

МИНОБРНАУКИ РОССИИ

*Федеральное государственное бюджетное образовательное учреждение высшего образования*

***«МИРЭА – Российский технологический университет»***

# РТУ МИРЭА

Отчет

Практическая работа №1

Дисциплина Структуры и алгоритмы обработки данных

Тема. Стандартные типы данных языка программирования для представления многоэлементных однородных структур данных задачи в программе

|  |  |
| --- | --- |
| Выполнил студент |  |
|  | Фамилия И.О. |
| Группа |  |
|  | Номер группы |

**Москва 2022**

Вариант №\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ЗАДАНИЕ 1

Разработать программу для выполнения операций на статическом массиве в отдельном проекте решения

Задачи варианта

1. Сформировать массив из чисел Армстронга (153=13+53+33)
2. Удалить из массива максимальное число Армстронга
3. Вставить новый элемент с заданным значением перед максимальным числом Армстронга..

**Требования**

1. Разработать функции для заполнения и отображения значений массива.
2. Выполнить декомпозицию задач варианта (смотри пример оформления отчета).
3. Выполнить разработку задачи варианта, отмеченной символом \* или любой из трех задач варианта, В разработку включить этапы.
4. Постановка задачи
5. Модель решения

Выполнить декомпозицию задачи. Описать ваш подход к решению каждой выявленной подзадачи (может быть и одна подзадача),

Если решение задачи можно описать на языке математике, то приведите математический аппарат ее решения.

Приведите прототипы функций каждой подзадачи и опишите постусловие и постусловие.

1. Разработать алгоритм для каждой подзадачи и алгоритм основной задачи.
2. Реализовать функции подзадач и основной функции.
3. Отладить каждую функцию на успешных и безуспешных тестах. Результаты представить в форме скриншотов.
4. **Разработка программы задачи 1**
   1. **Постановка задачи**

Дано. Дан массив из n элементов целого типа long.

Результат. Сформировать массив из чисел Армстронга (153=13+53+33)

Ограничения. Массив натуральных чисел

* 1. **Описание модели решения**

Исходный массив А статический максимального размера N=100.

Текущий размер n массива А определяет пользователь n<=N.

Новый массив B, который формируется программой – статический максимального размера N, фактический размер массива nB будет определен программой.

Математическая модель определения числа Армстронга

Пусть Х исходное число, проверяемое на число Армстронга

Степень равна количеству цифр в числе -p .

Найти сумму возведенных в степень p цифр числа Х это S и сравнить исходное число Х с полученной суммой S. Если X=S то нашли число Армстронга.

Описание процесса формирования массива чисел Армстронга.

Каждое найденное число вставляется в массив В по индексу nB и после вставки nB увеличивается. nB – индекс элемента, в который вставляется значение и текущий размер массива В.

* 1. **Декомпозиция – список алгоритмов, которые требуется разработать в соответствии** с исследованной моделью
     1. Список подзадач

1. Определение количества цифр в числе
2. Возведение целого числа в степень
3. Определение числа Армстронга
4. Формирование нового массива из чисел исходного
   * 1. Определение прототипов функций
5. Заполнение исходного массива значениями с клавиатуры

Предусловие. n – число заполняемых элементов,0 ≤n≤Max, где MAX – максимальное число элементов, L – массив с переменной верхней границей.

Постусловие. Заполненный массив из n элементов

void input\_ar(Array\_Static L[], int n);

1. Вывод значений массива

Предусловие. n>0

Постусловие. Вывод значений массива

void output\_ar(Array\_Static L[], int n);

1. Функции декомпозиции

***//Определение количества цифр в числе***

Предуслвие. x≥10

Постусловие. Результат целое, сумма цифр

int count(long x);

***//Возведение целого числа в степень p***

Предуслвие. а>0 – цифра числа, p>0 – количество цифр в числе

Постусловие. Результат целое, возведение а в степень p

long double pow\_1(unsigned short a, unsigned short p);

***//Определение числа Армстронга***

Предуслвие. х>0 целое число,

Постусловие. True если х число Армстронга, false иначе

bool Armstrong(long x);

***//Формирование массива чисел Армстронга***

Предусловие. A исходный массив размера0<n<=N. В – указатель на массив, подготовленный в функции main под результат.

