

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**"МИРЭА - Российский технологический университет"**

**РТУ МИРЭА**

Институт информационных технологий (ИТ)

Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

# ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 3

**по дисциплине**

**«Структуры и алгоритмы обработки данных»**

Тема. Абстрактный тип данных и его реализация на

одномерном динамическом массиве и векторе

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы ИКБО-65-23 |  | Петров А. |
| Принял старший преподаватель |  | Скворцова Л.А. |

Москва 2023

# **Оглавление**

[Оглавление 2](#_Toc163680707)

[1. УСЛОВИЕ ЗАДАЧИ. 3](#_Toc163680708)

[2. ЗАДАНИЕ 1. 4](#_Toc163680709)

[2.1. Разработка абстрактного типа данных задачи, используя шаблон 4](#_Toc163680710)

[2.2. Разработка и запись на псевдокоде алгоритмов операций 7](#_Toc163680711)

[2.3. Код алгоритма на языке С++. 8](#_Toc163680712)

[2.4. Тестирование алгоритма. 12](#_Toc163680713)

[3. ЗАДАНИЕ 2. 14](#_Toc163680714)

[3.1. Коды функций операций вставки, удаления, формирования нового множества заданий 1 и 2, представленные в таблице 3: 14](#_Toc163680715)

[3.2. Код алгоритма на языке С++ 14](#_Toc163680716)

[3.3. Тестирование алгоритма. 19](#_Toc163680717)

[4. Выводы 20](#_Toc163680718)

[5. Информационные источники 21](#_Toc163680719)

# УСЛОВИЕ ЗАДАЧИ.

**Задание 1**

Реализовать АТД задачи, разработанное в практической работе 2, используя для представления значений множества динамический массив. Выполнить реализацию АТД задачи на динамическом массиве. Для управления динамической памятью использовать функции файла заголовка stdlib.h: malloc, free, realloc. Сведения по применению функций представлена в приложении 1.

**Требования к выполнению задания 1**

1. В АТД, разработанное в практической работе 2 добавить

дополнительные операции варианта, представленного в табл. 7.

2. Создать копию проекта практической работы 1.

3. Внести изменения в структуру, описывающую множество на основе типа

struct практической работы 1, в связи с применением динамического

массива.

4. Внести изменения в операции модификации множества, в связи с

применением динамического массива, реализуемого с применением

функций malloc, free, realloc.

5. Добавить в АТД операцию: добавление нового элемента в массив

значений множества и выполнить ее реализацию с применением

функций malloc, free, realloc.

6. Добавить в АТД операцию, определенную в вариантах дополнительного

задания этой практической работы в табл. 7 и выполнить ее реализацию.

7. Каждая операция должна быть оформлена отдельной функцией с

параметрами.

Примечание. Реализацию АТД можно выполнить в отдельном файле

заголовка, разрабатываемого проекта. В основной программе его нужно будет

подключить.

8. Разработать набор тестов для тестирования всех операций.

9. Разработать основную программу, демонстрирующую выполнение всех

операций над структурой данных на подготовленных тестах. Программа

должна управлять всем процессом посредством текстового меню.

10. Выполнить тестирование основной программы на представленных

тестах. Представить скрины результатов тестирования.

**Вариант задачи № 22**

Сформировать новый массив из чисел исходного, которые делятся на каждую

цифру числа.

# ЗАДАНИЕ 1.

## 2.1. Разработка абстрактного типа данных задачи, используя шаблон

|  |
| --- |
| АТД MyStruct  {  Данные:  size – количество элементов множества.  А – список значений элементов множества. |

