ГУАП

КАФЕДРА № 33

ОТЧЕТ  
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | А.И. Дубинин |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2 |
| ИСПОЛЬЗОВАНИЕ ОБРАБОТЧИКОВ ИСКЛЮЧЕНИЙ ПРИ РАБОТЕ С МАССИВАМИ. |
| по курсу: Технологии и методы программирования |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 3136 |  |  |  | В.А. Панчук |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2023 г.

Оглавление

[1. Цель лабораторной работы 3](#_Toc131280288)

[2. Постановка задачи 3](#_Toc131280289)

[3. Часть 1(С++) 3](#_Toc131280290)

[3.1. Листинг программы 4](#_Toc131280291)

[3.2. Пример работы программы 5](#_Toc131280292)

[4. Часть 2(С#) 6](#_Toc131280293)

[4.1. Пункт 1 6](#_Toc131280294)

[4.1.1. Листинг алгоритма 7](#_Toc131280295)

[4.1.2. Пример работы программы 7](#_Toc131280296)

[4.2. Пункт 2 9](#_Toc131280297)

[4.2.1. Листинг программы 10](#_Toc131280298)

[4.2.1. Пример работы программы 10](#_Toc131280299)

[4.3. Пункт 3 11](#_Toc131280300)

[4.3.1. Листинг программы 12](#_Toc131280301)

[4.3.2. Пример работы программы 12](#_Toc131280302)

[5. Дополнительное задание 14](#_Toc131280303)

[5.1. Часть программы 15](#_Toc131280304)

[5.2. Пример работы программы 17](#_Toc131280305)

[6. Вывод 17](#_Toc131280306)

# Цель лабораторной работы

Получить навыки работы обработки исключений при работе с массивами, формируемыми в различных режимах. изучить средства среды программирования Microsoft Visual Studio Community 2019 для создания статистических библиотек (файлов формата \*.lib), содержащих описание ASM-модулей с поддержкой языка С/С++/С#, и их включения в основной проект.

# Постановка задачи

В лабораторной работе, задания из первой части (с использованием языков Assembler и C++) выполнялись под вариантом 3, а из второй(С#) – под вариантом 14.

В первой части лабораторной работы на языке Ассемблера реализовать алгоритм преобразования информации с использованием логических выражений или методов криптографического преобразования символьной информации в виде отдельной статической библиотеки. В модуле С/С++ выполнить подключение статистической библиотеки и вызов ASM-процедуры и обеспечить потоковый ввод/вывод внешних данных.

Вторая часть лабораторной работы включает в себя разработку программы на языке C#. Программа должна была быть выполнена с использованием интерфейса Windows Forms, причем ввод и вывод массивов производить с использованием компонента dataGridView. Обработанные исключения фиксировать на главной форме проекта и в текстовом файле с указанием даты и времени проявления исключений, а также свойств класса Exception, рассмотренных на лекции.

Интерфейс пользователя должен предлагать выбор способа формирования массива.

# Часть 1(С++)

Задача: Реализовать процедуры умножения знакового целого числа (вводится с клавиатуры) на число, равное степени 2, с использованием побитовой операции сдвига влево <<.

Блок схема алгоритма данной задачи представлена на рис.1.

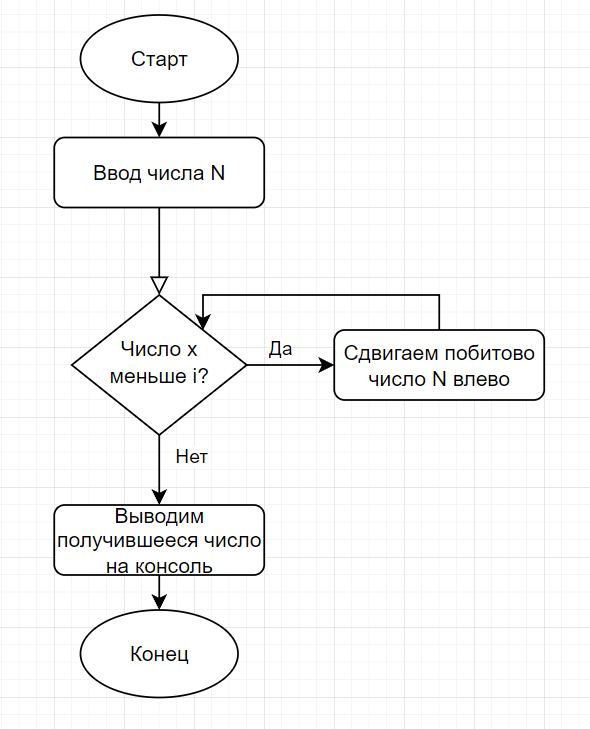


Рисунок 1− процедура умножения знакового целого числа на число, равное степени 2

## Листинг программы

На языке С++:

#include<iostream>

using namespace std;

extern "C" int mult();

extern "C" unsigned int a\_asm = 0;

int main() {

register int a, b = 32;

cin >> a;

for (int i = 1; i < b; i \*= 2)

{

a\_asm = a;

mult();

}

cout << a;

return 0;

}

На языке Assembler:

.686

.MODEL FLAT, C

.STACK 256

.DATA

; -----------Local data------------------------------

; a\_asm dw 2 ; Описание локальной переменной типа DefineWord 16 бит (2 байта)

.CODE ; Сегмент кода программы

;-----------External usage-------------------------

; Возвращаемое значение процедуры во внешнее приложение

EXTRN a\_asm : WORD

;-----------Function definitions-------------------

; Описание экспортируемой функции

PUBLIC C mult ;

mult PROC far ; Заголовок экспортируемой функции

; far - возможность обращения к процедуре из другого участка кода проекта

mov ax, a\_asm ; Помещаем значение переменной a\_asm в регистрax

shl ax, 1 ; Сложение значений двух регистров и сохранение результатов в регистре ax

mov a\_asm, ax ; Содержимое регистра ax перемещаем во внешнююпеременную c\_asm

; Оператор Return from procedure обеспеSчивает возврат управлениявызывающей программе

retn

mult ENDP

END

## Пример работы программы

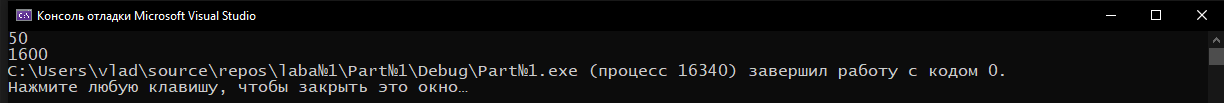


Рисунок − Пример программы 1

# Часть 2(С#)

Создать Windows Forms проект, выполняющий следующее задание:

Дан массив целых чисел (n=25), заполненный случайным образом числами из промежутка [-35, 75] (предусмотреть возможность ввода массива вручную и случайно с заданной частотой)

* удалить из него все элементы, первая цифра которых четная;
* вставить число k1 после всех элементов, больших заданного числа, а число k2 - перед всеми элементами, кратными 3 (k1 и k2 вводить с клавиатуры);
* перенести первые k элементов в конец, то есть: a[k+1], a[k+2], a[k+2], ..., a[n], a[1], a[2], ..., a[k].

## Пункт 1

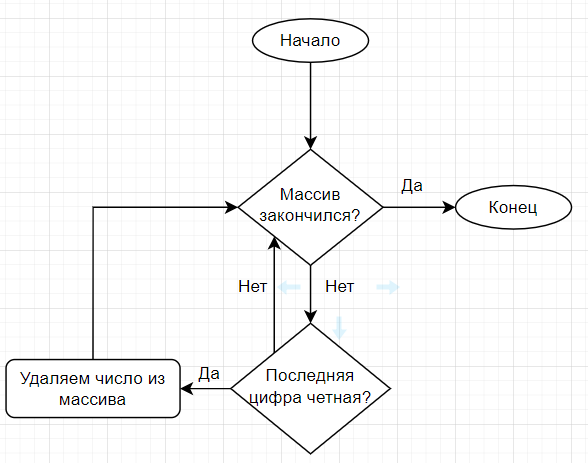
Алгоритм удаления из массива всех элементов, первая цифра которых четная, представлен на рис 3.