Постусловие. Результат массив В из nB элементов. Если в массиве А нет чисел Армстронга, то nB=0 и массив В пустой.

void newArrayB(long \*A, int n, long \*B, int &nB);

***//Найти максимальное число Армстронга***

Предусловие. В – массив чисел Армстронга, nB – размер массива В

Постусловие. Результат максимальное значение массива В

long maxArmstronga(long \*B, int nB);

***//Удалить максимальное число Армстронга из исходного массива А***

Предусловие. A исходный массив размера 0<n<=N. max – максимальное число Армстронга в массиве А

Постусловие. Результат массив А без одного значения – максимального числа Армстронга

void delete\_element(long \*A, int &n, long max);

***//Вставить новый элемент перед максимальным числом Армстронга*** Предусловие. A исходный массив размера 0<n<=N. max – максимальное число Армстронга в массиве А, х – значение нового массива

Постусловие. Результат массив А увеличенный на один элемент

void insert\_element(long \*A, int &n, long max);

* 1. Разработка алгоритмов операций и представление его **на псевдокоде**

Алгоритмы задач декомпозиции

1. Алгоритм функции Armstrong(long x);

Armstrong(long x){

int p←count(x); //количество цифр в числе и значение степени

int sum←0;

long copyx←x; //копия исходного числа

while(x!=0)

do

sum←sum+pow\_1(x%10,p);

x=x/10;

od

if (copyx=sum)

then результат true

else результат false

}

1. Количество цифр в числе

count(x)

{

k ← 1;

while (x > 10) do

k←k+1;

понизить разрядность числа

od

вернуть k;

}

1. Возведение числа а в степень р

pow\_1(a, p){

pow1 ← 1;

p ←|p|;

i ← 1;

while(i <= p) do

i←i+1;

pow1 ← pow1 \* a;

od

if (p = 0)

вернуть 1;

else

if (p < 0) вернуть 1 / pow1;

else

вернуть pow1;

}

1. Формирование нового массива из чисел Армстронга массива А

newArrayB(A,n,B,&nB){

i←0;

nB=0;

while(i<n) do

if A[i] – число Армстронга

B[nB] ←A[i]

nB←nB+1;

endif

i←i+1

od

}

1. Найти максимальное число в массиве В

maxArmstronga(B,nB){

indexmax←0;

i←1;

while(i<nB) do

if(B[i]>B[indexmax]) indexmax←i;

i←i+1;

od

вернуть B[indexmax]

}

1. Удаление максимального числа Армстронга из массива А

delete\_elements(A, n, max){

Найти индекс i элемента значение которого равного max

for j=i to n-2 do

A[j]←A[j+1];

od

n←n-1

}

1. Вставить новый элемент перед максимальным числом Армстронга

insert\_elements(A, n, max){

Найти индекс i элемента значение которого равного max

for j=n downto i+1 do

A[j]←A[j-1];

od

n←n+1

}

* 1. Набор тестов для тестирования задач декомпозиции

1. Количество цифр в числе

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Результат программы | Тест пройден/не пройден |
| 1 | X=2 | 1 |  |  |
| 2 | X=12 | 2 |  |  |
| 3 | X=123 | 3 |  |  |
| 4 | X=-123 | 3 |  |  |

1. Возведение числа в степень

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Результат программы | Тест пройден/не пройден |
| 1 | a=5, p=0 | 1 |  |  |
| 2 | a=5, p=3 | 125 |  |  |
| 3 | a=5, p=-2 | Недопустимое значение p |  |  |

1. Определение числа Армстронга

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Результат программы | Тест пройден/не пройден |
| 1 | х=1 | да |  |  |
| 2 | Х=12 | нет |  |  |
| 3 | Х=153 | да |  |  |
| 4 | Х=-153 | нет |  |  |

