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

Студентка гр. 0383 _____ Куртова К. А.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург
2021

Цель работы.

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 8:

Преобразование введённых во входной строке шестнадцатеричных цифр в десятичную СС, остальные символы входной строки передаются в выходную строку непосредственно.

Ход работы.

В качестве ЯВУ выбран C++, программа написана в среде разработки Microsoft Visual Studio. Входная строка считывается в массив символов *instring* (не более 80 символов), выходная строка заполняется в массив *outstring* (не более 160 символов). Ассемблерный код вставляется in-line с помощью ключевого слова __asm.

Команды со строками берут в качестве источника адрес, лежащий в ESI, а в качестве назначения адрес, лежащий в EDI, поэтому приписываем данным регистрам смещения *instring* и *outstring* соответственно.

Организуем цикл (*loop_string*), проходящий по входной строке и выполняющий необходимые действия. Так как в задании необходимо заменить шестнадцатеричные числа десятичными, то будем проверять каждый новый символ на соответствие следующим символам: A, B, C, D, E, F (латинские).

Каждый новый символ из строки загружается в AL с помощью команды *lodsb* (так как символ занимает 1 байт), после чего значение символа в AL сравнивается сначала с символом конца строки, а затем с символами из набора. Если символ не совпадает ни с одним символом из набора, то неизменный символ записывается в выходную строку с помощью команды *stosb*, после чего совершается переход на начало цикла. Если символ совпадает с одним из символов из набора, то в AX заносится значение данного шестнадцатеричного числа в десятичном представлении (цифры записываются в обратном порядке), после чего значение в AX записывается в выходную строку с помощью команды *lodsw*. Если на следующей итерации загруженный символ равен символу конца строки, то совершается выход из цикла (переход к метке *loop_end*). Полученная строка выводится в консоль и в файл.

Тестирование программы.

Таблица 1 — Тестирование программы

Входные данные	Результирующие данные	Комментарий
Aa Bb Cc Dd Ee Ff Gg Hh	10a 11b 12c 13d 14e 15f Gg Hh	Верно
this is a test string	this is a test string	Верно
THIS IS A TEST STRING	THIS IS 10 T14ST STRING	Верно
11 12 13 14 B C D E	11 12 13 14 11 12 13 14	Верно

Выводы.

В ходе данной лабораторной работы была рассмотрена работа со строками на языке ассемблера и необходимые для этого команды.

ПРОТОКОЛ

Код программы lb4.cpp

```
//Вариант 8
//Преобразование введенных во входной строке шестнадцатиричных цифр в десятичную СС,
//остальные символы входной строки передаются в выходную строку непосредственно.

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

char instring[81];
char outstring[161];

int main() {

   fgets(instring, 81, stdin);

   __asm {
      push ds
      pop es
      mov esi, offset instring
      mov edi, offset outstring

      loop_string :
         lodsb              //Загрузка символа из строки в al
         cmp al, '\0'
         je loop_end        //Если встречен конец строки, выходим из цикла

         cmp al, 'A'
         je case_a
         cmp al, 'B'
         je case_b
         cmp al, 'C'
         je case_c
         cmp al, 'D'
         je case_d
         cmp al, 'E'
         je case_e
         cmp al, 'F'
         je case_f

         stosb              //Если не является ни одним из данных символов, то записываем символ и переходим в начало цикла
```

```
        jmp loop_string

        case_a :
        mov ax, '01'
        stosw
        jmp loop_string

        case_b :
        mov ax, '11'
        stosw
        jmp loop_string

        case_c :
        mov ax, '21'
        stosw
        jmp loop_string

        case_d :
        mov ax, '31'
        stosw
        jmp loop_string

        case_e :
        mov ax, '41'
        stosw
        jmp loop_string

        case_f :
        mov ax, '51'
        stosw
        jmp loop_string

        loop_end :
    }

    std::cout << outstring;

    FILE* fout;
    fopen_s(&fout, "output.txt", "w");
    fputs(outstring, fout);

    return 0;
}
```