

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

ассистент

должность, уч. степень, звание

подпись, дата

Синёв Н. И.

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №4

ОБРАБОТКА СИМВОЛЬНЫХ СТРОК

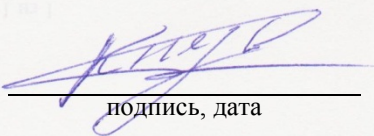
КУРС 1

по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

1941



подпись, дата

Князюк Р.А.

инициалы, фамилия

Санкт-Петербург

2020

1. Постановка задачи.

В символьной строке оставить только те слова, в которых встречаются подряд идущие одинаковые буквы.

2. Формализация.

- **Список переменных в данной программе:**

- Символьные константы:

TRUE – 1;

FALSE – 0;

SIZE – 1024 (кол-во элементов в массиве);

- Массив типа **char**:

string[*SIZE*] – массив для хранения строки (кол-во эл. *SIZE*);

- Переменные типа **char**:

temp – переменная хранящая предыдущий символ для будущего сравнения;

- Переменные типа **int**:

flag – переменная указывающая ,что сейчас вводится слово;

begin_word – переменная для хранения позиции начала слова;

rep_char – счетчик повторяющихся подряд символов в слове;

- Все данные вводятся с клавиатуры.
- Ввод массива производится циклически, пока не будет считан символ конца строки – **EOF** (End Of File). Для системы Windows – Ctrl+Z, для системы Linux – Ctrl+D.
- Будем считать, что слова в потоке символов разделяются пробелами, точками, запятыми, концом строки, переносом строки, восклицательными знаками, вопросительными знаками и любыми их комбинациями. Следовательно, разделение слова между строками запрещён.
- Числа и их комбинации **не будем** считать словом (или частью слова), а также не считается словом любая комбинация разделительных знаков (то есть “102311” не будет являться словом, а “одиннадцать” будет словом).
- Все слова, в которых не встречаются подряд идущие одинаковые буквы будем, замещать пробелами.
- В конце программа должна вывести отредактированную строку, игнорируя лишние пробелы(если подряд идет больше двух пробелов напечатан будет только один).

- Переменная *flag* хранит состояние определяющее ввод слова если *true* и *false* если сейчас вводятся разделяющие символы.
- Алгоритм вывода и ввода будет отдельной подпрограммой.
- Операционная система: Windows 7 Максимальная
- Написание кода: Visual Studio Code
- Компилятор: MinGW GCC-8.2.0-5
- Запуск программы через *.exe

Таблица 1 – Тестовые примеры

Пример	№1	№2
Данные	‘Hello world! 3333’	‘Hello,world! How are you?How your feelings? 213’
Результат	Hello ! 3333	Hello, ! ? feelings? 213

3. Алгоритмизация. Схема алгоритма.
а. Подпрограмма вывода массива.