1. Формирование массива В из чисел Армстронга массива А

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Результат программы | Тест пройден/не пройден |
| 1 | n=3  А(1, 2, 3) | nB=3  B(1,2,3) |  |  |
| 2 | n=5  А(11, 12, 13,14, 15) | nB=0  Нет таких чисел |  |  |
| 3 | n=3  А(1,22,153) | nB=2  B(1,153) |  |  |
| 4 | n=3  А(1,22,-153) | nB=1  B(1) |  |  |

1. Удаление максимального числа Армстронга массива А

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Результат программы | Объем памяти массива А до/после при N=100 | Тест пройден/не пройден |
| 1 | n=3, max=3  А(1, 2, 3) | n=2  А(1,2) |  | 400/400 |  |
| 2 | n=5  А(11, 12, 153,14, 15) | N=4  А(11, 12, 14, 15) |  | 400/400 |  |

1. Вставить новый элемент перед максимальным числом Армстронга массива А

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Результат программы | Объем памяти массива А до/после при N=100 | Тест пройден/не пройден |
| 1 | n=3, max=3, х=10  А(1, 2, 3) | n=4  А(1,2, 10, 3) |  | 400/400 |  |
| 2 | n=5  А(153, 12, 13,14, 15) | n=6  А(10,153, 12, 13, 14, 15) |  | 400/400 |  |

1. **Код программы**

// Armstronga.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include <iostream>

#include"malloc.h"

#include"string.h"

using namespace std;

const int N = 100;

//Предусловие. n - размер массива - целое число. 0<n<=N. L - массив

//Постусловие.вывод массива в консоль

void output\_ar(long \*L, int n);

//Предусловие. L - массив, 0<n<=N. - размер массива - целое число.

//Постусловие.Заполненный массив L

void input\_ar(long \*L, int n);

//Вставить новый элемент перед элементом, содержащим максимальное число Армстронга, содержащееся в массиве

// сохраняя порядок следования остальных элементов массива.

//Предусловие. L - массив, 0<n<N. - размер массива - целое число.

//x - целое число, вставляемое в массив

//Постусловие. Результат - измененный массив размером n+1<=N

void insert\_elems(long\* L, int& n, long max, long x);

//Предусловие. L - массив, 0<n<=N. - размер массива - целое число.

//Постусловие.Результат - измененный массив c размером n-1

void delete\_elems(long\* L, int& n, long max);

//Предусловие. L - массив, 0<n<=N. - размер массива - целое число.

//nB - длина массива с числами Армстронга

//Постусловие. Результат - массив с числами Армстронга размером nB

void newArrayB(long\* A, int n, long \*B, int& nB);

//Предусловие. x - целое число

//Постусловие. результат - количества цифр в числе

int count(long x);

//Возведение целого числа a в степень p

//Предусловие. a - целое положительное число,p>=0 - целое число

long pow\_1(unsigned int a, int p);

//Определение числа Армстронга

bool Armstrong(long x);

//Предусловие: В - массив из nB чисел Армстронга

//Постусловие: максимальное число массива

long maxArmstronga(long\* B, int nB);

int main(int argc, char\* argv[])

{

//тестирование функций декомпозиции

/\*cout << count(2) << " " << count(12) << " " << count(123) << "\n";

cout << pow\_1(5, 0) << " ";

cout << pow\_1(5, 3);

cout<< " " << pow\_1(5, -2) << "\n";\*/

int n;

long A[N];

cout << "Input size array"; cin >> n;

if (n<0 || n>N) {

cout << "Error: size array n<0 || n>N"; return 1;

}

input\_ar(A, n);

//Формирование массива чисел Армстронга

long B[N];

int nB;

newArrayB(A, n, B, nB);

if (nB == 0) cout << " No numbers found Armstronga ";

else

{

cout << "Numbers found Armstronga\n";

output\_ar(B, nB);

int x = 10;

long maxArmst = maxArmstronga(B, nB);

insert\_elems(A, n, maxArmst, x); //добавили еще одно значение в массив А

output\_ar(A, n);

delete\_elems(A, n, maxArmst); //удалили элемент из массива

output\_ar(A, n);

