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.

Студент гр. 0382 _____ Сергеев Д.А,

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Изучение представление и обработку символьной информации, научиться работать со строковыми командами языка Ассемблер. Написать программу для обработки текста.

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 16:

Преобразование введенных во входной строке русских букв в латинские в соответствие с правилами транслитерации, остальные символы входной строки передаются в выходную строку непосредственно.

Выполнение работы.

Первым делом прописываются команды для работы с кириллицей, далее вводится исходная строка input, максимальный размер которой 80 символов, если строка длиннее 80 символов, хвост обрезается. Далее объявляется ассемблерный блок __asm, в нём мы с начала задаём сегмент ES=DS, а в регистры esi и edi,

помещаем начала исходной и выходной строк. Начинается цикл MyLoop, в нём инструкцией lodsb в регистр al помещается символ исходной строки, далее с помощью инструкции cmp и условных переходов определяется что за символ перед нами, если это буква кириллицы, то она меняется на представление латинскими буквами в соответствии с правилами транслитерации и результат записывается в регистр al или ax, далее она записывается в выходную строку с помощью инструкции stosb и stosw мы переходим в начало цикла, если это символ конца строки, то мы записываем его в выходную строку и заканчиваем ассемблерный блок, если это не конец строки и не символ кириллицы, то он записывается в выходную строку без изменений.

Исходный программный код смотрите в приложении А.

Тестирование.

Результаты представлены в таблице 1.

Таблица 1 – Результаты тестирования

№	Входные данные	Выходные данные	Комментарий
1	ПРИВЕТ kak dela12367!!!	PRIVET kak dela12367!!!	Правильно.
2	А Б В Г Д Е Ё Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Э Ю Я Ъ Ь	A B V G D E E J Z I I K L M N O P R S T U F H C CH SH SC E IU IA _ _	Правильно.

Выводы.

В ходе работы было изучено представление и обработка символьной информации, получены навыки работать со строковыми командами языка Ассемблер. Была написана программа для обработки текста в соответствии с заданием

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include <iostream>
#include <fstream>
#include <wchar.h>
#include <stdio.h>

char input[81];
char output[161];

int main() {
    system("chcp 1251 > nul");
    setlocale(LC_CTYPE, "rus");
    std::cout << "VAR 16: change russian letters into latin\nby Dmitry Sergeev 0382\n";
    fgets(input, 81, stdin);
    input[strlen(input)] = '\0';

    __asm
    {
        push ds
        pop es
        mov esi, offset input
        mov edi, offset output

        MyLoop :
        lodsb

            cmp al, '\0'
            je final

            cmp al, 'Ё'
            je replace_e

            cmp al, 'А'
            je replace_a

            cmp al, 'Б'
            je replace_b

            cmp al, 'В'
            je replace_v

            cmp al, 'Г'
            je replace_g

            cmp al, 'Д'
            je replace_d

            cmp al, 'Е'
            je replace_e

            cmp al, 'Ж'
            je replace_zh

            cmp al, 'З'
            je replace_z

            cmp al, 'И'
            je replace_i
```

```
cmp al, 'Й'
je replace_i

cmp al, 'К'
je replace_k

cmp al, 'Л'
je replace_l

cmp al, 'М'
je replace_m

cmp al, 'Н'
je replace_n

cmp al, 'О'
je replace_o

cmp al, 'П'
je replace_p

cmp al, 'Р'
je replace_r

cmp al, 'С'
je replace_s

cmp al, 'Т'
je replace_t

cmp al, 'У'
je replace_u

cmp al, 'Ф'
je replace_f

cmp al, 'Х'
je replace_h

cmp al, 'Ц'
je replace_c

cmp al, 'Ч'
je replace_ch

cmp al, 'Ш'
je replace_sh

cmp al, 'Щ'
je replace_sc

cmp al, 'Ы'
je replace_y

cmp al, 'Э'
je replace_e

cmp al, 'Ю'
je replace_iu

cmp al, 'Я'
je replace_ia

cmp al,'Ъ'
je skip
```

```
    cmp al,'ь'
    je skip

    stosb
    jmp MyLoop

    skip:
    mov al,'_'
    stosb
    jmp MyLoop

    replace_e:
mov al, 'E'
    stosb
    jmp MyLoop

    replace_a :
mov al, 'A'
    stosb
    jmp MyLoop

    replace_b :
mov al, 'B'
    stosb
    jmp MyLoop

    replace_v :
mov al, 'V'
    stosb
    jmp MyLoop

    replace_g :
mov al, 'G'
    stosb
    jmp MyLoop

    replace_d :
mov al, 'D'
    stosb
    jmp MyLoop

    replace_zh :
mov al, 'J'
    stosb
    jmp MyLoop

    replace_z :
mov al, 'Z'
    stosb
    jmp MyLoop

    replace_i :
mov al, 'I'
    stosb
    jmp MyLoop

    replace_k :
mov al, 'K'
    stosb
    jmp MyLoop

    replace_l :
mov al, 'L'
    stosb
```

```
    jmp MyLoop

    replace_m :
mov al, 'M'
    stosb
    jmp MyLoop

    replace_n :
mov al, 'N'
    stosb
    jmp MyLoop

    replace_o :
mov al, 'O'
    stosb
    jmp MyLoop

    replace_p :
mov al, 'P'
    stosb
    jmp MyLoop

    replace_r :
mov al, 'R'
    stosb
    jmp MyLoop

    replace_s :
mov al, 'S'
    stosb
    jmp MyLoop

    replace_t :
mov al, 'T'
    stosb
    jmp MyLoop

    replace_u :
mov al, 'U'
    stosb
    jmp MyLoop

    replace_f :
mov al, 'F'
    stosb
    jmp MyLoop

    replace_h :
mov al, 'H'
    stosb
    jmp MyLoop

    replace_c :
mov al, 'C'
    stosb
    jmp MyLoop

    replace_ch :
mov ax, 'HC'
    stosw
    jmp MyLoop

    replace_sh :
mov ax, 'HS'
    stosw
```

```
            jmp MyLoop

            replace_sc :
        mov ax, 'CS'
            stosw
            jmp MyLoop

            replace_y :
        mov al, 'Y'
            stosb
            jmp MyLoop

            replace_iu :
        mov ax, 'UI'
            stosw
            jmp MyLoop

            replace_ia :
        mov ax, 'AI'
            stosw
            jmp MyLoop

            final:
            stosb
    }
    std::ofstream out;
    out.open(R"(C:\Task\log.txt)");
    out << output;
    out.close();
    std::cout << output;
    return 0;
}
```