МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Организация ЭВМ и систем»
Тема: Представление и обработка символьной информации с использованием строковых команд.
Вариант 15

Студент гр. 0383 _____ Смирнов И.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Изучить представление и обработку символьной информации с использованием строковых команд на языке Ассемблера. Разработать программу, которая обрабатывает строку.

Задание.

Разработать программу обработки символьной информации, реализующую функции:

- инициализация (вывод титульной таблички с указанием вида преобразования и автора программы) - на ЯВУ;

- ввода строки символов, длиной не более Nmax (<=80), с клавиатуры в заданную область памяти - на ЯВУ; если длина строки превышает Nmax, остальные символы следует игнорировать;

- выполнение заданного в таблице 5 преобразования исходной строки с записью результата в выходную строку - на Ассемблере;

- вывода результирующей строки символов на экран и ее запись в файл - на ЯВУ.

Ассемблерную часть программы включить в программу на ЯВУ по принципу встраивания (in-line).

Вариант №15: Исключение русских букв и цифр, введенных во входной строке, при формировании выходной строки.

Ход работы:

Объявлены 2 массива символов – in и out – для хранения входной и выходной строк.

В начале работы программы выводятся строки с фамилией, именем и номером группы, номером варианта, содержанием задания, сообщением с просьбой ввода строки.

Изменением кодовой страницы через systeqm(chcp…) и локали через setlocale подготавливаем программу к работе с кириллицей.

Далее объявляется ассемблерный блок через_asm, в котором происходит посимвольное считывание строки через lodsb в цикле, все условия цикла, кроме условия считывания символа конца строки, ведут к возврату к его началу. Условие считывание символа конца строки переносит нас к метке конца цикла. С помощью сравнений (cmp) и переходов (je, jne, jl, jg…) символ сравнивается с промежутками А-я и 0-9, отдельно проверяются Ё и ё, в случае, если символ проходит все эти проверки не попадая в данные промежутки он отправляется на вывод.

В конце работы программы происходит вывод строки out в текстовый документ и консоль.

Исходный код программы см. в приложении А.

Тестирование:

Для проверки работоспособности программы были проведены тесты, см. Таблицу 1.

Таблица 1 — Результаты тестирования.

№ теста	Входные данные	Выходные данные	Оценка результата
1	Собака села на banan ну и ok 322	banan ok	Верно
2	Solve et coagula	Solve et coagula	Верно
3	Я поджигал города пока ты обитал на комментах капитан по кабине летал		Верно

Выводы.

В результате лабораторной работы была изучена обработка символьной информации с использованием ассемблерного блока в коде на ЯВУ

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include "stdafx.h"
#include <iostream>
#include <cstdlib>
#include <fstream>

using namespace std;
char in[81];
char out[81];
int main(){
    system("chcp 1251 > nul");
    setlocale(LC_CTYPE, "rus");
    cout << "Smirnov Ivan gr.0383" << endl;
       cout << "Variant 15" << endl;
       cout << "Deleting digits and russian symbols" << endl;
       cout << "Enter the string to process (81 symbols or less):" << endl;
    ofstream file;
    file.open("C:\\Users\\hippo\\Desktop\\output.txt");
    cin.getline(in, 81);
    __asm {
        mov esi, offset in
        mov edi, offset out

        loop_start:
            lodsb
            cmp al, '\0'
            je loop_finish
            cmp al, 'Ё'
                     je loop_start
            cmp al, 'ё'
                     je loop_start
            cmp al, 'A'
                     jl check_if_digit
            cmp al, 'я'
                     jg check_if_digit
            jmp loop_start

        check_if_digit:
            cmp al, '0'
                     jl write_passed
            cmp al, '9'
                     jg write_passed
            jmp loop_start

        write_passed:
            stosb
            jmp loop_start

        loop_finish:
    };
    cout << out;
    file << out;
    file.close();
       system("pause");
    return 0;

}
```