}

return 0;

}

long\* create\_array(int& n)

{

cout << "Введите количество элементов n=";

cin >> n;

long\* ptr = new long[n];

return ptr;

}

void insert(long\*& L, int& n, long x) //вставка нового значения в массив

{

memcpy(L, L, n + 1); //увеличение размера массива –функция из string.h

L[n] = x; n++;

}

void output\_ar(long\* L, int n)

{

cout << "Arrays\n";

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

cout << L[i] << " ";

cout << endl;

}

void input\_ar(long\* L, int n)

{

cout << "Input " << n << " numbers in arrays";

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

cin>>L[i];

cout << endl;

}

long maxArmstronga(long\* B, int nB) {

int indexmax = 0;

for (int i = 1; i < nB; i++)

if (B[i] > B[indexmax]) indexmax = i;

return B[indexmax];

}

void insert\_elems(long\* L, int& n, long max, long x)

{

int indexmax,i;

for (i = 0; i < n && max != L[i]; i++);

if (i < n) {

for (int j = n; j >= i + 1; j--) L[j] = L[j - 1];

L[i] = x;

n++;

}

}

void delete\_elems(long\* L, int& n, long max) //удаление элемента из массива

{

int indexmax, i;

for (i = 0; (i < n && (max != L[i])); i++);

if (i < n) {

n--;

for (int j = i; j <n; j++) L[j] = L[j +1];

}

}

void newArrayB(long\* A, int n, long \*B, int& nB)

{

nB = 0;

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

{

if (Armstrong(A[i]) == true)

{

B[nB] = A[i];

nB++;

}

}

}

int count(long x)

{

int k = 1;

x = abs(x);

while (x > 10) {

k++;

x = x / 10;

}

return k;

}

long pow\_1(unsigned int a, int p)

{

unsigned int pow1 = 1;

int p1 = abs(p);

for (int i = 1; i <= p1; i++) pow1 = pow1 \* a;

if (p == 0)

return 1;

else

return pow1;

}

bool Armstrong(long x)

{

int p=count(x); //количество цифр в числе и значение степени

int sum=0;

long copyx=x; //копия исходного числа

while (x != 0) {

sum = sum + pow\_1(x % 10, p);

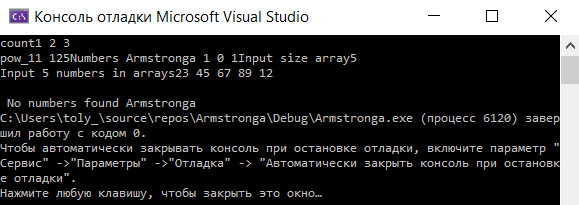
x = x / 10;

}

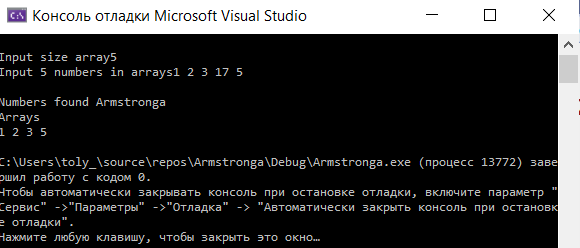
return (copyx == sum) ? true : false;

}

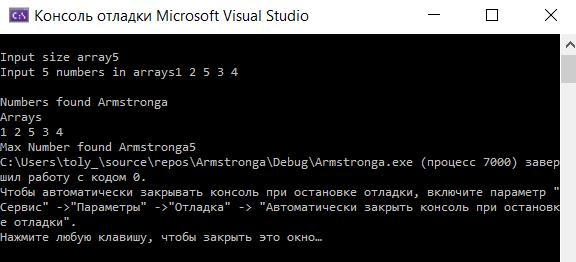
1. **Скриншоты**
   1. Тестирование функций декомпозиции: количество цифр в числе, возведение в степень, числа Армстронга