Рисунок – алгоритм удаления числа из массива

## Листинг алгоритма

try

{

for (int i = 0; i < (dataGridView1.RowCount - 1); i++)

{

int intValue;

bool succes = Int32.TryParse((string)dataGridView1.Rows[i].Cells[1].Value, out intValue);

if (!succes || intValue > 60 || intValue < -10) throw new Exception("Введенные данные не соответствуют требованиям");

if ((intValue % 10) != 0)

{

if ((intValue % 10) % 2 == 0 && intValue % (intValue % 10)==0)

{

dataGridView1.Rows.RemoveAt(i);

i--;

}

}

}

}

catch (Exception error)

{

Fm1.textBox1.Text = error.Message + '\n';

Fm1.textBox1.Text = Convert.ToString(error.InnerException) + '\n';

Fm1.textBox1.Text = error.Source + '\n';

Fm1.textBox1.Text += error.StackTrace;

}

}

## Пример работы программы

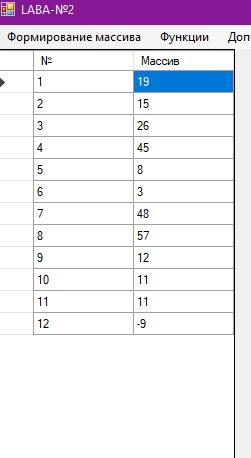


Рисунок − Массив до вызова функции удаления

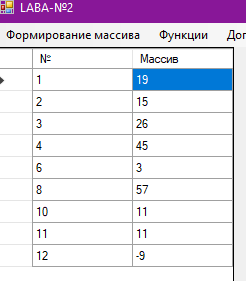


Рисунок − Массив после вызыва функции удаления

## Пункт 2

Функция вставки числа k перед и после всех элементов, заканчивающихся на данную цифру (k вводить с клавиатуры)

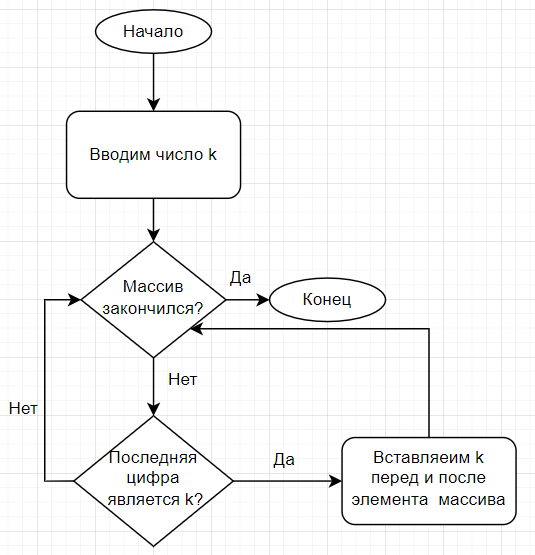
Алгоритм для выполнения данной задачи представлен на рисунке 6.

Рисунок – алгоритм вставки числа k

## Листинг программы

try

{

Form3 newForm = new Form3(this);

newForm.ShowDialog();

for (int i = 0; i < (dataGridView1.RowCount - 1); i++)

{

int intValue;

bool succes = Int32.TryParse((string)dataGridView1.Rows[i].Cells[1].Value, out intValue);

if (!succes || intValue > 60 || intValue < -10) throw new Exception("Введенные данные не соответствуют требованиям");

if (intValue % 10 == number)

{

dataGridView1.Rows.Insert(i);

dataGridView1.Rows[i].Cells[1].Value = number.ToString();

i++;

dataGridView1.Rows.Insert(i + 1);

dataGridView1.Rows[i + 1].Cells[1].Value = number.ToString();

i++;

}

}

for (int i = 0; i < dataGridView1.RowCount; i++)

{

dataGridView1.Rows[i].Cells[0].Value = (i + 1).ToString();

}

}

catch (Exception error)

{

Fm1.textBox1.Text = error.Message + '\n';

Fm1.textBox1.Text = Convert.ToString(error.InnerException) + '\n';

Fm1.textBox1.Text = error.Source + '\n';

Fm1.textBox1.Text += error.StackTrace;

}

## Пример работы программы

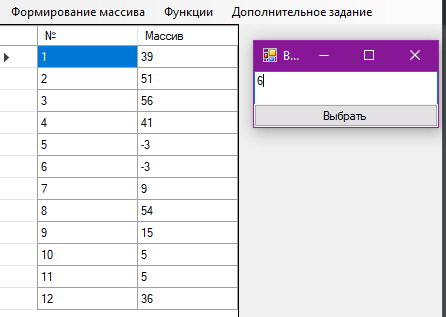


Рисунок − Массив до вызова функции вставки числа k

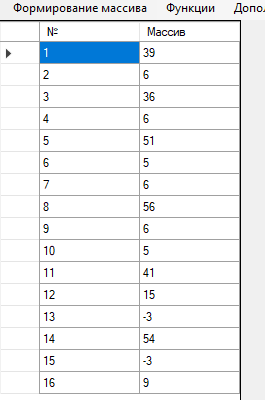


Рисунок − Массив после вызова функции вставки числа k

## Пункт 3

Алгоритм перестановки элементов следующим образом: a[1], a[12], a[2], a[11], a[3], a[10],...,a[5],a[8],a[6],a[7], предоставлен на рисунке 8.

