МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

**КАФЕДРА ИИТ**

ОТЧЁТ

по лабораторной работе №9

**«Пользовательские функции»**

Выполнил:

студент 1 курса

группы ПО-9

Харитонович Захар Сергеевич

Проверила:

Хацкевич М. В.

Брест 2022

**Цель работы:** Изучить основные принципы написания пользовательских

функций, ознакомиться с возможностями передачи данных в функции и получения

результата по итогам работы функции. Реализовать собственные функции для

обработки данных составных и простых типов.

***Задание 1***

*Вариант 10*

**Текст задания**

Описать процедуру DoubleX(A, N, X), дублирующую в целочисленном массиве A размера N элементы, равные целому числу X. Массив A и число N являются входными и выходными параметрами. С помощью этой процедуры продублировать числа XA, XB, XC в массивах A, B, C размера NA, NB, NC соответственно и вывести размер и содержимое полученных массивов.

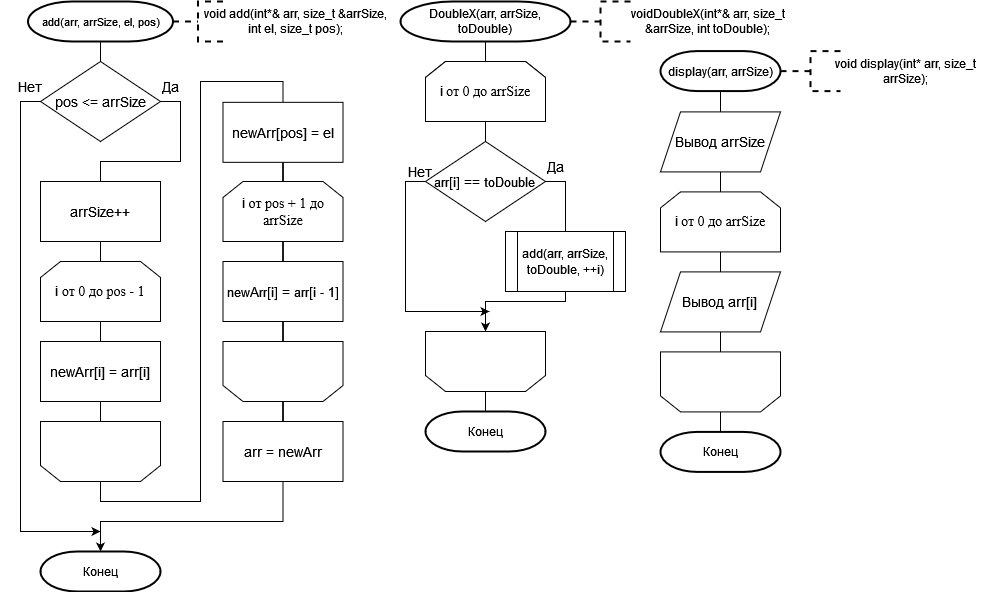
**Описание способа решения задачи**

Реализуем процедуру DoubleX(), которая в качестве аргументов принимает ссылку на указатель массива *\*&arr*, ссылку на размер массива *&arrSize* и целое число *toDouble*, которое в данном массиве необходимо продублировать. Перебирая массив, проверяем равность текущего элемента числу *toDouble* и если они равны, на следующую за текущим элементом позицию мы вставляем со сдвигом такой же элемент.

Для удобства использования и повышения читаемости кода вставку со сдвигом нового элемента в массив реализуем в виде отдельной процедуры add(), которая в качестве аргументов принимает ссылку на указатель массива *\*&arr*, ссылку на размер массива *&arrSize*, значение добавляемого элемента *el* и позицию вставки *pos*. Создав новый массив, мы копируем в него элементы из массива *arr* до позиции *pos,* вставляем новый элемент *el* и продолжаем копирование исходного массива. Изменяем адрес входного массива на адрес нового.

