

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

**ОТЧЕТ**  
**по лабораторной работе №4**  
**по дисциплине «ОргЭВМиС»**  
**Тема: Представление и обработка символьной информации с**  
**использованием строковых команд.**

Студент гр. 0382

Гудов Н.Р..

Преподаватель

Ефремов М.А.

Санкт-Петербург

2021

## **Цель работы.**

Изучение представления и обработке символьной информации на языке Ассемблера. Использование кода на языке Ассемблера вместе с кодом на ЯВУ.

## **Задание.**

Вариант 21.

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  $N_{\max}$  ( $\leq 80$ ), с клавиатуры в заданную область памяти - на ЯВУ; если длина строки превышает  $N_{\max}$ , 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).

Заменить введенные во входной строке латинские буквы на числа, соответствующие их номеру по алфавиту, представленному в шестнадцатиричной СС, остальные символы входной строки передать в выходную строку непосредственно.

## **Выполнение работы.**

Считывается символ. После этого проверяется его принадлежность маленьким латинским буквам, путем сравнения с первым и последним строчным символом алфавита. Если условие не выполняется, проверяется принадлежность к заглавным буквам по той же схеме. И если это заглавная буква латинского

алфавита, то буква заменяется на ту же, но строчную и обрабатывается на предыдущем шаге. В ином случае символ не обрабатывается.

Обработка строчной буквы происходит так: Устанавливается, принадлежит ли этот символ первым 15 символам алфавита или же оставшимся. После этого в каждой группе символы делятся на группу символов с номерами в виде десятичной цифры или группу с номерами в виде латинской заглавной буквы. После установления принадлежности к группе и подгруппе символ заменяется на две шестнадцатеричной цифры, первая из которых 0 или 1, а вторая от 0 до F.

### **Тестирование.**

Здесь результаты тестирования, которые помещаются на одну страницу.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	a b c d e f g h i j k l m n o p q r s t u v w x y z	01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A	ok
2.	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z	01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A	ok
3.	1 2 3 a б в А Б В а б с А В С	1 2 3 a б в А Б В 01 02 03 01 02 03	ok

### **Выводы.**

В ходе лабораторной работы была разработана программа обработки символьной информации путем встраивания Ассемблерного кода в код на ЯВУ.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <iostream>
#include <fstream>

using namespace std;

char input[81];
char output[81];
int main()
{
    cout << "Gudov Nikita 0382, task: Replace Latin letters with
hexadecimal numbers" << endl;
    ofstream file;
    file.open(R"(D:\tools\lab4text.txt)");
    cin.getline(input, 80);
    __asm {
        mov ax, ds
        mov es, ax
        mov esi, offset input
        mov edi, offset output
        loop_start :
        lodsb
        cmp al, '\0'
        je loop_final

        cmp al, 'a'
        jl otherAB
        cmp al, 'z'
        jg otherAB

        cmp al, 'o'
        jg p_z_begin

    a_o_begin :
        cmp al, 'i'
        jg a_o_H

        a_o_L:
            sub al, 48
            mov ah, al
            jmp a_o_end

        a_o_H:
            sub al, 41
            mov ah, al

    a_o_end:
        mov al, '0'
        stosw
        jmp loop_start

    p_z_begin:
        cmp al, 'y'
        jg p_z_H
```

```

        p_z_L :
            sub al, 64
            mov ah, al
            jmp p_z_end

        p_z_H :
            sub al, 57
            mov ah, al

p_z_end :
    mov al, '1'
    stosw
    jmp loop_start

otherAB :
    cmp al, 'A'
    jl other
    cmp al, 'Z'
    jg other

    add al, 32

    cmp al, 'o'
    jg p_z_begin

    cmp al, 'o'
    jle a_o_begin

    other :
stosb
    jmp loop_start
    loop_final :
stosb
};
std::cout << output << std::endl;
file << output;
file.close();
system("pause");
return 0;
}

```