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 _____ Литягин С.М.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Изучить представление и обработку символьной информации с использованием строковых команд.

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 №11:

Преобразование введенных во входной строке десятичных цифр в двоичную СС, остальные символы входной строки передаются в выходную строку непосредственно.

Выполнение работы:

Программа была разработана в среде разработки VisualStudio на языке C++.

Входная строка записывается в массив символов strin (80 символов) с помощью функции fgets. Выходная строка записывается в массив символов strout (320 символов), он же и выводится в консоль и в файл out.txt после выполнения программы.

Блок ассемблерного кода находится после ключевого слова __asm. При работе со строками: чтение происходит из памяти по адресу DS:ESI, запись происходит в ячейку памяти по адресу ES:EDI. Поэтому, регистру ES мы присваиваем значение регистра DS. Регистрам ESI и EDI присваиваем значение смещения strin и strout соответственно. Далее с метки line начинается обработка строки. С помощью команды lodsb считывается в регистр AL из strin один байт, т.е. один символ. Данный символ сравнивается с помощью команды cmp последовательно с ‘2’, ‘3’, ‘4’, ‘5’, ‘6’, ‘7’, ‘8’ и ‘9’. Если символ равен одному из перечисленных, то он заменяется на ‘10’, ‘11’, ‘100’, ‘101’, ‘110’, ‘111’, ‘1000’, ‘1001’ соответственно. Замена ‘2’ и ‘3’ происходит следующим образом: в регистр ax помещается соответствующая запись в обратном порядке, а затем с помощью stosw (вносим 2 байта, поскольку записывается 2 символа) отправляется в выходной массив strout. Замена ‘4’-‘7’ происходит следующим образом: в регистр AX помещается первые 2 символа соответствующей записи в обратном порядке, а затем с помощью stosw отправляется в выходной массив strout; потом в регистр AL помещаем последний оставшийся символ и с помощью stosb (вносим 1 байт, поскольку символ 1) записываем его в выходной массив. Замена ‘8’ и ‘9’ происходит так: в регистр EAX помещаются соответствующая символам запись в обратном порядке, потом, с помощью stosd (поскольку заносим 4 символа, то и записываем 4 байта), отправляется в выходной массив. Если же символ входной строки не равен ни одному из перечисленных символов, то с помощью stosb данный символ вносится в выходной массив.

После замены или внесения исходного символа переходим на метку final; если по смещению ESI находится символ конца строки, то работа с блоком ассемблерного кода заканчивается. Результат выводится в консоль и в файл “out.txt”.

Исходный код программы см. в приложении А.

Тестирование:

Для проверки работоспособности были проведены тесты, результаты представлены в таблице 1.

Таблица 1 – Тестирование и результаты

Номер теста	Входные данные	Выходные данные	Верность результата
1	1 2 3 4 5 6 7 8 9 10 11 12	1 10 11 100 101 110 111 1000 1001 10 11 110	верно
2	fffffffffffffffff234fffffffffff fffffffffffffffffffffffffffffffffff fffffffffffff231fffffffffffffff fffffffff78 (введено более 80 символов)	fffffffffffffffff1011100fffffffffff fffffffffffffffffffffffffffffffffff ffffff (обработано ровно 80 символов согласно условию)	верно
3	12345678900987654321	11011100101110111100010010 01001100011111010110011101	верно

Выводы.

В ходе работы было изучено представление и обработка символьной информации с использованием строковых команд. Также была написана программа, преобразующая в строке десятичные цифры в двоичную сс.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb4.cpp

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

char strin[81];
char strout[321];

int main()
{
	std::cout << "Author: Semyon Lityagin, group 0382" << std::endl <<
"The program outputs a string in which it converts all numbers from
decimal ns to binary ns; without changing the remaining characters" <<
std::endl;
	fgets(strin, 81, stdin);
	strin[strlen(strin) - 1] = '\0';
	__asm {
		push ds
		pop es
		mov esi, offset strin
		mov edi, offset strout
		line:
			lodsb

			cmp al, '2'
			jne skip1
			mov ax, '01'
			stosw
			jmp final

		skip1:
			cmp al, '3'
			jne skip2
			mov ax, '11'
			stosw
			jmp final

		skip2:
			cmp al, '4'
			jne skip3
			mov ax, '01'
			stosw
			mov al, '0'
			stosb
			jmp final

		skip3:
			cmp al, '5'
			jne skip4
			mov ax, '01'
			stosw
```

```
            mov al, '1'
            stosb
            jmp final

        skip4:
            cmp al, '6'
            jne skip5
            mov ax, '11'
            stosw
            mov al, '0'
            stosb
            jmp final

        skip5:
            cmp al, '7'
            jne skip6
            mov ax, '11'
            stosw
            mov al, '1'
            stosb
            jmp final

        skip6:
            cmp al, '8'
            jne skip7
            mov eax, '0001'
            stosd
            jmp final

        skip7:
            cmp al, '9'
            jne skip8
            mov eax, '1001'
            stosd
            jmp final

        skip8:
            stosb

        final:
            mov  ecx, '\0'
            cmp  ecx, [esi]
            je   lineEnd; выход, если был найден конец
            jmp  line
        lineEnd:
    };
    std::cout << strout;
    FILE* f;
    fopen_s(&f, "out.txt", "w");
    fwrite(strout, sizeof(char), strlen(strout), f);
    return 0;
}
```