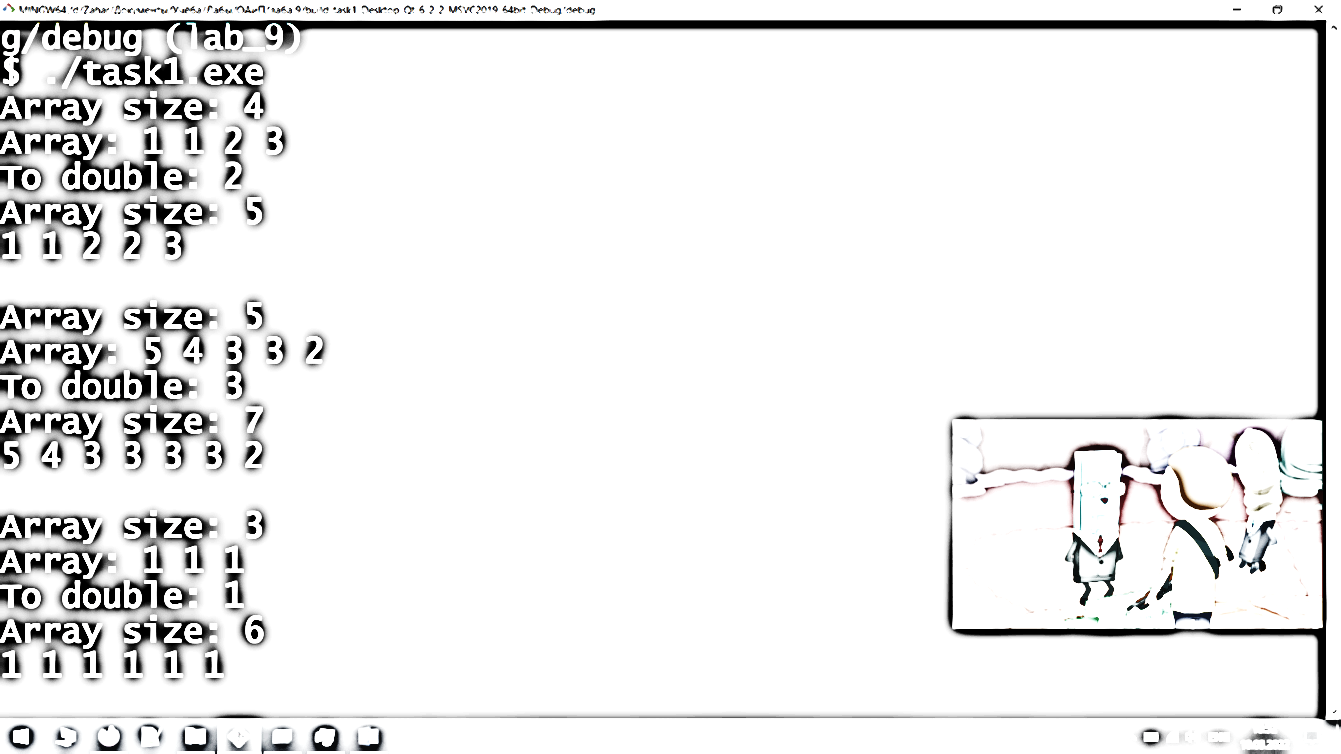
Для сокращения кода также реализуем процедуру *display()* – процедуру вывода массива.

**Блок-схема**



**Код программы:**

#include <iostream> **Результат работы:**



using namespace std;

void add(int\*& arr, size\_t &arrSize, int el, size\_t pos);

void DoubleX(int\*& arr, size\_t &arrSize, int toDouble);

void display(int\* arr, size\_t arrSize);

int main() {

for(short i = 0; i < 3; i++) {

int \*A;

size\_t n;

int x;

cout << "Array size: ";

cin >> n;

A = new int[n];

cout << "Array: ";

for (size\_t i = 0; i < n; i++) cin >> A[i];

cout << "To double: ";

cin >> x;

DoubleX(A, n, x);

display(A, n);



}

return 0;

