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. 1303 _____ Иванов А. С.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Научиться представлять и обрабатывать символьную информацию с использованием строковых команд на языке Ассемблера. Разработать программу, которая обрабатывает введенную пользователем строку, в соответствии с вариантом.

Задание.

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

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.

Ход работы.

Выполнение работы.

Для реализации задачи был написан программный код на языке C++ с использованием принципа встраивания ассемблерной части. С помощью функции fgets входная строка записывается в массив символов input_string, который по условию должен состоять из 80 символов.

Выходная строка записывается в массив из 320 символов. Длина массива определяется след. образом — максимальный результат числа после

преобразования 4 цифры, это F = 1111, то есть если мы введем 80 символов F, то нам понадобится output длиной именно в 320 символов.

Введенная строка начинает обработку с метки line. Команда lodsb отвечает за чтение байта из строки. С помощью команды cmp последовательно сравниваются считанные символы: 1 2 3 4 5 6 7 8 9 A B C D E F. Если символ равен какому-либо из вышеуказанных то он заменяется в соответствии с таблицей:

Шестнадцатеричная система счисления	Двоичная система счисления
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

- 2 3 заменяются на два символа каждый, то их запись происходит следующим образом: в регистр ax помещается в обратном порядке соответствующая запись, а затем отправляется в выходной массив output с помощью stosw.
- 4 5 6 7 заменяются на три символа каждый, поэтому
  замена происходит следующим образом: первые два символа соответствующей записи помещаются в обратном порядке в регистр ax и отправляются в выходной массив с помощью команды stosw, далее оставшийся символ помещается в регистр al с помощью команды stosb и также записывается в массив output.

- 8 9 A B C D E F занимающие четыре символа в двоичной системе счисления заменяются следующим образом: в регистр eax помещается соответствующая запись в обратном порядке и отправляется в выходной массив с помощью команды stosd (команда отвечает за запись двойного слова в строку, после выполнения команды регистр di увеличивается на 4).

Если после сравнения символ не оказался равен ни одному из предыдущих, то с помощью команды stosb она записывается в output. После всех замен и внесений исходных символов в выходной массив переходим к метке next. Если по смещению esi находится символ конца строки, то работа ассемблерного блока заканчивается. Получившийся результат выводится в файл result.txt

Исходный код программы см. в приложении А.

Результаты тестирования программы lab4.exe представлены в табл. 1.

Таблица 1 – Тестирование программы lab4.exe.

№ Теста	Ввод	Вывод
1	0123456789ABCDEF	01101110010111011110001001 10101011110011011110111 (0 — 0, 1 — 1, 2 — 10 и т.д.)
2	ABABABA	10101011101010111010101111 010 (A – 1010, B – 1011 и т. д.)
3	BAD213	10111010110110111 (B – 1011, A – 1010, D – 1101 и т.д.)
4	12893735897AEFD	11010001001111111110110001 00111110101110111111101 (1 — 1, 2 — 10, 8 — 1000 и т.

		д.)
5	7898798989387479119289374DDD DDDFFFFFF	1111000100110001111001100010011000100111100011110011110011110011010001001111111001110111011101111111111111111111111111111 (7 — 111, 8 — 1000, 9 — 1001 и т.д.)

Вывод.

В результате лабораторной работы было изучено представление и обработка символьной информации с использованием строковых команд на языке Ассемблера.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab4.cpp

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

char input[81];
char output[321];

int main() {
std::cout << "Ivanov Artur from 1303\nTask 11: conversion from hex to binary\n";
fgets(input, 81, stdin);
input[strlen(input)] = '\0';

__asm {
          push ds
          pop es
          mov esi, offset input
          mov edi, offset output

     line :
          lodsb
          cmp al, '2'
          jne count3
          mov ax, '01'
          stosw
          jmp next

     count3 :
          cmp al, '3'
          jne count4
```

```
        mov ax, '11'
        stosw
        jmp next

count4 :
        cmp al, '4'
        jne count5
        mov ax, '01'
        stosw
        mov al, '0'
        stosb
        jmp next

count5 :
        cmp al, '5'
        jne count6
        mov ax, '01'
        stosw
        mov al, '1'
        stosb
        jmp next

count6 :
        cmp al, '6'
        jne count7
        mov ax, '11'
        stosw
        mov al, '0'
        stosb
        jmp next

count7 :
        cmp al, '7'
        jne count8
```

```
        mov ax, '11'
        stosw
        mov al, '1'
        stosb
        jmp next

count8 :
        cmp al, '8'
        jne count9
        mov eax, '0001'
        stosd
        jmp next

count9 :
        cmp al, '9'
        jne count10
        mov eax, '1001'
        stosd
        jmp next

count10 :
        cmp al, 'A'
        jne count11
        mov eax, '0101'
        stosd
        jmp next

count11 :
        cmp al, 'B'
        jne count12
        mov eax, '1101'
        stosd
        jmp next
```

```
count12 :
     cmp al, 'C'
     jne count13
     mov eax, '0011'
     stosd
     jmp next

count13 :
     cmp al, 'D'
     jne count14
     mov eax, '1011'
     stosd
     jmp next

count14 :
     cmp al, 'E'
     jne count15
     mov eax, '0111'
     stosd
     jmp next

count15 :
     cmp al, 'F'
     jne letter
     mov eax, '1111'
     stosd
     jmp next

letter :
     stosb

next :
     mov ecx, '\0'
     cmp ecx, [esi]
```

```
            je end
            jmp line
            end :
};

FILE* f;
fopen_s(&f, "result.txt", "w");
fwrite(output, sizeof(char), strlen(output), f);
fclose(f);
return 0;
```