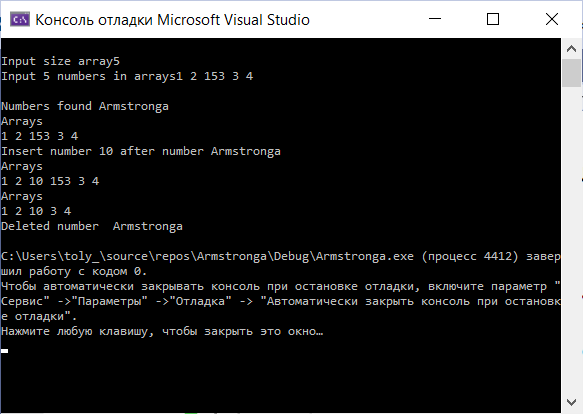
* 1. Тестирование операции формирования массива чисел Армстронга

****

1. **Тестирование операции найти максимальное значение в массиве В**

****

1. **Тестирование операций: вставить и удалить**



**ЗАДАНИЕ 2**

Разработать программу для выполнения всех операций варианта над динамическим массивом

В отдельном проекте решения, созданного в задании 1. Скопируйте в проект функции из проекта задания 1, внесите изменения в функции, изменяющие размер массива.

* + - 1. Постановка задачи.

Дано. Исходный массив А динамический из n элементов.

Результат. Динамический массив В из чисел Армстронга.

Массив А: изменение размера массива после операций вставки и удаления.

Ограничения: Массив А содержит числа Армстронга. Массив А не содержит числа Армстронга.

* + - 1. Модель решения

Операции декомпозиции, выполненные для задач варианта при применении статического массива, используются без изменения.

Изменению подлежать функции формирования массива, вставки нового значения и удаления элемента.

1. Операция Формирование массива чисел Армстронга. Массив исходный динамический, массив чисел Армстронга тоже формируется как динамический, при добавлении нового значения массив расширяется функцией realloc.
2. Операция вставки нового значения в массив А. Должна предварительно расширить массив А на один элемент функцией realloc. Для реализации используем соответствующую функцию реализации на статическом массиве.
3. Операция удаления элемента из массива. После сдвига элементов влево (удаление значения) надо освободить память последнего элемента, которая освободилась после удаления элемента. Для реализации используем соответствующую функцию реализации на статическом массиве.
   * + 1. Тесты операций
4. Создания массива В из чисел массива А

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Объем массива В в бвйтах | Тест пройден/не пройден |
| 1 | n=3  А(1, 2, 3) | nB=3  B(1,2,3) | 12 |  |
| 2 | n=5  А(11, 12, 13,14, 15) | nB=0  Нет таких чисел | 0 |  |
| 3 | n=3  А(1,22,153) | nB=2  B(1,153) | 8 |  |
| 4 | n=3  А(1,22,-153) | nB=1  B(1) | 4 |  |

1. Вставка нового значения в массив А

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Объем массива А до/после вставки в байтах | Тест пройден/не пройден |
| 1 | n=3, max=3, х=10  А(1, 2, 3) | n=4  А(1,2, 10, 3) | 12/16 |  |
| 2 | n=5  А(153, 12, 13,14, 15) | n=6  А(10,153, 12, 13, 14, 15) | 20/24 |  |

1. Удаление элемента из массива А

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер теста | Исходные данные | Ожидаемый результат | Объем массива А до/после вставки в байтах | Тест пройден/не пройден |
| 1 | n=3, max=3  А(1, 2, 3) | n=2  А(1,2) | 12/8 |  |
| 2 | n=5  А(11, 12, 153,14, 15) | n=4  А(11, 12, 14, 15) | 20/16 |  |
| 3 | n=4  A(11,12,13,14) | N=4  A(11,12,13,14) | 16/16 |  |

* + - 1. Код программы и функций управления динамическим массивом

// Armstronga.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//отключение предупреждения по использованию realloc

**#if defined(\_WIN32) || defined(\_\_WIN32\_\_) || defined(WIN32)**

**#define \_CRT\_SECURE\_NO\_WARNINGS**

**#pragma warning(disable:4996)**

#endif

#include <iostream>

#include»malloc.h»

#include»string.h»

using namespace std;

//Предусловие. N – размер массива – целое число. 0<nL – массив

//Постусловие.вывод массива в консоль

void output\_ar(long \*L, int n);

//Предусловие. L – массив, 0<n. – размер массива – целое число.