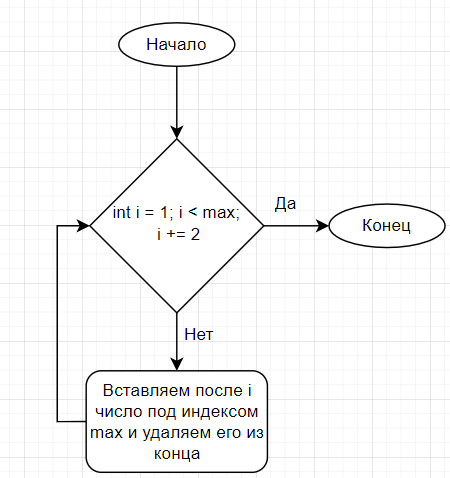


Рисунок – Алгоритм перестановки массива

## Листинг программы

try

{

int max = dataGridView1.RowCount - 1;

for (int i = 1; i < max; i += 2)

{

dataGridView1.Rows.Insert(i);

dataGridView1.Rows[i].Cells[1].Value = dataGridView1.Rows[max + 1].Cells[1].Value;

dataGridView1.Rows[i].Cells[0].Value = dataGridView1.Rows[max + 1].Cells[0].Value;

dataGridView1.Rows.RemoveAt(max + 1);

}

}

catch (Exception error)

{

Fm1.textBox1.Text = error.Message + '\n';

Fm1.textBox1.Text = Convert.ToString(error.InnerException) + '\n';

Fm1.textBox1.Text = error.Source + '\n';

Fm1.textBox1.Text += error.StackTrace;

}

## Пример работы программы

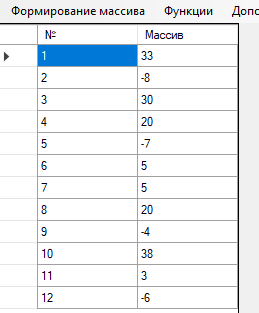


Рисунок − Массив до вызова функции перестановки

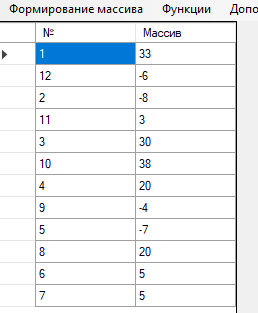


Рисунок − Массив после вызова функции перестановки

# Дополнительное задание

Зашифровать с помощью шифра Цезаря свои имя и фамилию. (шаг = 6)

Алгоритм данного шифра представлен блок-схемой на рис 10.

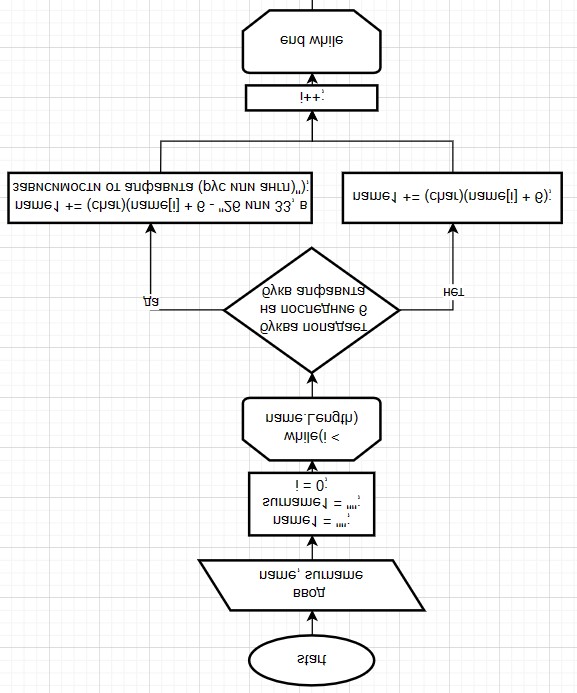
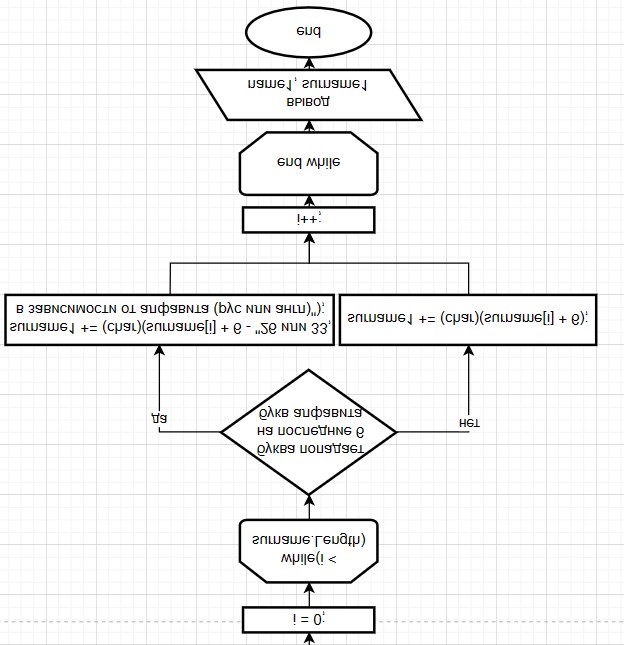