Операции:

|  |
| --- |
| **Операция 1**  //Операция Создание массива в структуре.  //Предусловие: А – список значений элементов множества, n - количество элементов множества.  createArray (A);.- заголовок  ДОБАВИТЬ ПОСТУСЛОВИЕ И В ФУНКЦИЮ ВСЕ ПЕРМЕННЫЕ |

|  |
| --- |
| **Операция 2**  //Операция: Удаление массива в структуре.  //Предусловие: А – список значений элементов множества,  deleteArray (A, );.- заголовок |

|  |
| --- |
| **Операция 3**  //Операция: заполнение структуры данных значениями.  //Предусловие: А – входные данные, temp входные данные  fill (A, temp);.- заголовок |

|  |
| --- |
| **Операция 4**  //Операция: вывод структуры в консоль.  //Предусловие: А – входные данные, size > 0.  //Постусловие: вывод множества A  print(A);.- заголовок |

|  |
| --- |
| **Операция 5**  //Операция: удалить элемент в заданной позиции.  //Предусловие: pos – входные данные, size >0.  //Постусловие: результат при успешном поиске – 0, при безуспешном код завершения -1.  deletePos(A, pos);.- заголовок |

|  |
| --- |
| **Операция 6**  //Операция: вставить элемент в заданную позицию.  //Предусловие: pos, element – входные данные, size >0.  //Постусловие: результат при успешном поиске – element, при безуспешном код завершения -1.  addPos(A, pos, element);.- заголовок |

|  |
| --- |
| **Операция 7**  //Операция: Найти позицию элемента массива значение которого делится на каждую из цифр числа.\*  //Предусловие: temp2 – входные данные, size >0.  //Постусловие: результат при успешном поиске – pos, при безуспешном код завершения -1.  findElement(A, temp2);.- заголовок |

|  |
| --- |
| **Операция 8**  //Операция: Вставить в массив новый элемент после элемента, значение которого делится на каждую цифру значения.  //Предусловие: element, temp2 – входные данные, size >0.  //Постусловие: результат при успешном поиске – (pos+1), при безуспешном код завершения -1.  addElement (A, temp2, element,);.- заголовок |

|  |
| --- |
| **Операция 9**  //Операция: Удалить из массива все элементы, кратные трем.  //Предусловие: size >0.  //Постусловие: результат при успешном удаление – 1, при безуспешном код завершения -1.  deleteKrat3(A);.- заголовок |

|  |
| --- |
| **Операция 10**  //Операция: Удалить из массива все элементы, кратные трем.  //Предусловие: А, temp2 – входные данные.  //Постусловие: Старый массив заменяется на новый  newArray (A, temp2);.- заголовок |

## 2.2. Разработка и запись на псевдокоде алгоритмов операций

Таблица 1. Форма представления операции 1 при получении функции зависимости количества выполняемых инструкций от размера задачи

|  |  |  |
| --- | --- | --- |
| Номер инструкции | Код алгоритма newArray | Количество операций в инструкции |
|  | pos←0 | 1 |
|  | count ←0 | 1 |
|  | Int\* B ← (int\*)malloc(s.size \* sizeof(int)) | 4 |
|  | For i←0 to size do | 1+3(size +1) |
|  | pos← findElement(s, temp2) | Size(6+5size+14size ^2) |
|  | if (pos != -1) | size |
|  | B[count] ← s.A[pos] | 3size |
|  | count++; | size |
|  | deletePos(s, pos); | Size(6+6 size -6pos) |
|  | od | - |
|  | s.size ← count; | 1 |
|  | free((void\*)s.A); | 1 |
|  | B ← (int\*)realloc(B, s.size \* sizeof(int)); | 4 |
|  | s.A ← B; | 1 |

T(Size) = 17+20size+ 11Size^2+ 14size^3 - Size6pos

Поскольку pos будет какой-то константой то алгоритм будет зависеть только от переменной size.

Асимптотическая временная сложность кубическая – O(Size^3).