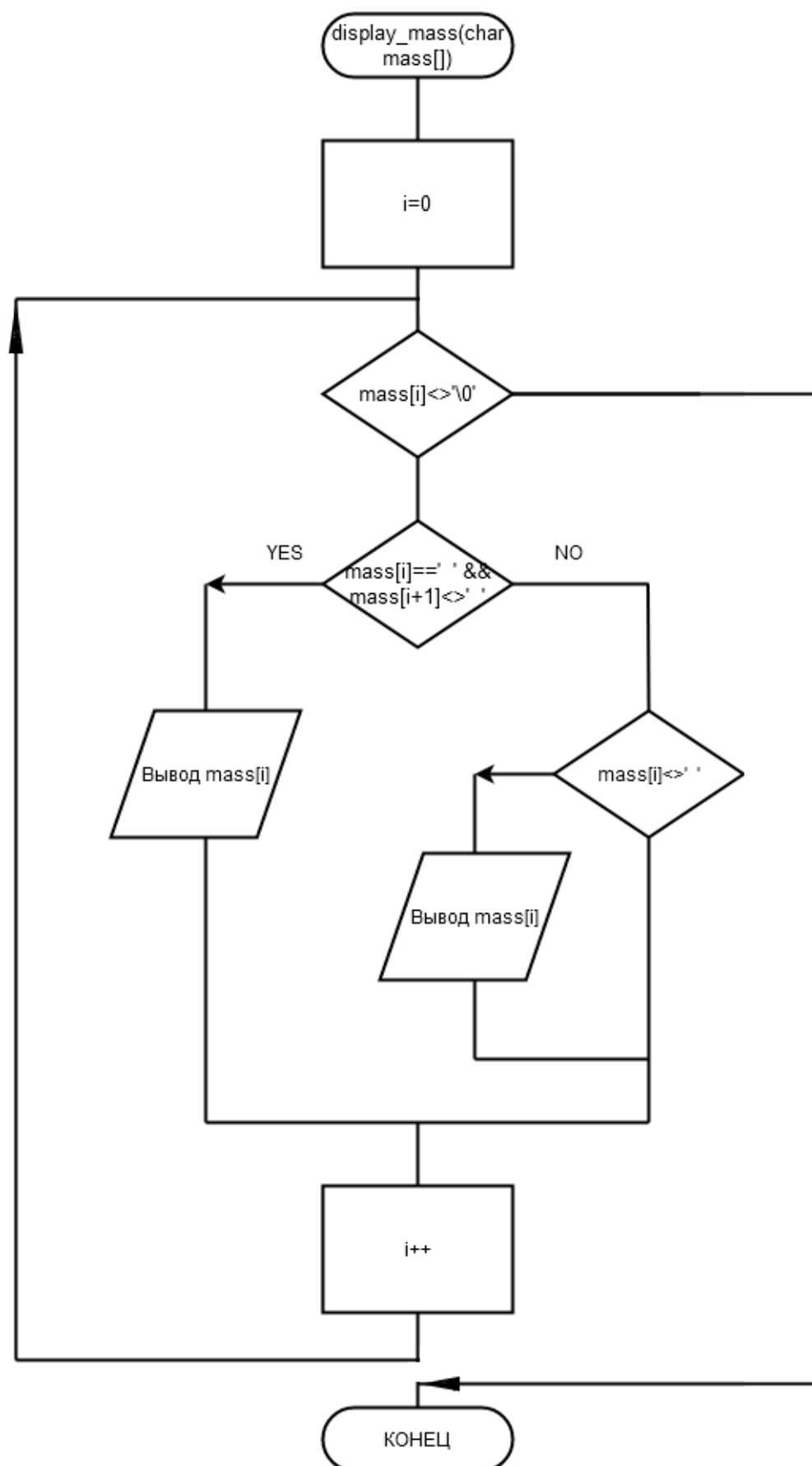


Рисунок 1 – подпрограммы вывода
`display_mass(char mass[])`

в. Подпрограмма ввода массива.