//Постусловие.Заполненный массив L

void input\_ar(long \*L, int n);

//Вставить новый элемент перед элементом, содержащим максимальное число Армстронга, содержащееся в массиве

// сохраняя порядок следования остальных элементов массива.

//Предусловие. L – массив, 0<n. – размер массива – целое число.

//x – целое число, вставляемое в массив

//Постусловие. Результат – измененный массив размером n+1

void insert\_elems(long\* &L, int& n, long max, long x);

//Предусловие. L – массив, 0<n. – размер массива – целое число.

//Постусловие.Результат – измененный массив c размером n-1

void delete\_elems(long\* &L, int& n, long max);

//Предусловие. L – массив, 0<n. – размер массива – целое число.

//nB – длина массива с числами Армстронга

//Постусловие. Результат – массив с числами Армстронга размером nB

void newArrayB(long\* A, int n, long \*&B, int& nB);

//Предусловие. X – целое число

//Постусловие. Результат – количества цифр в числе

int count(long x);

//Возведение целого числа a в степень p

//Предусловие. A – целое положительное число,p>=0 – целое число

long pow\_1(unsigned int a, int p);

//Определение числа Армстронга

bool Armstrong(long x);

//Предусловие: В – массив из nB чисел Армстронга

//Постусловие: максимальное число массива

long maxArmstronga(long\* B, int nB);

int main(int argc, char\* argv[])

{

//тестирование функций декомпозиции

/\*cout << “count”<<count(2) << “ “ << count(12) << “ “ << count(123) << “\n”;

cout << «pow\_1»<<pow\_1(5, 0) << « «<< pow\_1(5, 3);

cout << “Numbers Armstronga “ << Armstrong(2) << “ “ << Armstrong(12) << “ “ << Armstrong(153);\*/

int n;

cout << “\nInput size array”; cin >> n;

if (n<0 || n>N) {

cout << “Error: size array n<0 || n>N”; return 1;

}

long \*A=new long[n];

if (A == NULL) {

cout << “Error: There is no free memory in the heap in the specified amount”;

return 1;

}

input\_ar(A, n);

//Формирование массива чисел Армстронга

long \*B=NULL;

int nB;

newArrayB(A, n, B, nB);

if (nB == 0) cout << “ No numbers found Armstronga “;

else

{

cout << “Numbers found Armstronga\n”;

output\_ar(B, nB);

cout << “memory in a given amount”<<sizeof(B)/nB<<endl;

int x = 10;

long maxArmst = maxArmstronga(B, nB);

insert\_elems(A, n, maxArmst, x);

cout << «Insert number 10 after number Armstronga\n»;// добавили еще одно значение в массив А

output\_ar(A, n);

delete\_elems(A, n, maxArmst); //удалили элемент из массива

output\_ar(A, n);

cout<< “Deleted number Armstronga\n”;

}

return 0;

}

long\* create\_array(int& n)

{

cout << «Введите количество элементов n=»;

cin >> n;

long\* ptr = new long[n];

return ptr;

}

void insert(long\*& L, int& n, long x) //вставка нового значения в массив

{

memcpy(L, L, n + 1); //увеличение размера массива –функция из string.h

L[n] = x; n++;

}

void output\_ar(long\* L, int n)

{

cout << «Arrays\n»;

for (int I = 0; I < n; i++)

cout << L[i] << « «;

cout << endl;

}

void input\_ar(long\* L, int n)

{

cout << “Input “ << n << “ numbers in arrays”;

for (int I = 0; I < n; i++)

cin>>L[i];

cout << endl;

}

long maxArmstronga(long\* B, int nB) {

int indexmax = 0;

for (int I = 1; I < nB; i++)

if (B[i] > B[indexmax]) indexmax = I;

return B[indexmax];