## 2.3. Код алгоритма на языке С++.

|  |
| --- |
| #include <vector>  #include <iostream>  #include <cstdlib>  #include <iostream>  #include <stdlib.h>  #include <ctime>  using namespace std;  typedef unsigned int typesize;  typedef int typeitem;  struct MyStruct  {  typesize size;  int\* A;  };  //создание массива в структуре  void createArray(MyStruct& s, int n);  //удаление массива в структуре  void deleteArray(MyStruct& s);  //заполнение структуры данных значениями  void fill(MyStruct& s, int temp);  //вывод структуры в консоль  void print(MyStruct s);  //удалить элемент в заданной позиции  int deletePos(MyStruct& s, typesize pos);  //вставить элемент в заданную позицию  int addPos(MyStruct& s, typesize pos, int element);  //Найти позицию элемента массива значение которого делится на каждую из цифр числа\*  int findElement(MyStruct& s, int temp2);  //Вставить в массив новый элемент после элемента, значение которого делится на каждую цифру значения  int addElement(MyStruct& s, int temp2, int element);  //Удалить из массива все элементы, кратные трем  void deleteKrat3(MyStruct& s);  //Сформировать новый массив из чисел исходного, которые делятся на каждую цифру числа.  void newArray(MyStruct& s, int temp2);  // текстовое меню для управления программой  void menu(MyStruct& A);  int main()  {  setlocale(LC\_ALL, "Rus");  MyStruct A;  menu(A);  }  void menu(MyStruct& A)  {  cout << "Доступные функции программы" << endl  << "1. создание массива в структуре - createArray" << endl  << "2. удаление массива в структуре - deleteArray" << endl  << "3. заполнение структуры данных значениями - fill" << endl  << "4. вывод структуры в консоль - print" << endl  << "5. удалить элемент в заданной позиции - deletePos" << endl  << "6. вставить элемент в заданную позицию - addPos" << endl  << "7. Найти позицию элемента массива значение которого делится на каждую из цифр числа - findElement" << endl  << "8. Вставить в массив новый элемент после элемента, значение которого делится на каждую цифру значения - addElement" << endl  << "9. Удалить из массива все элементы, кратные трем - deleteKrat3" << endl  << "10. Сформировать новый массив из чисел исходного, которые делятся на каждую цифру числа - newArray" << endl  << "0. Выход" << endl << endl;  int choice, pos, element;  while (true)  {  cout << "Выберете функцию программы = ";  cin >> choice;  switch (choice)  {  case 1:  cout << "Количество элементов в массиве = ";  cin >> element;  createArray(A, element);  break;  case 2:  deleteArray(A);  break;  case 3:  cout << "Введите 0 или 1 для выбора метода заполнения вручную или с помощью случайных чисел = ";  cin >> element;  fill(A, element);  break;  case 4:  print(A);  break;  case 5:  cout << "Введите позицию элемента = ";  cin >> pos;  deletePos(A, pos);  break;  case 6:  cout << "Введите позицию элемента = ";  cin >> pos;  cout << "Введите элемент который надо вставить = ";  cin >> element;  addPos(A, pos, element);  break;  case 7:  cout << "Введите число на которое надо делить = ";  cin >> element;  findElement(A, element);  