|  |
| --- |
| 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 |  |  |  | Синёв Н. И. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| ОБРАБОТКА ПОТОКА СИМВОЛОВЛАБОРАТОРНАЯ РАБОТА №3 |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 1941 |  |  |  | Князюк Р.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2020

1. **Постановка задачи.**

В потоке символов сосчитать число слов, в которых встречаются подряд идущие одинаковые буквы.

1. **Формализация задачи.**

* **Список переменных в данной программе:**
  + Тип переменной **char** (для хранения символов):

***ch* –** переменная для хранения текущего введенного символа;

***temp*** – переменная хранящая предыдущий символ для будущего сравнения;

* + Символьные константы :

TRUE – 1;

FALSE – 0;

* + Тип переменной **int** (для хранения состояния или для хранения числа):

***flag*** – переменная указывающая ,что сейчас вводится

слово;

***cnt\_words*** – счетчик слов;

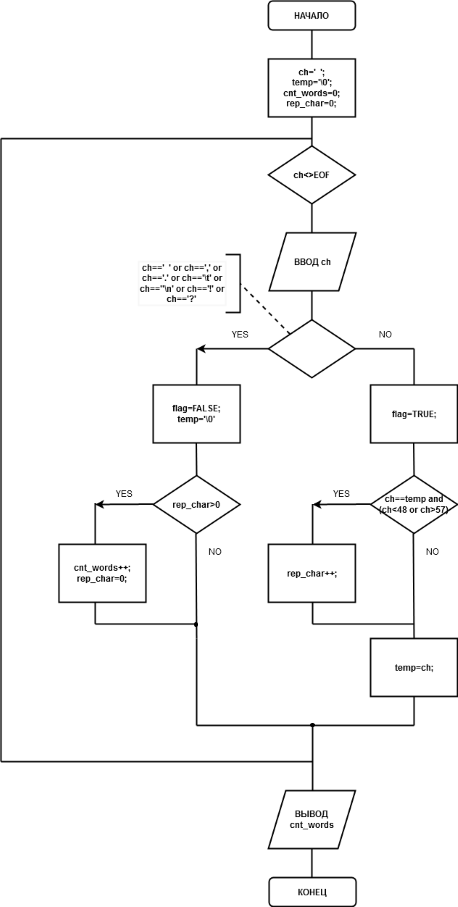
***rep\_char*** – счетчик повторяющихся подряд символов в слове;

* Все данные вводятся с клавиатуры.
* Ввод 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, перенос слов между строками запрещён.
* Числа и их комбинации **не будем** считать словом (или частью слова) (то есть “102311” не будет являться слово, а “одиннадцать” будет словом).
* В конце программа должна вывести количество слов с повторяющимися подряд символами.
* Переменная ***flag*** хранит состояние определяющее ввод слова если ***true* и *false*** если сейчас вводятся разделяющие символы.

|  |  |  |  |
| --- | --- | --- | --- |
| Пример | №1 | №2 | №3 |
| Данные | ‘Hello’ | No, it’s bad !! | Hello, worrld!! Number 1000?? !! ?? .. |
| Результат | 1 | 0 | 2 |

**Таблица 1 – Тестовые примеры**

1. **Алгоритмизация. Схема алгоритма.**

****

Лист 1

**Рисунок 1 – Схема алгоритма**

1. **Код программы на Си.**

#include <stdio.h>

//константы для флага (flag)

#define FALSE 0

#define TRUE 1

int main(void){

    char ch=' ',temp='\0';

    int flag=FALSE,   //флаг для проверки слова

        cnt\_words=0,  //счетчик слов

        rep\_char=0;   //счетчик повторов подряд идущих символов

    //WIN+Z - EOF комбинация для Windows

    //WIN+D - EOF для Linux

    while((ch=getchar())!=EOF){

        if(ch==' ' || ch=='.' || ch==',' || ch=='\t' || ch=='\n' || ch=='!' || ch=='?'){

            flag=FALSE;

            temp='\0';

            if(rep\_char>0){

                cnt\_words++;

                rep\_char=0;

            }

        }else{

            flag=TRUE;

            if(ch==temp && (ch<48 || ch >57))

                rep\_char++;

            temp=ch;

        }

    }

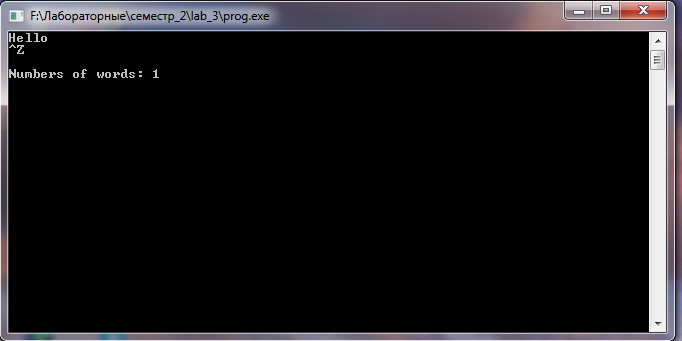
    printf("\nNumbers of words: %d\n",cnt\_words);

    getchar();

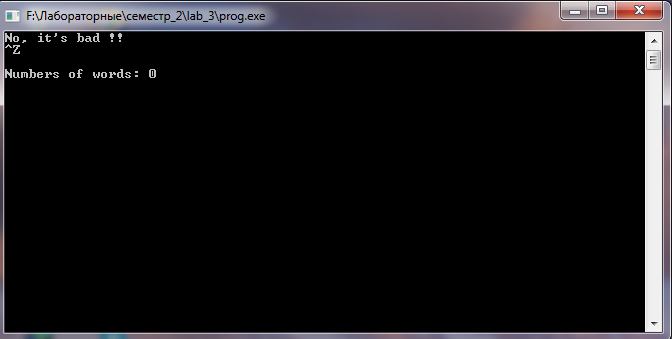
    return 0;

}

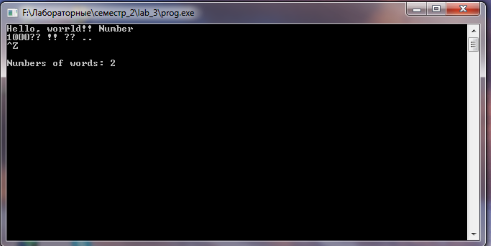
1. **Проверка работы программы на тестовых примерах.**



**Рисунок 2 – Тестовый пример 1**

****

**Рисунок 3 – Тестовый пример 2**



**Рисунок 4 – Тестовый пример 3**

1. **Вывод**.

Алгоритм и программа работает корректно, что доказывают тестовые примеры в 5 пункте отчета. Таким образом программа способна корректно решать задачу подсчета количества слов, в которых встречаются подряд идущие одинаковые буквы. .