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. 0382 _____ Крючков А.М.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

Разработать программу обработки символьной информации.

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 встраивания (in-line).

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

Порядок выполнения работы.

Программа разработана под MSVC x86, текст считывается с консоли и в неё же выводится. Также результат работы записывается в файл с исполняемым файлом.

Сначала в программе считывается символ с помощью команды loadsb. Символы 0123456789ABCDEF переводятся в числа и программа переходит по метке tobin. С помощью функции and происходит побитовое сравнение с 8, 4, 2, 1 и результирующую строку добавляются соответсующие значения.

Вывод.

Были разработана программа обработки текста на ЯВУ с использование ассемблерной вставки.

ТЕСТИРОВАНИЕ

Таблица 1. Результат тестирования.

№ т.	Входные данные	Результат	Комментарий
1	0 1 2 3 4 5 6 7 8 9 A B C D E F	0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111	Верно
2	HELLO, WORLD! 1 + 2 = 3	H1110LLO, WORL1101! 0001 + 0010 = 0011	Верно
3	12345678901234567890123456789012345678901234567890123456789012345678901234567890Z	000100100011010001010110011110001001000000010010001101000101011001111000100100000001001000110100010101100111100010010000000100100011010001010110011110001001000000010010001101000101011001111000100100000001001000110100010101100111100010010000000100100011010001010110011110001001	Верно. Здесь на вход даёся 81 символ. Последний символ игнорируется.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММ

Файл LAB4.CPP

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

char s[81];
char outstr[321];

int main()
{
     fgets(s, 81, stdin);
     s[strlen(s) - 1] = '\0';
     std::cout << "Kryuchkov Artem. Hex -> bin" // Указал автора и что
делает программа
     __asm {
push ds
pop es
mov esi, offset s
mov edi, offset outstr
next :
    lodsb;
    cmp al, '0'
    jl writeSymbol
    cmp al, 'F'
    jg writeSymbol
    cmp al, '9'
    jle digit
    cmp al, 'A'
    jge letter
    jmp writeSymbol
    digit :
        sub al, '0'
        jmp tobin
    letter :
        sub al, 'A'
        add al, 10
        jmp tobin
    tobin :
        mov bl, al
        mov cl, 8
        and cl, bl; 1000 and ?XXX
        jnz writeOne1
        mov al, '0'
        jmp checkSecondBit
        writeOne1:
        mov al, '1'
        checkSecondBit :
        stosb
        mov cl, 4
        and cl, bl; 0100 and X?XX
        jnz writeOne2
        mov al, '0'
        jmp checkThirdBit
        writeOne2 :
        mov al, '1'
```

```
        checkThirdBit :
        stosb
        mov cl, 2
        and cl, bl; 0010 and XX?X
        jnz writeOne3
        mov al, '0'
        jmp checkFourthBit
        writeOne3 :
        mov al, '1'
        checkFourthBit :
        stosb
        mov cl, 1
        and cl, bl; 0001 and XXX?
        jnz writeOne4
        mov al, '0'
        stosb
        jmp checkNewSymbol
        writeOne4 :
        mov al, '1'
        stosb
        jmp checkNewSymbol
    writeSymbol :
        stosb; кладем в выходную строку байт из al
    checkNewSymbol :
        mov  ecx, '\0'
        cmp  ecx, [esi]
        je   Exit; выход из цикла, если текущий символ завершающий
        jmp  next
Exit :
     };
     std::cout << outstr;
     FILE* f;
     fopen_s(&f, "out.txt", "w");
     fwrite(outstr, sizeof(char), strlen(outstr), f);
     return 0;
}
```