

**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»**  
**Тема: Представление и обработка символьной информации с**  
**использованием строковых команд.**

Студент гр. 9383

Чумак М.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

### **Цель работы.**

Научиться обрабатывать строку на языке Ассемблера. Написать программу, включая ассемблерную часть в ЯВУ, по принципу in-line.

### **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 ( $\leq 80$ ), 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 №23:**

Преобразование всех строчных латинских букв входной строки в заглавные, а шестнадцатеричных цифр в инверсные, остальные символы входной строки передаются в выходную строку непосредственно.

### **Выполнение работы.**

В ходе работы была реализована программа на языке C++ с in-line вставкой языка Ассемблера. Функция start\_message() – функция для вывода титульной таблички с указанием вида преобразования и автора программы.

Функция main() – функция считывания строки, а также её обработки при помощи языка Ассемблера. Если нашлась латинская буква нижнего регистра, то она заменяется на такую же букву верхнего регистра. Если нашлась шестнадцатеричная цифра, то заменяем ее на инверсионную. В конце выводим обработанную строку в консоль и в файл.

**Тестирование.**

Входные данные

&lt;пусто&gt;

abcdefABCDEF

ABrACADABrA0123456789

Выходные данные

&lt;пусто&gt;

ABCDEF543210

54R535254R5FEDCBA9876

**Выводы.**

Получены знания об обработке строк на языке ассемблера. Написана программа, в которой ассемблерная часть включена в ЯВУ.

## ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММ

### Файл lab4.asm

```
#include <iostream>
```

```
#include <fstream>
```

```
using namespace std;
```

```
void start_message() {
```

```
    cout << "Вариант №23." << endl
```

```
        << "Вид преобразования следующий:" << endl
```

```
        << "Преобразование всех строчных латинских букв входной строки в  
заглавные, а" << endl
```

```
        << "шестнадцатиричных цифр в инверсные, остальные символы  
входной строки передаются в" << endl
```

```
        << "выходную строку непосредственно." << endl
```

```
        << "Работу выполнил студент группы 9383 Чумак Михаил." << endl;
```

```
}
```

```
int main() {
```

```
    setlocale(0, "");
```

```
    start_message();
```

```
    char str1[80], str2[80];
```

```
    cout << "Введите строку: ";
```

```
    cin.getline(str1, 80);
```

```
    __asm {
```

```
        sub eax, eax; eax = 0
```

```
        sub esi, esi; esi = 0
```

```
        lea edi, str1; edi указывает на начало str1
```

```
    start:
```

mov al, [edi]; считываем текущий символ str1 по индексу edi в al  
cmp al, 'z'; если больше чем z,  
jg write; то записываем данные в выходную строку  
cmp al, 'a'; если меньше чем a,  
jl check\_hex; то проверяем, является ли число шестнадцатеричным,  
sub al, 0x20; иначе переводим латинскую букву к верхнему регистру  
jmp write; и записываем символ в выходную строку

check\_hex:

cmp al, 0x30; если меньше чем 0,  
jl write; то записываем символ в выходную строку  
cmp al, 0x46; если больше чем F,  
jg write; то записываем символ в выходную строку  
cmp al, 0x3A; если входим в диапазон цифр 0 - 9,  
jl sub\_code\_numbers; то вычитаем коды на 0x30  
cmp al, 0x41; если не входим в диапазон A - F,  
jl write; то записываем символ между двумя диапазонами в

выходную строку

sub al, 0x7; вычитаем коды на 0x7 для A - F и далее вычитаем 0x30

sub\_code\_numbers:

sub al, 0x30

xor al, 0xF

add\_codes:

add al, 0x30

cmp al, 0x3A; если встретили цифру

jl write; то выводим её

add al, 0x7; иначе это из A - F и прибавляем 0x7

write:

mov str2[esi], al; помещаем текущий символ в выходную строку

cmp al, 0; если был встречен нулевой символ,

je end\_func; то заканчиваем

inc edi; иначе увеличиваем индексы

inc esi

jmp start; и возвращаемся к считыванию строки

end\_func:

}

cout << str2;

ofstream output;

output.open("out.txt");

output << str2;

output.close();

return 0;

}