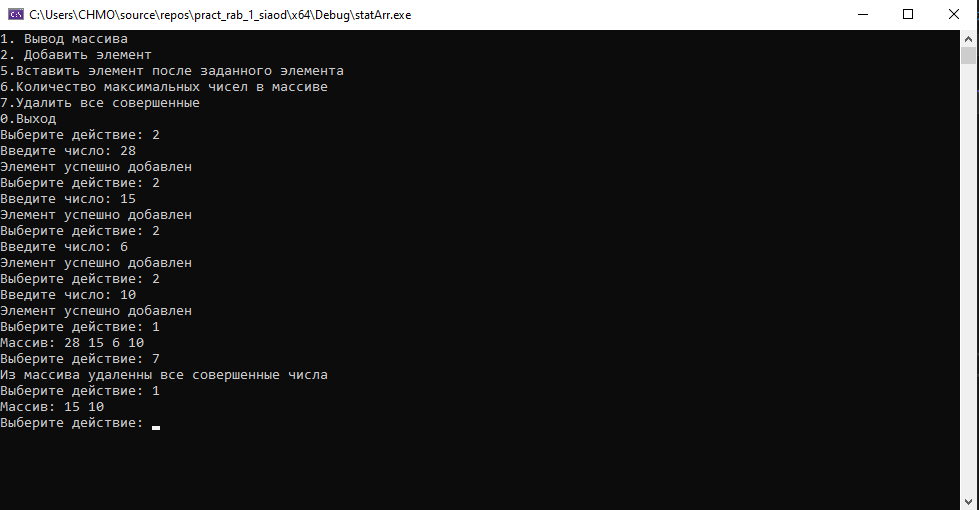


Рисунок 10. Тестирование третьей задачи на статическом массиве

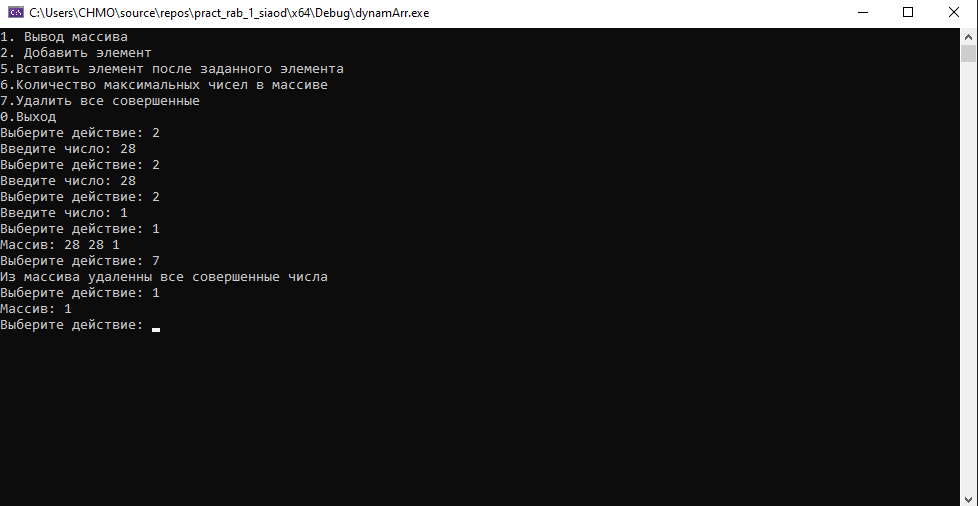


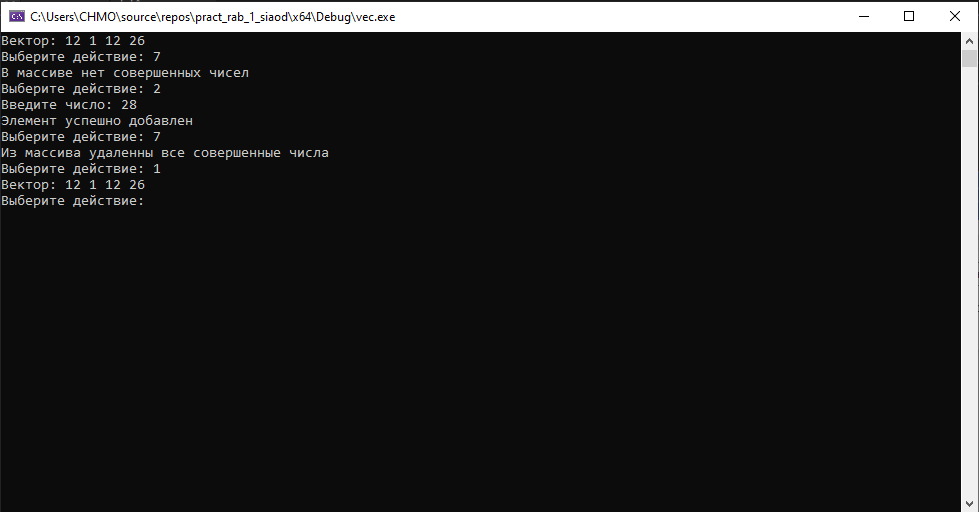
Рисунок 11. Тестирование третьей задачи на динамическом массиве 

Рисунок 12. Тестирование четвертой задачи на векторе

1. **Вывод**

В результате выполнения работы я приобрел навыки по определению одномерного массива для структуры данных задачи и для создания алгоритмов операций над одномерным массивом в языке программирования C++.