}

void insert\_elems(long\* &L, int& n, long max, long x)

{

int indexmax,i;

for (I = 0; I < n && max != L[i]; i++);

if(L!=NULL){

L = (long\*)realloc((long\*)L, sizeof(long) \* (n + 1));

if (i < n) {

for (int j = n; j >= I + 1; j--)

**if (L != NULL) L[j] = L[j – 1];//только после проверки без предупр.**

L[i] = x;

n++;

}}

}

//if (p != NULL) {

// int\* tmp;

// p = (int\*)realloc((int\*)p, 4 \* sizeof(int));

// if (p != NULL) {

// p[3] = 4;

// for (int i = 0; i < 4; i++)

// std::cout << p[i];

// }

//}

void delete\_elems(long\* &L, int& n, long max) //удаление элемента из массива

{

int indexmax, i;

for (i = 0; (i < n && (max != L[i])); i++);

if (i < n) {

n--;

for (int j = i; j <n; j++) L[j] = L[j +1];

L = (long\*)realloc((long\*)L, sizeof(long) \* (n));

}

}

void newArrayB(long\* A, int n, long \*&B, int& nB)

{

nB = 0;

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

{

if (Armstrong(A[i]) == true)

{

if (nB == 0)

B = new long[1];

else

if (B=(long\*)realloc(B, sizeof(long) \* (nB + 1))){

if(B!=NULL) B[nB] = A[i];

nB++;

}

}

}

}

int count(long x)

{

int k = 1;

x = abs(x);

while (x > 10) {

k++;

x = x / 10;

}

return k;

}

long pow\_1(unsigned int a, int p)

{

unsigned int pow1 = 1;

int p1 = abs(p);

for (int i = 1; i <= p1; i++) pow1 = pow1 \* a;

if (p == 0)

return 1;

else

return pow1;

}

bool Armstrong(long x)

{

int p=count(x); //количество цифр в числе и значение степени

int sum=0;

long copyx=x; //копия исходного числа

while (x != 0) {

sum = sum + pow\_1(x % 10, p);

x = x / 10;

}

return (copyx == sum) ? true : false;

}

"

Задание 3

Использование вектора в решении всех задач варианта

* + - 1. Постановка задачи

Дано. Массив А из n целых чисел, реализованный вектором.

Результат.

Массив В сформированный на векторе

Удаление из массива А элемента.

Вставка в массив А нового значения.

* + - 1. Модель решения

1. Определим вектор А.
2. Заполним А n значениями и вернем его.
3. Сформируем массив В как вектор из чисел Армстронга массива А и вернем его.
4. Удалим максимальное число Армстронга из вектора, изменив переданный вектор.
5. Вставим в А перед максимальным числом Армстронга новый элемент и вернем измененный вектор.
   * + 1. Код реализации

// vectorArmstrong.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

//

#include <iostream>

#include <algorithm>

#include"string.h"

#include <vector>

using namespace std;

//Предусловие. n - размер массива - целое число. 0<n<=N. L - массив

//Постусловие.вывод массива в консоль

void output\_ar(vector<long> L );

//Предусловие. L - массив, 0<n<=N. - размер массива - целое число.

//Постусловие.Заполненный массив L

vector<long> input\_ar(int n);

//Вставить новый элемент перед элементом, содержащим максимальное число Армстронга, содержащееся в массиве

// сохраняя порядок следования остальных элементов массива.

//Предусловие. L - массив, 0<n<N. - размер массива - целое число.

//x - целое число, вставляемое в массив

//Постусловие. Результат - измененный массив размером n+1<=N

void insert\_elems(vector<long> &L, long max, long x);

//Предусловие. L - массив, 0<n<=N. - размер массива - целое число.

//Постусловие.Результат - измененный массив c размером n-1

void delete\_elems(vector<long> &L, long max);

//Предусловие. L - массив, 0<n<=N. - размер массива - целое число.