}

void add(int\*& arr, size\_t &arrSize, int el, size\_t pos) {

if (pos <= arrSize) {

int\* newArr = new int[++arrSize];

for (size\_t i = 0; i < pos; i++)

newArr[i] = arr[i];

newArr[pos] = el;

for (size\_t i = pos + 1; i < arrSize; i++) {

newArr[i] = arr[i-1];

}

delete[] arr;

arr = newArr;

}

}

void DoubleX(int\*& arr, size\_t &arrSize, int toDouble) {

for (size\_t i = 0; i < arrSize; i++) {

if (arr[i] == toDouble) {

add(arr, arrSize, toDouble, ++i);

}

}

}

void display(int\* arr, size\_t arrSize) {

cout << "Array size: " << arrSize << endl;

for (size\_t i = 0; i < arrSize; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

***Задание 2***

*Вариант 5*

**Текст задания**

Описать процедуру TrimLeftC(S, C), удаляющую в строке S начальные символы, совпадающие с символом C. Строка S является входным и выходным параметром. Дан символ C и пять строк. Используя процедуру TrimLeftC, преобразовать данные строки.

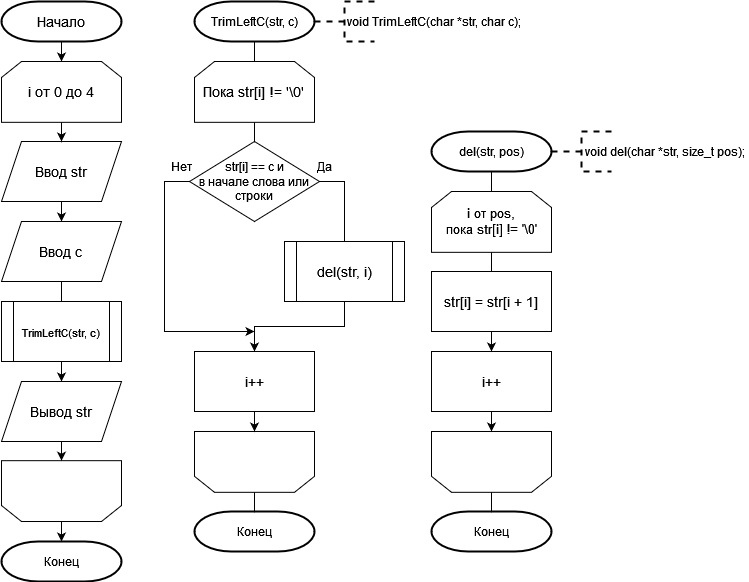
**Описание способа решения задачи**

Реализуем процедуру TrimLeftC(), которая в качестве параметров получает указатель на строку *str* и символ *c*, который в данной строке должен быть удалён.

Перебираем массив символов и сравниваем каждый элемент с переменной *с*. Если значение элемента совпадает с подлежащим к удалению символом и при этом оно находится в начале слова, то он удаляется.

Для удаления символа из строки реализуем процедуру *del()*, принимающую в качестве параметров указатель на строку и индекс символа, который необходимо удалить.