1. **Исходный код программы**

|  |
| --- |
| СТАТИЧЕСКИЙ МАССИВ  #include <iostream>  #include <format> //C++20  using namespace std;  const int maxStaticSize = 10;  int printArray(int arrayToPrint[], int size) {  cout << "Массив: ";  for (int i = 0; i < size; i++)  cout << arrayToPrint[i] << " ";  cout << endl;  return 0;  }  int addToArray(int staticArray[], int& size, int newNumber)  {  if (size >= maxStaticSize) return -1;  staticArray[size] = newNumber;  size++;  return 0;  }  int countMAX(int staticArray[], int size, int& maxchislo, int& kolvo)  {  kolvo = 0;  for (int i = 0; i < size; i++) {  int chislo = staticArray[i];  if (maxchislo < chislo) {  kolvo = 0;  }  if (staticArray[i] >= maxchislo) {  maxchislo = staticArray[i];  kolvo++;  }  }  return 0;  }  bool isAbsolute(int n) {  int summa = 0;  for (int i = 1; i < n; i++) {  if (n % i == 0) {  summa += i;  }  }  if (summa == n) {  return 1;  }  return 0;  }  int dellAllAbsolute(int staticArray[], int& size) {  if (size <= 0) return -1;  for (int i = 0; i < size; i++) {  if (isAbsolute(staticArray[i]) == 1) {  for (int z = i; z < size - 1; z++) {  staticArray[z] = staticArray[z + 1];  }  size--;  i--;  }  }  return 0;  }  int addToArrayAfterPos(int staticArray[], int& size, int newNumber, int poscion) {  if (size >= maxStaticSize) return -1;  size++;  for (int i = size - 1; i > poscion + 1; i--) {  staticArray[i] = staticArray[i - 1];  }  staticArray[poscion + 1] = newNumber;  return 0;  }  int main()  {  setlocale(0, "");  int currentStaticSize = 0;  int staticArray[maxStaticSize];  int tempNumber,pos,result,sizeCase7, raznocstCase7,maxchislo = 0,kolvo = 0;  cout << "1. Вывод массива\n"  << "2. Добавить элемент\n"  << "5.Вставить элемент после заданного элемента\n"  << "6.Количество максимальных чисел в массиве\n"  << "7.Удалить все совершенные\n"  << "0.Выход\n";  char menu;  while (true) {  cout << "Выберите действие: ";  cin >> menu;  switch (menu) {  case'1':  printArray(staticArray, currentStaticSize);  break;  case'2':  cout << "Введите число: ";  cin >> tempNumber;  if (addToArray(staticArray, currentStaticSize, tempNumber) == -1) cout << "Массив полон\n";  else cout << "Элемент успешно добавлен\n";  break;  case '5':  cout << "Введите число: ";  cin >> tempNumber;  cout << "Введите позицию, после которой добавить число: ";  cin >> pos;  if (addToArrayAfterPos(staticArray, currentStaticSize, tempNumber, pos-1) == -1) cout << "Массив полон\n";  else cout << "Элемент успешно добавлен\n";  break;  case '6':  countMAX(staticArray, currentStaticSize, maxchislo, kolvo);  cout << format("Максимальный элемент {:d} встречается в массиве {:d} раз(а)", maxchislo, kolvo) << '\n';  break;  case '7':  sizeCase7 = currentStaticSize;  dellAllAbsolute(staticArray, currentStaticSize);  raznocstCase7 = sizeCase7 - currentStaticSize;  if (raznocstCase7 != 0) {  cout << "Из массива удаленны все совершенные числа\n";  }  else  {  cout << "В массиве нет совершенных чисел\n";  }  break;  case'0':  return 0;  break;  default:  cout << "Неверный ввод!";  cout << "1. Вывод массива\n"  << "2. Добавить