Рисунок − Блок-схема шифра Цезаря с шагом 6

## Часть программы

if (textBox1.Text == "" || textBox2.Text == "")

{

label3.Text = "Заполните поля";

return;

}

string name = textBox1.Text;

foreach (char c in name)

{

if (!char.IsLetter(c))

{

label3.Text = "Введите имя с использованием только букв";

return;

}

}

string surname = textBox2.Text;

foreach (char c in surname)

{

if (!char.IsLetter(c))

{

label3.Text = "Введите фамилию с использованием только букв";

return;

}

}

string endString = "";

label3.Text = "";

for (int i = 0; i < name.Length; i++)

{

if (name[i] >= 'а' && name[i] <= 'я')

{

if (name[i] >= 'ъ' && name[i] <= 'я')

{

endString += (char)(name[i] + 6 - 33);

}

else

{

endString += (char)(name[i] + 6);

}

}

else if (name[i] >= 'А' && name[i] <= 'Я')

{

if (name[i] >= 'Ъ' && name[i] <= 'Я')

{

endString += (char)(name[i] + 6 - 33);

}

else

{

endString += (char)(name[i] + 6);

}

}

else if (name[i] >= 'a' && name[i] <= 'z')

{

if (name[i] >= 'u' && name[i] <= 'z')

{

endString += (char)(name[i] + 6 - 26);

}

else

{

endString += (char)(name[i] + 6);

}

}

else if (name[i] >= 'A' && name[i] <= 'Z')

{

if (name[i] >= 'U' && name[i] <= 'Z')

{

endString += (char)(name[i] + 6 - 26);

}

else

{

endString += (char)(name[i] + 6);

}

}

}

textBox3.Text = endString;

string sur = " ";

for (int i = 0; i < surname.Length; i++)

{

if (surname[i] >= 'а' && surname[i] <= 'я')

{

if (surname[i] >= 'ъ' && surname[i] <= 'я')

{

sur += (char)(surname[i] + 6 - 33);

}

else

{

sur += (char)(surname[i] + 6);

}

}

else if (surname[i] >= 'А' && surname[i] <= 'Я')

{

if (surname[i] >= 'Ъ' && surname[i] <= 'Я')

{

sur += (char)(surname[i] + 6 - 33);

}

else

{

sur += (char)(surname[i] + 6);

}

}

else if (surname[i] >= 'a' && surname[i] <= 'z')

{

if (surname[i] >= 'u' && surname[i] <= 'z')

{

sur += (char)(surname[i] + 6 - 26);

}

else

{

sur += (char)(surname[i] + 6);

}

}

else if (surname[i] >= 'A' && surname[i] <= 'Z')

{

if (surname[i] >= 'U' && surname[i] <= 'Z')

{

sur += (char)(surname[i] + 6 - 26);

}

else

{

sur += (char)(surname[i] + 6);

}

}

}

textBox3.Text += sur;

## Пример работы программы

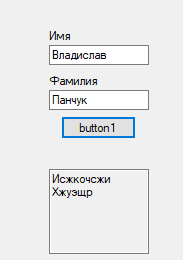


Рисунок − Пример работы шифра Цезаря.

# Вывод

Освоили навыки работы обработки исключений при работе с массивами, формируемыми в различных режимах.