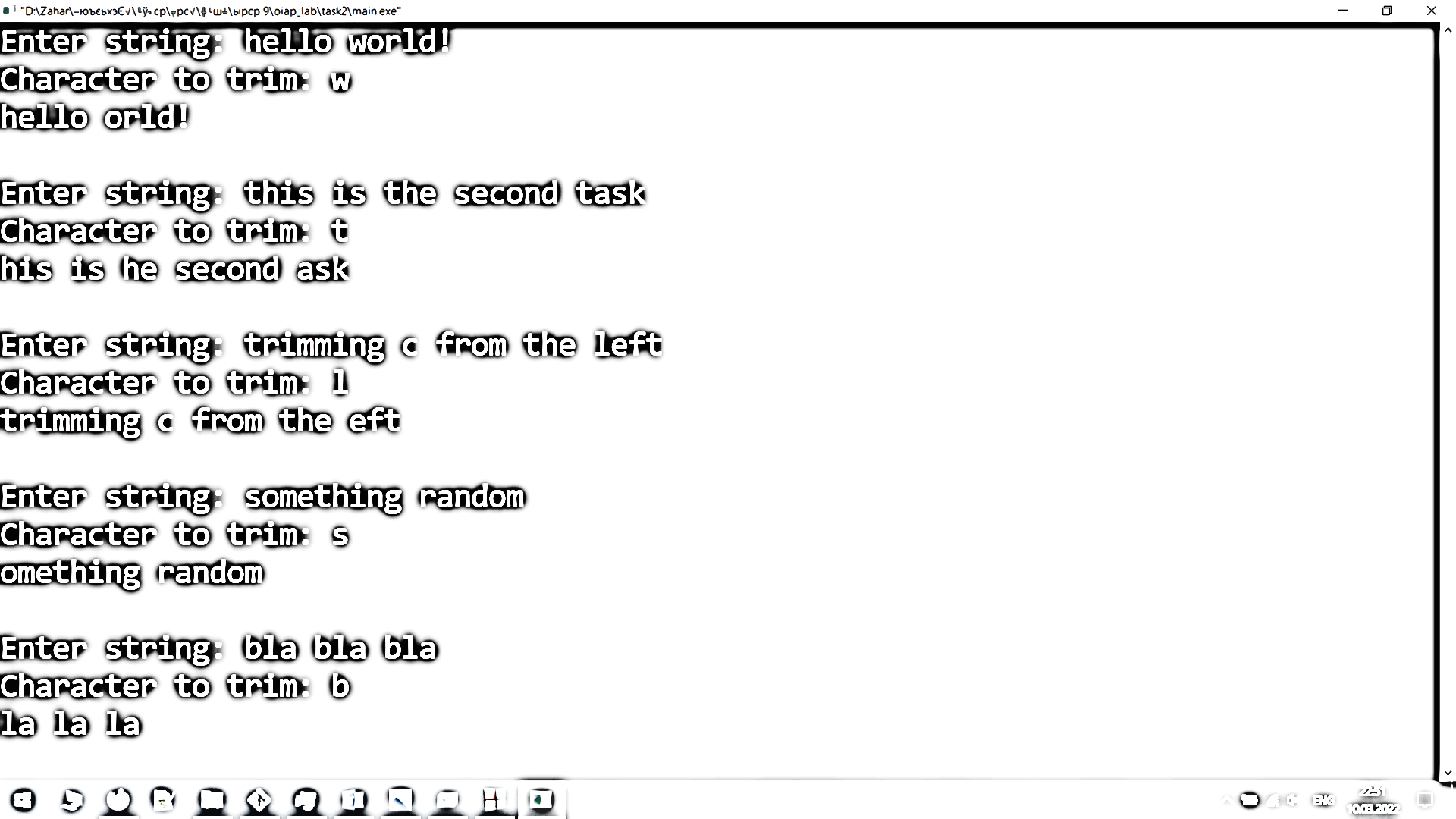
**Блок-схема**



**Текст программы**

#include <stdio.h>

#include <stdlib.h> **Результат работы**



void TrimLeftC(char \*str, char toTrim);

void del(char \*str, size\_t pos);

int main() {

for (short i = 0; i < 5; i++) {

char str[256], c;

printf("Enter string: ");

gets(str);

printf("Character to trim: ");

scanf("%c", &c);

TrimLeftC(str, c);

puts(str);

printf("\n");

fflush(stdin);

}

return 0;

}

void TrimLeftC(char \*str, char toTrim) {

size\_t curSize = 0;

for (size\_t i = 0; str[i] != '\0'; i++) {

if (str[i] == toTrim && (str[i-1] == ' ' || i == 0)) {

del(str, i);

}

}

}

void del(char \*str, size\_t pos) {

for (size\_t i = pos; str[i] != '\0'; i++) {

str[i] = str[i+1];

}

}

**Вывод:** По итогам выполнения лабораторной работы, были изучены основные принципы написания пользовательских функций, возможности передачи данных в функции и получения результата работы функций. Были реализованы функции для обработки данных составных и простых типов.