break;  case 8:  cout << "Введите какой элемент нужно вставить = ";  cin >> pos;  cout << "Введите число на которое надо делить = ";  cin >> element;  addElement(A, pos, element);  break;  case 9:  deleteKrat3(A);  break;  case 10:  cout << "Введите число на которое надо делить = ";  cin >> element;  newArray(A, element);  break;  case 0:  cout << "Выход" << endl << endl;  return;  default:  cout << "Не правильный выбор" << endl;  break;  }  }    }  void createArray(MyStruct& s, int n)  {  s.size = n;  if (s.size < 0)  {  cout << "Вы ввели не правильное значение n";  return;  }  s.A = (int\*)malloc(s.size \* sizeof(int));  }  void deleteArray(MyStruct& s)  {  free(s.A);  }  void fill(MyStruct& s, int temp)  {  typesize temp1;  if(s.size < 0)  {  cout << "В массиве нет ячеек";  return;  }  if (temp == 0)  {  for (typesize i = 0; i < s.size; i++)  {  cout << "Введите значение " << i << endl;  cin >> temp1;  s.A[i] = temp1;  }  }  else if (temp == 1)  {  for (typesize i = 0; i < s.size; i++)  s.A[i] = rand();  }  else  {  cout << "Вы ввели не правильное значение нужно 0 или 1" << endl;  return;  }  }  void print(MyStruct s)  {  cout << "Множество чисел \n";  for (typesize pos = 0; pos < s.size; pos++)  {  cout << "Элемент " << pos << " = " << s.A[pos] << endl;  }  }  int deletePos(MyStruct& s, typesize pos)  {  if (pos <= s.size && pos >= 0)  {  for (typesize i = pos; i < s.size; i++)  {  s.A[i] = s.A[i + 1];  }  s.size--;  s.A = (int\*)realloc(s.A, s.size \* sizeof(int));  return 0;  }  else  {  cout << "нет заданной позиции" << endl;  return -1;  }  }  int addPos(MyStruct& s, typesize pos, int element)  {  if (pos <= s.size && pos >= 0)  {  s.size++;  s.A = (int\*)realloc(s.A, s.size \* sizeof(int));  for (int i = s.size; i > pos; i--)  {  s.A[i] = s.A[i-1];  }  s.A[pos] = element;  return 0;  }  else  {  cout << "нет заданной позиции" << endl;  return -1;  }  }  int findElement(MyStruct& s, int temp2)  {  int temp1 = 1;  int t;  for (int pos = 0; pos < s.size; pos++)  {  t = temp2;  for (int i = 10; t != 0; i \*= 10)  {  temp1 = t % 10;  if ((s.A[pos] % temp1) != 0)  {  break;  }  if ((s.A[pos] % temp1 == 0) && (t < 10))  return pos;  t = temp2 / i;  }  }  return -1;  }  int addElement(MyStruct& s, int temp2, int element)  {  int temp1 = 1;  int t;  for (int pos = 0; pos < s.size; pos++)  {  t = temp2;  for (int i = 10; t != 0; i \*= 10)  {  temp1 = t % 10;  if ((s.A[pos] % temp1) != 0)  {  break;  }  if ((s.A[pos] % temp1 == 0) && (t < 10))  {  addPos(s, pos + 1, element);  return pos + 1;  }  t = temp2 / i;  }  }  return -1;  }  void deleteKrat3(MyStruct& s)  {  for (int pos = 0; pos < s.size; pos++)  {  if (s.A[pos] % 3 == 0)  deletePos(s, pos);  }  }  void newArray(MyStruct& s, int temp2)  {  int pos;  int count = 0;  int\* B = (int\*)malloc(s.size \* sizeof(int));  for (int i = 0; i < s.size; i++)  {  pos = findElement(s, temp2);  if (pos != -1)  {  B[count] = s.A[pos];  count++;  deletePos(s, pos);  }  }  s.size = count;  free((void\*)s.A);  B = (int\*)realloc(B, s.size \* sizeof(int));  s.A = B;  } |