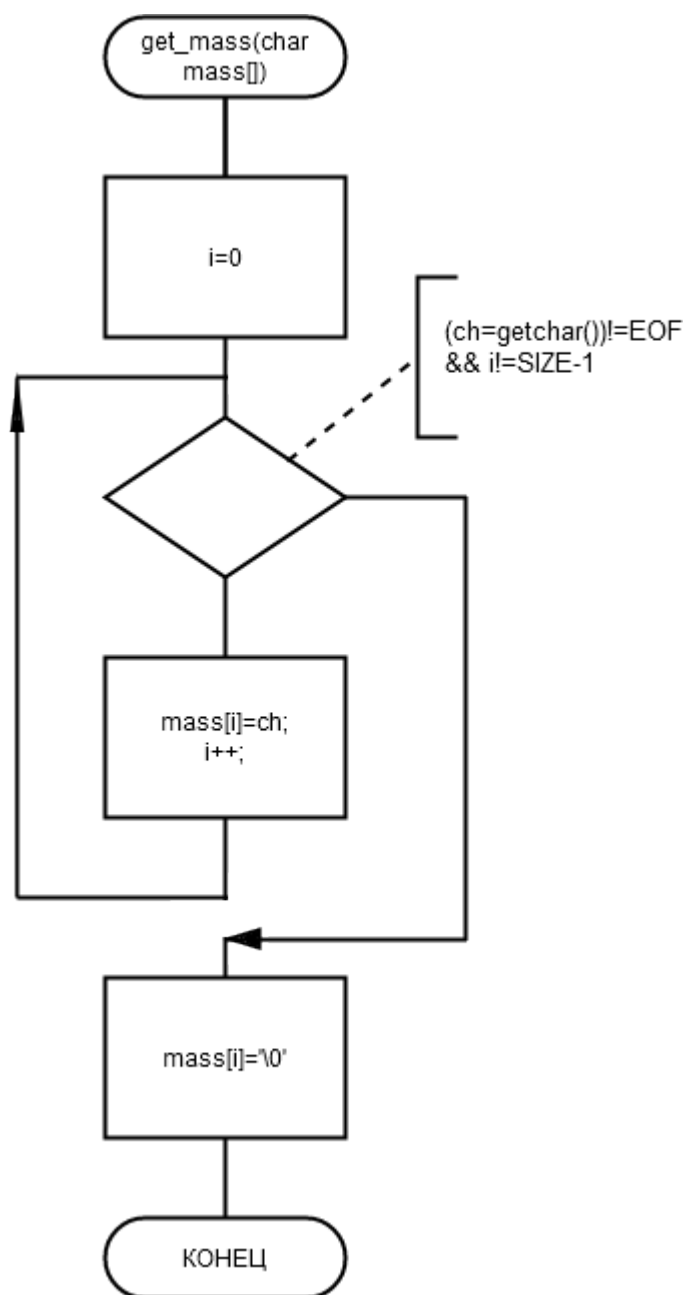


Рисунок 2 – подпрограммы ввода
`get_mass(char mass[])`

с. Основная программа.