элемент\n"  << "3. Удалить элемент\n"  << "4. Найти\n"  << "0. выход\n"  << "Выберите действие: ";  break;  }  }  }  ДИНАМИЧЕСКИЙ МАССИВ  #include <iostream>  #include <format> //C++20  using namespace std;  int printDynArray(int arrayToPrint[], int size) {  cout << "Массив: ";  for (int i = 0; i < size; i++)  cout << arrayToPrint[i] << " ";  cout << endl;  return 0;  }  int addToDynArray(int\*& dynamicArray, int& size, int newNumber) {  size++;  dynamicArray = (int\*)realloc(dynamicArray, size \* sizeof(int));  dynamicArray[size-1] = newNumber;  return 0;  }  int countMAX(int dynamicArray[], int size, int& maxchislo, int& kolvo) {  kolvo = 0;  maxchislo = 0;  for (int i = 0; i < size; i++) {  if (dynamicArray[i] > maxchislo) {  maxchislo = dynamicArray[i];  }  }  for (int i = 0; i < size; i++) {  if (dynamicArray[i] == maxchislo) {  kolvo += 1;  }  }  return 0;  }  int isAbsolute(int n) {  int summa = 0;  for (int i = 1; i < n; i++) {  if (n % i == 0) {  summa += i;  }  }  if (summa == n) {  return 1;  }  return 0;  }  int dellAllAbsolute(int dynamicArray[], int& size) {  for (int i = 0; i < size; i++) {  if (isAbsolute(dynamicArray[i]) == 1) {  for (int z = i; z < size - 1; z++) {  dynamicArray[z] = dynamicArray[z + 1];  }  size--;  i--;  }  }  return 0;  }  int addToArrayAfterPos(int dynamicArray[], int& size, int newNumber, int position) {  size++;  for (int i = size - 1; i > position + 1; i--) {  dynamicArray[i] = dynamicArray[i - 1];  }  dynamicArray[position] = newNumber;  return 0;  }  int main()  {  setlocale(0, "");  int pos;  int currentDynamicSize = 0;  int\* dynamicArray = new int[currentDynamicSize];  int tempNumber,result,sizeCase7, raznocstCase7,maxchislo, kolvo;  cout << "1. Вывод массива\n"  << "2. Добавить элемент\n"  << "5.Вставить элемент после заданного элемента\n"  << "6.Количество максимальных чисел в массиве\n"  << "7.Удалить все совершенные\n"  << "0.Выход\n"  << "Выберите действие: ";  char menu;  while (true) {  cout << "Выберите действие: ";  cin >> menu;  switch (menu) {  case'1':  printDynArray(dynamicArray, currentDynamicSize);  break;  case'2':  cout << "Введите число: ";  cin >> tempNumber;  addToDynArray(dynamicArray, currentDynamicSize, tempNumber);  break;  case '5':  cout << "Введите число: ";  cin >> tempNumber;  cout << "Введите позицию, после которой добавить число: ";  cin >> pos;  if (addToArrayAfterPos(dynamicArray, currentDynamicSize, tempNumber, pos ) == -1) cout << "Массив полон\n";  else cout << "Элемент успешно добавлен\n";  break;  case '6':  countMAX(dynamicArray, currentDynamicSize, maxchislo, kolvo);  cout << format("Максимальный элемент {:d} встречается в массиве {:d} раз(а)", maxchislo, kolvo) << '\n';  break;  case '7':  sizeCase7 = currentDynamicSize;  dellAllAbsolute(dynamicArray, currentDynamicSize);  raznocstCase7 = sizeCase7 - currentDynamicSize;  if (raznocstCase7 != 0) {  cout << "Из массива удаленны все совершенные числа\n";  }  else  {  cout << "В массиве нет совершенных чисел\n";  }  break;  case'0':  return 0;  break;  default:  cout << "Неверный ввод!";  cout << "1. Вывод массива\n"  << "2. Добавить элемент\n"  << "5.Вставить элемент после заданного элемента\n"  << "6.Количество максимальных чисел в массиве\n"  << "7.Удалить все совершенные\n"  << "0.Выход\n"  << "Выберите действие: ";  break;  }  }  delete[] dynamicArray;  } |