## 2.4. Тестирование алгоритма.

Составленные тесты (Таблица 2), которые обрабатывают базовый случай для размерности массива равной 20 (n = 20).

Таблица 2 - Таблица тестирования

|  |  |  |
| --- | --- | --- |
| заполнение структуры данных значениями newArray (A, temp2) | | |
| Номер теста | Входные данные | Эталон результата |
| 1 | createArray (A, 20)  fill(A, 1)  print(A)  newArray(A, 4);  print(A) | Элемент 0 = 26500  Элемент 1 = 5705  Элемент 2 = 28145  Элемент 3 = 2995 |

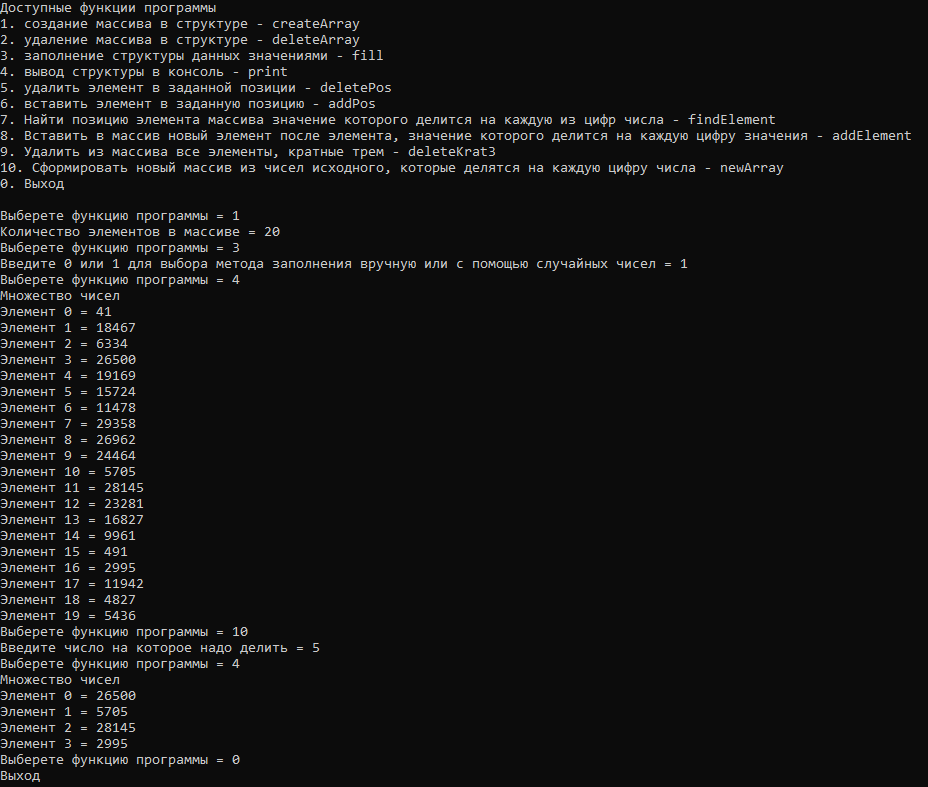


Рисунок 1 - Скриншот тестирования

# 3. ЗАДАНИЕ 2.

## 3.1. Коды функций операций вставки, удаления, формирования нового множества заданий 1 и 2, представленные в таблице 3:

Таблица 3.

|  |  |  |
| --- | --- | --- |
| Операция | Коды функций задания 1 | Коды функций задания 2 |
| Вставить элемент | s.size++;  s.A = (int\*)realloc(s.A, s.size \* sizeof(int));  for (int i = s.size; i > pos; i--)  {  s.A[i] = s.A[i-1];  }  s.A[pos] = element;  return 0; | auto iter = s.A.cbegin();  s.A.emplace(iter+pos,element);  s.size = s.A.size();  return 0; |
| Удалить элемент | for(typesize i = pos; i<s.size;i++)  {  s.A[i] = s.A[i + 1];  }  s.size--;  s.A=(int\*)realloc(s.A,s.size\*sizeof(int));  return 0; | auto iter = s.A.cbegin();  s.A.erase(iter + pos);  s.size = s.A.size();  return 0; |
| Формирование нового множества | int pos;  int count = 0;  int\* B = (int\*)malloc(s.size \* sizeof(int));  for (int i = 0; i < s.size; i++)  {  pos = findElement(s, temp2);  if (pos != -1)  {  B[count] = s.A[pos];  count++;  deletePos(s, pos);  }  }  s.size = count;  free((void\*)s.A);  B = (int\*)realloc(B, s.size \*sizeof(int));  s.A = B; | int pos;  int count = 0;  vector<int> B (s.A.size());  for (int i = 0; i <s.size;i++)  {  pos = findElement(s, temp2);  if (pos != -1)  {  B[count] = s.A[pos];  count++;  deletePos(s, pos);  }  }  s.size = count;  B.resize(s.size);  s.A.assign(B.begin(),B.end()); |