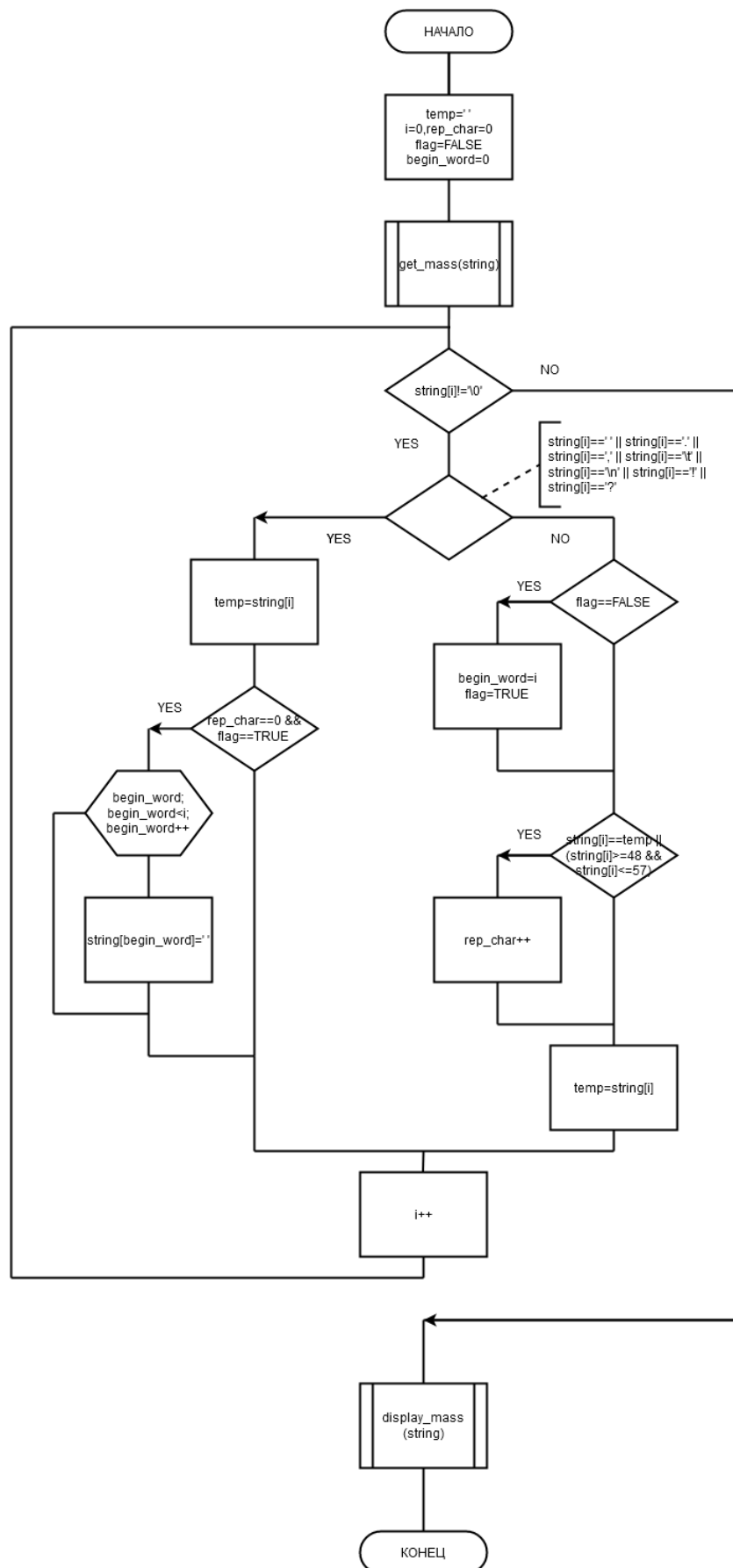


Рисунок 3 – Блок-схема основной программы

4. Код программы на Си.

```
#include <stdio.h>
#include <stdlib.h> //для работы system("pause");

//размер массива string
#define SIZE 1024
//константы для флага (flag)
#define FALSE 0
#define TRUE 1

void get_mass(char mass[]){
    char ch;
    int i=0;

    while((ch=getchar())!=EOF && i!=SIZE-1)
        mass[i++]=ch;
    mass[i]='\0';
}

void display_mass(char mass[]){
    int i=0;

    while(mass[i]!='\0'){
        //выводим без лишних пробелов
        if(mass[i]==' ' && mass[i+1]!=' ')
            putchar(mass[i]);
        else if(mass[i]!=' ')
            putchar(mass[i]);
        i++;
    }
}

int main(void){
    char string[SIZE];
    char temp=' ';
    int flag=FALSE, //флаг для проверки слова
        begin_word=0, //счетчик слов
        rep_char=0, //счетчик повторов подряд идущих символов
        i=0; //счетчик для циклов

    get_mass(string);

    while(string[i]!='\0'){
        if(string[i]==' ' || string[i]=='.' || string[i]==',' || string[i]=='\t' |
        | string[i]=='\n' || string[i]=='!' || string[i]=='?'){
            temp=string[i];
            //вырезаем слово, в котором нет подряд идущих одинаковых символов
            if(rep_char==0 && flag==TRUE)
                for(begin_word;begin_word<i;begin_word++)
                    string[begin_word]=' ';
        }
    }
}
```

```
rep_char=0;
flag=FALSE;
}else{
if(flag==FALSE){
//запоминаем начало слова
begin_word=i;
flag=TRUE;
}
//проверяем на подряд идущие одинаковые символы
if(string[i]==temp || (string[i]>=48 &&string[i]<=57))
rep_char++;
//сохраняем текущий символ для последующей проверки
temp=string[i];
}
i++;
}

display_mass(string);

system("pause");
return 0;
}
```


5. Проверка работы программы на тестовых примерах.