|  |
| ВЕКТОР  #include <iostream>  #include <format> //C++20  #include <vector>  using namespace std;  void printVec(vector<int> sVector) {  cout << "Вектор: ";  for (int i = 0; i < sVector.size(); i++)  cout << sVector[i] << " ";  cout << endl;  }  int addToVector(vector<int>& sVector, int newNumber) {  sVector.push\_back(newNumber);  return 0;  }  int countMAX(vector<int> sVector, int& maxchislo, int& kolvo) {  kolvo = 0;  maxchislo = 0;  for (int i = 0; i < sVector.size(); i++) {  int chislo = sVector[i];  if (maxchislo < chislo) {  kolvo = 0;  }  if (sVector[i] >= maxchislo) {  maxchislo = sVector[i];  kolvo++;  }  }  return 0;  }  int isAbsolute(int n) {  int summa = 0;  for (int i = 1; i < n; i++) {  if (n % i == 0) {  summa += i;  }  }  if (summa == n) {  return 1;  }  return 0;  }  int dellAllAbsolute(vector<int>& sVector) {  for (int i = 0; i < sVector.size(); i++) {  if (isAbsolute(sVector[i]) == 1) {  sVector.erase(sVector.begin() + i);  i--;  }  }  return 0;  }  void addToVectorAfterPos(vector<int>& sVector, int newNumber, int poscion) {  sVector.resize(sVector.size() + 1);  for (int i = sVector.size() - 1; i > poscion + 1; i--) {  sVector[i] = sVector[i - 1];  }  sVector[poscion + 1] = newNumber;  }  int main()  {  setlocale(0, "");  int currentStaticSize = 0;  vector <int> sVector;  int tempNumber, pos,result,sizeCase7, raznocstCase7,maxchislo = 0,kolvo = 0;  cout << "1.Вывод вектора \n"  << "2.Добавить элемент\n"  << "5.Вставить элемент после заданного элемента\n"  << "6.Количество максимальных чисел в векторе\n"  << "7.Удалить все совершенные\n"  << "0.Выход\n";  char menu;  while (true) {  cout << "Выберите действие: ";  cin >> menu;  switch (menu) {  case'1':  printVec(sVector);  break;  case'2':  cout << "Введите число: ";  cin >> tempNumber;  addToVector(sVector, tempNumber);  cout << "Элемент успешно добавлен\n";  break;  case '5':  cout << "Введите число: ";  cin >> tempNumber;  cout << "Введите позицию, после которой добавить число: ";  cin >> pos;  addToVectorAfterPos(sVector, tempNumber, pos - 1);  cout << "Элемент успешно добавлен\n";  break;  case '6':  countMAX(sVector, maxchislo, kolvo);  cout << format("Максимальный элемент {:d} встречается в массиве {:d} раз(а)", maxchislo, kolvo) << '\n';  break;  case '7':  sizeCase7 = sVector.size();  dellAllAbsolute(sVector);  raznocstCase7 = sizeCase7 - sVector.size();  if (raznocstCase7 != 0) {  cout << "Из массива удаленны все совершенные числа\n";  }  else  {  cout << "В массиве нет совершенных чисел\n";  }  break;  case'0':  return 0;  break;  default:  cout << "Неверный ввод!";  cout << "1. Вывод массива\n"  << "2. Добавить элемент\n"  << "5.Вставить элемент после заданного элемента\n"  << "6.Количество максимальных чисел в векторе\n"  << "7.Удалить все совершенные\n"  << "0. выход\n"  << "Выберите действие: ";  break;  }  }  } |