## 3.2. Код алгоритма на языке С++

|  |
| --- |
| #include <vector>  #include <iostream>  #include <cstdlib>  #include <iostream>  #include <stdlib.h>  #include <ctime>  using namespace std;  typedef unsigned int typesize;  typedef int typeitem;  struct MyStruct  {  typesize size;  vector<int> A;  };  //создание массива в структуре  void createArray(MyStruct& s, int n);  //удаление массива в структуре  void deleteArray(MyStruct& s);  //заполнение структуры данных значениями  void fill(MyStruct& s, int temp);  //вывод структуры в консоль  void print(MyStruct s);  //удалить элемент в заданной позиции  int deletePos(MyStruct& s, typesize pos);  //вставить элемент в заданную позицию  int addPos(MyStruct& s, typesize pos, int element);  //Найти позицию элемента массива значение которого делится на каждую из цифр числа\*  int findElement(MyStruct& s, int temp2);  //Вставить в массив новый элемент после элемента, значение которого делится на каждую цифру значения  int addElement(MyStruct& s, int temp2, int element);  //Удалить из массива все элементы, кратные трем  void deleteKrat3(MyStruct& s);  //Сформировать новый массив из чисел исходного, которые делятся на каждую цифру числа.  void newArray(MyStruct& s, int temp2);  // текстовое меню для управления программой  void menu(MyStruct& A);  int main()  {  setlocale(LC\_ALL, "Rus");  MyStruct A;  menu(A);  }  void menu(MyStruct& A)  {  cout << "Доступные функции программы" << endl  << "1. создание массива в структуре - createArray" << endl  << "2. удаление массива в структуре - deleteArray" << endl  << "3. заполнение структуры данных значениями - fill" << endl  << "4. вывод структуры в консоль - print" << endl  << "5. удалить элемент в заданной позиции - deletePos" << endl  << "6. вставить элемент в заданную позицию - addPos" << endl  << "7. Найти позицию элемента массива значение которого делится на каждую из цифр числа - findElement" << endl  << "8. Вставить в массив новый элемент после элемента, значение которого делится на каждую цифру значения - addElement" << endl  << "9. Удалить из массива все элементы, кратные трем - deleteKrat3" << endl  << "10. Сформировать новый массив из чисел исходного, которые делятся на каждую цифру числа - newArray" << endl  << "0. Выход" << endl << endl;  int choice, pos, element;  while (true)  {  cout << "Выберете функцию программы = ";  cin >> choice;  switch (choice)  {  case 1:  cout << "Количество элементов в массиве = ";  cin >> element;  createArray(A, element);  break;  case 2:  deleteArray(A);  break;  case 3:  cout << "Введите 0 или 1 для выбора метода заполнения вручную или с помощью случайных чисел = ";  cin >> element;  fill(A, element);  break;  case 4:  print(A);  break;  case 5:  cout << "Введите позицию элемента = ";  cin >> pos;  deletePos(A, pos);  break;  case 6:  cout << "Введите позицию элемента = ";  cin >> pos;  cout << "Введите элемент который надо вставить = ";  cin >> element;  addPos(A, pos, element);  break;  case 7:  cout << "Введите число на которое надо делить = ";  cin >> element;  findElement(A, element);  break;  case 8:  cout << "Введите какой элемент нужно вставить = ";  cin >> pos;  cout << "Введите число на которое надо делить = ";  cin >> element;  addElement(A, pos, element);  break;  case 9:  deleteKrat3(A);  break;  case 10:  cout << "Введите число на которое надо делить = ";  cin >> element;  newArray(A, element);  break;  case 0:  cout << "Выход" << endl << endl;  return;  default:  cout << "Не правильный выбор" << endl;  break;  }  }  }  void createArray(MyStruct& s, int n)  {  s.size = n;  if (s.size < 0)  {  cout << "Вы ввели не правильное значение n";  return;  }  s.A.reserve(n);  }  void deleteArray(MyStruct& s)  {  s.A.clear();  s.A.resize(0);  }  void fill(MyStruct& s, int temp)  {  typesize temp1;  if (s.size < 0)  {  cout << "В массиве нет ячеек";  return;  }  if (temp == 0)  {  for (typesize i = 0; i < s.size; i++)  {  cout << "Введите значение " << i << endl;  cin >> temp1;  s.A.emplace\_back(temp1);  }  }  else if (temp == 1)  {  for (typesize i = 0; i < s.size; i++)  s.A.emplace\_back(rand());  }  else  {  cout << "Вы ввели не правильное значение нужно 0 или 1" << endl;  return;  }  }  void print(MyStruct s)  {  cout << "Множество чисел \n";  for (typesize pos = 0; pos < s.size; pos++)  {  cout << "Элемент " << pos << " = " << s.A[pos] << endl;  }  }  int deletePos(MyStruct& s, typesize pos)  {  if (pos <= s.size && pos >= 0)  {  auto iter = s.A.cbegin(); // константный итератор указывает на первый элемент  s.A.erase(iter + pos);  s.size = s.A.size();  return 0;  }  else  {  cout << "нет заданной позиции" << endl;  return -1;  }  }  int addPos(MyStruct& s, typesize pos, int element)  {  if (pos <= s.size && pos >= 0)  {  auto iter = s.A.cbegin(); // константный итератор указывает на первый элемент  s.A.emplace(iter + pos, element);  s.size = s.A.size();  return 0;  }  else  {  cout << "нет заданной позиции" << endl;  return -1;  }  }  int findElement(MyStruct& s, int temp2)  {  int temp1 = 1;  int t;  for (int pos = 0; pos < s.size; pos++)  {  t = temp2;  for (int i = 10; t != 0; i \*= 10)  {  temp1 = t % 10;  if ((s.A[pos] % temp1) != 0)  {  break;  }  if ((s.A[pos] % temp1 == 0) && (t < 10))  return pos;  t = temp2 / i;  }  }  return -1;  }  int addElement(MyStruct& s, int temp2, int element)  {  int temp1 = 1;  int t;  for (int pos = 0; pos < s.size; pos++)  {  t = temp2;  for (int i = 10; t != 0; i \*= 10)  {  temp1 = t % 10;  if ((s.A[pos] % temp1) != 0)  {  break;  }  if ((s.A[pos] % temp1 == 0) && (t < 10))  {  addPos(s, pos + 1, element);  return pos + 1;  }  t = temp2 / i;  }  }  return -1;  }  void deleteKrat3(MyStruct& s)  {  for (int pos = 0; pos < s.size; pos++)  {  if (s.A[pos] % 3 == 0)  deletePos(s, pos);  }  }  void newArray(MyStruct& s, int temp2)  {  int pos;  int count = 0;  vector<int> B (s.A.size());  for (int i = 0; i < s.size; i++)  {  pos = findElement(s, temp2);  if (pos != -1)  {  B[count] = s.A[pos];  count++;  deletePos(s, pos);  }  }  s.size = count;  B.resize(s.size);  s.A.assign(B.begin(), B.end());  } |