//nB - длина массива с числами Армстронга

//Постусловие. Результат - массив с числами Армстронга размером nB

vector<long> newArrayB(vector<long> L);

//Предусловие. x - целое число

//Постусловие. результат - количества цифр в числе

int count(long x);

//Возведение целого числа a в степень p

//Предусловие. a - целое положительное число,p>=0 - целое число

long pow\_1(unsigned int a, int p);

//Определение числа Армстронга

bool Armstrong(long x);

//Предусловие: В - массив из nB чисел Армстронга

//Постусловие: максимальное число массива

long maxArmstronga(vector<long> B);

int main(int argc, char\* argv[])

{

int n;

cout << "\nInput size array"; cin >> n;

vector<long> A;

if (A.empty()) {

A=input\_ar( n);

output\_ar(A);

}

if (!A.empty()) {

//Формирование массива чисел Армстронга

vector<long> B = newArrayB(A);

if (!B.empty()) {

cout << "Numbers found Armstronga\n";

output\_ar(B);

cout << "memory in a given amount" << B.max\_size()<< endl;

int x = 10;

long maxArmst = maxArmstronga(B);

insert\_elems(A, maxArmst, x);

cout << "Insert number 10 after number Armstronga\n";// добавили еще одно значение в массив А

output\_ar(A);

delete\_elems(A, maxArmst); //удалили элемент из массива

output\_ar(A);

cout << "Deleted number Armstronga\n";

}

}

return 0;

}

void output\_ar(vector<long> L)

{

cout << "Arrays\n";

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

cout << L[i] << " ";

cout << endl;

}

vector<long> input\_ar(int n)

{

vector<long> L;

cout << "Input " << n << " numbers in arrays";

int a;

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

cin >> a;

L.push\_back(a);

}

return L;

}

long maxArmstronga(vector<long> B) {

int indexmax = 0;

for (int i = 1; i < B.size(); i++)

if (B[i] > B[indexmax]) indexmax = i;

return B[indexmax];

}

void insert\_elems(vector<long> &L, long max, long x)

{

int indexmax, i;

for (i = 0; i < L.size() && max != L[i]; i++);

if (i <L.size()) {

L.insert(L.begin()+i, x);

}

}

void delete\_elems(vector<long> &L, long max) //удаление элемента из массива

{

int indexmax, i;

for (i = 0; (i < L.size() && (max != L[i])); i++);

if (i < L.size()) {

L.erase(L.begin() + i);

}

}

vector<long> newArrayB(vector<long> L)

{

vector<long> B;

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

{

if (Armstrong(L[i]) == true)

{

B.push\_back(L[i]);

}

}return B;

}

int count(long x)

{

int k = 1;

x = abs(x);

while (x > 10) {

k++;

x = x / 10;

}

return k;

}

long pow\_1(unsigned int a, int p)

{

unsigned int pow1 = 1;

int p1 = abs(p);

for (int i = 1; i <= p1; i++) pow1 = pow1 \* a;

if (p == 0)

return 1;

else

return pow1;

}

bool Armstrong(long x)

{

int p = count(x); //количество цифр в числе и значение степени

int sum = 0;

long copyx = x; //копия исходного числа

while (x != 0) {

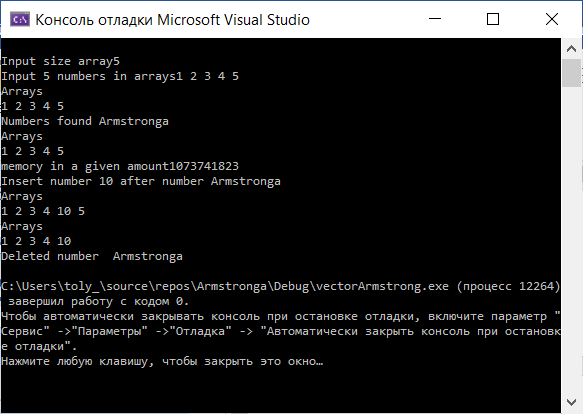
sum = sum + pow\_1(x % 10, p);

x = x / 10;

}

return (copyx == sum) ? true : false;

* }
  + - 1. Тестирование



ВЫВОДЫ

Знания, умения, навыки полученные при выполнении работы