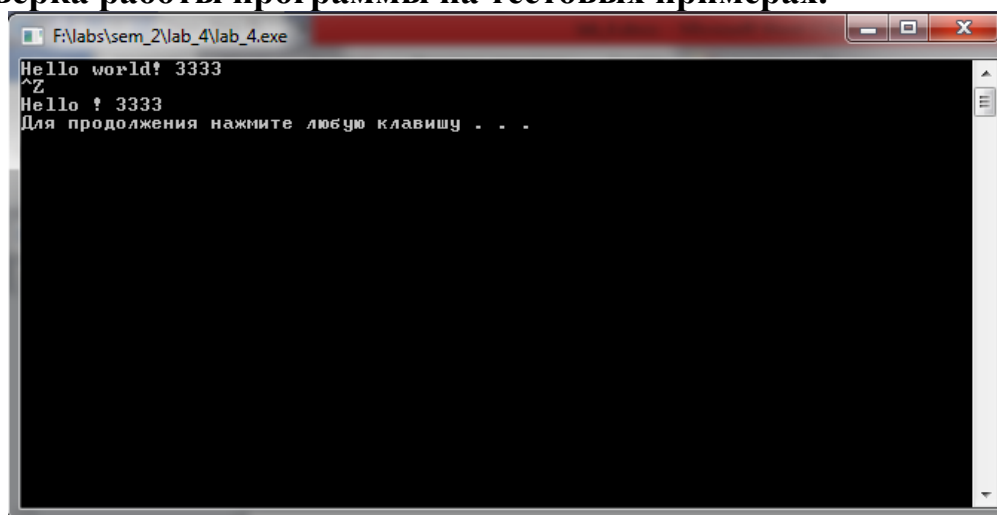


Рисунок 4 – Тестовый пример 1

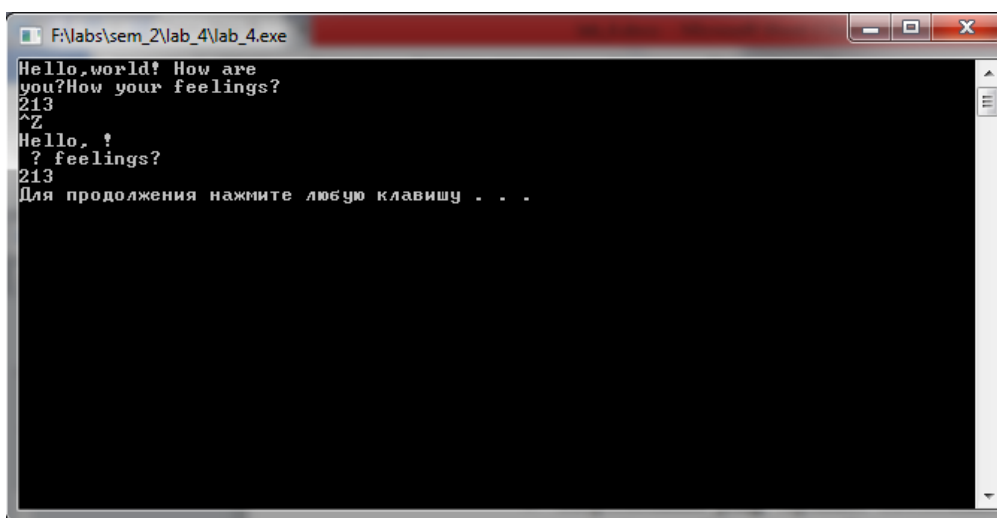


Рисунок 5 – Тестовый пример 2

6. Вывод.

Тестовые примеры (5 пункт отчета) доказывают правильность работы алгоритма и программы. Таким образом программа способна редактировать строку , удаляя из нее слова, в которых не встречаются подряд идущие символы.