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

ОТЧЕТ

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

Студент гр. 9383 _____ Камзолов Н.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2020

Цель работы.

Научиться обрабатывать строки на языки ассемблера. Написать программу включая ассемблерную часть в ЯВУ по принципу (in-line).

Текст задания(Вариант 4).

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).

Ход работы.

Функция initialization() **–** функция для печати таблички, в которой информация о виде преобразования строки и информации об авторе.

Функция main() – здесь происходит считывание строки, а также, встроенный код Ассемблера, в котором рекурсивным образом обрабатывается строка. Если нашлась латинская буква верхнего регистра, она заменяется на такую же букву нижнего регистра. Если нашлась восьмеричная цифра, то заменяем ее на инверсивную к ней.

Затем выводим обработанную строку в консоль и в файл.

Тестирование.

Входные данные	Ожидаемый результат	Результат работы программы
<пустая строка>	<пустая строка>	<пустая строка>
ADCadc01234567	adcadc76543210	adcadc76543210
Hello There!	hello there!	hello there!

Вывод.

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

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ

Название файла: Source.cpp

```
#include "iostream"
#include <stdio.h>
#include <fstream>
#define N 80

using namespace std;

void initialization()
{
	cout << ".....\n";
	cout << ". Вид преобразования: Преобразование всех заглавных .\n"
		<< ". латинских букв входной строки в строчные, а .\n"
		<< ". восьмеричных цифр в инверсные, остальные .\n"
		<< ". символы входной строки передаются в .\n"
		<< ". выходную строку непосредственно. .\n";
	cout << ". Разработал Камзолов Никита, студент группы 9383 .\n";
	cout << ".....\n";
}


int main()
{
	setlocale(0, "");
	initialization();
	char str[N+1];
	cout << "Введите строку\n";
	char c;
	cin.getline(str, N);
	char str_out[N * 2 + 1];
	_asm {
```

```
sub eax, eax; eax = 0
sub esi, esi; esi = 0
lea edi, str; edi указывает на начало str
mov ecx, 80; счетчик - 80
f1:
mov al, [edi]; считывает текущий символ str по индуксу edi в al
    cmp al, 'Z'; если больше чем Z, то просто пихаем символ в выходную строку
    jg writeOut
    cmp al, 'A'; если меньше чем A, то идем проверять на восьмеричное число
    jl octalCheck
    add al, 0x20; переводим в нижний регистр
    jmp writeOut
OctalCheck:
    cmp al, 0x30; если меньше чем 0, то просто пихаем символ в выходную строку
    jl writeOut
    cmp al, 0x37; если больше чем 7, то просто пихаем символ в выходную строку
    jg writeOut
    sub al, 0x30; вычитаем 30, чтобы получить число
    xor al, 0x7; инвертируем 3 бита
    add al, 0x30; добавляем 30, чтобы получить символ
writeOut:
    mov str_out[esi], al; помещаем текущий символ в выходную строку
    cmp al, 0; если нулевой символ, то заканчиваем выполнять код Ассемблера
    je f_end
    inc edi; увеличиваем индексы
    inc esi
    jmp f1; возвращаемся к началу
f_end:
}
cout << str_out;
```

```
        ofstream file;
        file.open("out.txt");
        file << str_out;
        file.close();
        return 0;
}
```