## 3.3. Тестирование алгоритма.

Составленные тесты (Таблица 4), которые обрабатывают базовый случай для размерности массива равной 20 (n = 20).

Таблица 4 - Таблица тестирования

|  |  |  |
| --- | --- | --- |
| заполнение структуры данных значениями newArray (A, temp2) | | |
| Номер теста | Входные данные | Эталон результата |
| 1 | createArray (A, 20)  fill(A, 1)  print(A)  newArray(A, 4);  print(A) | Элемент 0 = 26500  Элемент 1 = 5705  Элемент 2 = 28145  Элемент 3 = 2995 |

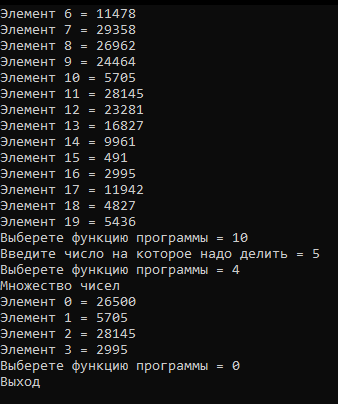


Рисунок 2 - Скриншот тестирования

# 4. Выводы

В ходе самостоятельной работы была изучена технология разработки алгоритмов на основе абстрактных типов данных. Были приобретены умения и навыки в определении и реализации абстрактного типа данных задачи на одномерном статическом массиве, который является одной из наиболее эффективных структур представления данных в программе.

Также были приобретены навыки по реализации алгоритмов операций над массивом через аппарат функций языка С++. Были изучены операции модификации массива, такие как вставка и удаление элементов, а также другие операции обработки массива как структуры данных.

В результате работы было достигнуто основная цель - освоение технологии разработки программ на основе абстрактного типа данных и приобретение необходимых навыков для его реализации на одномерном статическом массиве.

# 5. Информационные источники

1. Приложение к практическим работам – СДО (online-edu.mirea.ru)

2. Структуры и алгоритмы обработки данных – методические указания / Скворцова Л.А., Филатов А.С., Гусев К.В., Трушин С.М. - Москва, МИРЭА - Российский технологический университет, 2023.