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 _____ Корсунов А.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Изучить представление и обработку символьной информации с использованием строковых команд на языке Ассемблер. Написать программу на основе изученного.

Текст задания.

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).

6. Инвертирование введенных во входной строке цифр в десятичной системе счисления (СС) и преобразование строчных русских букв в заглавные, остальные символы входной строки передаются в выходную строку непосредственно.

Ход работы:

int main():

Создается массив символов (размер массива — 80 элементов), переменная типа ofstream для записи строки в файл, вызывается функция table(), которая выводит таблицу с заданием в консоль, после чего происходит считывание символов с консоли, в указатель на строку символов передается значение функции change(char*), которая преобразует строку согласно заданию, после чего происходит запись строки в файл и ее вывод в консоль.

void table():

Функция выводит таблицу с помощью cout.

char* change(char*):

функция в помощью ассемблерной вставки берет по очереди каждый символ переданного массива в качестве аргумента и построчно выполняет команды.

Новые операнды, используемые в программе:

- lodsb – загружает в регистр al передаваемый байт из строки
- stosb – загружает байт в строку
- loop – уменьшает cx, пока cx не станет равно 0, когда станет равно 0, то перейдет по переданной метке

Примеры работы программы:

1) Ввод: 123456789

Вывод: 876543210

2) Ввод: 2156еоке34пр

Вывод: 7843ЕОКЕ65ПР

3) Ввод: кп3 ре КНЕ 34 КР кр !;% ывп

Вывод: КП6 РЕ КНЕ 65 КР КР !;% ЫВП

4) Ввод: йцукенгшщзхъфывапролджэячсмитьбюё

Вывод: ЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮЁ

5) Ввод: Хорошая программа, интеренсные тесты, очень люблю ассемблер :)

Вывод: ХОРОШАЯ ПРОГРАММА, ИНТЕРЕНСНЫЕ ТЕСТЫ, ОЧЕНЬ ЛЮБЛЮ АССЕМБЛЕР :)

Вывод:

Изучены представление и обработка символьной информации с использованием строковых команд на языке Ассемблер. Написана программа на основе изученного.

Приложение А

main.cpp

```
#include <iostream>
#include <fstream>
#include <windows.h>

using namespace std;

void table()
{
   cout << "-----------------------------------------------------\n";
   cout << "|Вид преобразования:                                |\n";
   cout << "|6. Инвертирование введенных во входной строке цифр     |\n";
   cout << "|в десятичной системе счисления (СС) и преобразование   |\n";
   cout << "|строчных русских букв в заглавные, остальные символы   |\n";
   cout << "|входной строки передаются                          |\n";
   cout << "|в выходную строку непосредственно.                 |\n";
   cout << "|Автор: Корсунов Антон                              |\n";
   cout << "-----------------------------------------------------\n";
}

char* change(char* arr)
{
   char* arr_ready = new char[80];
   asm
   (
      "mov  %0, %%rsi\n\t" //запись строк в регистры
      "mov  %1, %%rdi\n\t"
      "mov $80, %%ecx\n\t" //запись числа 80 в регистр cx для операнда cx
```

```
            "f1:"
            "lodsb (%%rsi)\n\t" //загрузка символа в al
            "cmpb $0x30, %%al\n\t"//сравнение символа с кодом цифры 0
            "jl f2\n\t"      //если меньше 0, то переход в f2
            "cmpb $0x39, %%al\n\t" //сравнение символа с кодом цифры 9
            "jg f2\n\t"       //Если больше, то переход в f2
            "neg %%al\n\t"          // по формуле 2*код числа '9' - текущий код
символа
            "add $0x69, %%al\n\t"
            "jmp f_read\n\t"

            "f2:"
            "cmpb $0xB8, %%al\n\t"  //сравнение с кодом 'ё'
            "jne f3\n\t"    //не равно - переход в f3
            "mov $0xA8, %%al\n\t"   //запись 'Ё' вместо 'ё'
            "jmp f_read\n\t"

            "f3:"
            "cmpb $0xE0, %%al\n\t"  //сравнение с кодом 'а'
            "jl f_read\n\t" //меньше - переход в вывод
            "cmpb $0xFF, %%al\n\t"  //сравнение с кодом 'я'
            "jg f_read\n\t" //больще - переход в вывод
            "sub $0x20, %%al\n\t" //получение заглавной

            "f_read:"
            "stosb (%%rdi)\n\t"       //запись символа в выходную строку
            "loop f1\n\t"         //возвращение в f1 пока cx не 0
            ::"m"(arr),"m"(arr_ready)
        );
        return arr_ready;
```

```
}

int main()
{
   SetConsoleCP(1251);
   SetConsoleOutputCP(1251);
   char arr[80];
   ofstream print_file("file_out.txt");
   table();
   cout << "\nВведите строку:\n";
   fgets(arr, 80, stdin);
   char* arr_ready = change(arr);
   print_file << arr_ready;
   cout << "\nПреобразованная строка:\t" << arr_ready;
   delete arr_ready;
   return 0;
}
```