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. 1383 _____ Чернякова А.Д.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2022

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.

Преобразования: Вариант 24

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

Выполнение работы.

Работа выполнена на языке C++ со встраиванием ассемблерного кода.

В начале программы прописываются команды для работы с кириллицей и на экран выводится сообщение-приветствие. Далее в массив input считывается входная строка и начинается ассемблерный блок.

Регистрам esi и edi присваиваются значения смещения входной строки и строки с результатом соответственно.

С метки start начинается цикл, в котором сначала с помощью команды lodsb в al записывается символ из input и затем он сравнивается с символом конца строки. Если в al записан символ конца строки, то происходит условный переход на метку final, в которой символ записывается в конец массива output, и цикл заканчивается.

Если символ, записанный в al, не равен символу конца строки, то происходит переход на метку from_0_to_5, где он сравнивается сначала с символов ‘0’, а потом с символом ‘5’. Если код записанного символа меньше кода символа ‘0’ или больше кода символа ‘5’ по таблице ASCII, то происходит условный переход на метки symbols_check или more_5_less_A соответственно, которые описаны ниже. Если же символ входит в диапазон от ‘0’ до ‘5’, то происходит инвертация шестнадцатиричной цифры следующим образом: в ah записывается код символа ‘F’, затем из него вычитается значение, записанное в al, и прибавляется 48, а в al записывается символ ‘F’. С помощью stosw значение, записанное в регистр ax, записывается в выходную строку и осуществляется переход на метку start.

Метка more_5_less_A. В ней проверяется, входит ли символ, записанный в al, в диапазон от ‘6’ до ‘9’. Если нет, то происходит условный переход на метку more_than_9, описанную ниже. Если да, то инвертация шестнадцатиричной цифры происходит следующим образом: в ah записывается код символа ‘F’, затем из него вычитается значение, записанное в al, и прибавляется 41, а в al записывается символ ‘F’. С помощью stosw значение, записанное в регистр ax, записывается в выходную строку и осуществляется переход на метку start.

Метка more_than_9. В ней происходит инвертация шестнадцатиричной цифры и запись аналогично метке from_0_to_5 в том случае, если символ, записанный в al, входит в диапазон от ‘A’ до ‘F’. В ином случае происходит переход на метку symbols_check.

Метка symbols_check. В ней происходит преобразование строчных русских букв в заглавные, если код символа, записанного в al, входит в диапазон от 224 до 255. Для преобразования из значения, записанного в al, вычитается 32

и с помощью команды stosb полученное в al значение записывается в выходную строку и осуществляется переход на метку start. Если код символа не попадает в заданный диапазон, то происходит переход по метке yo_check, где проверяется, не является ли записанный символ буквой ё. Если является, то он меняется на заглавную букву, в ином случае происходит переход на метку other, где символ из регистра al просто записывается в выходную строку и осуществляется переход на метку start.

В конце программы происходит вывод строки с результатов на экран.

Исходный код программы смотреть в приложении А.

Тестирование.

Результаты тестирования представлены в таблице 1.

Таблица 1 - Результаты тестирования

№	Входные данные	Выходные данные	Комментарии
1.	0 3 5 8 A B F	FF FC FA F7 F5 F4 F0	Результат верный
2.	абвгдя	АБВГДЯ	Результат верный
3.	AF jckd фыва	F5F0 jckd ФЫВА	Результат верный

Выводы.

В ходе работы было изучено представление и обработка символьной информации с использованием строковых команд. Разработана программа, которая обрабатывается символьную строку и преобразуют ее в соответствии с заданием.

.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab4.cpp

```
#include "pch.h"
#include <iostream>
#include <locale>
#include <fstream>

using namespace std;

char input[85];
char output[170];

int main()
{
    setlocale(LC_ALL,
"Russian");
    system("chcp 1251");
    wprintf(L"Чернякова
Александра, гр.
1383\nВариант
24\nИнвертирование
введенных во входной
строке цифр в
шестнадцатиричной СС и
преобразование строчных
русских букв в
заглавные.");
    cin.getline(input, 80);
    __asm {
        mov esi, offset input
        mov edi, offset output
        start :
        lodsb
            cmp al, '\0'
            je final

            from_0_to_5 :
            cmp al, '0'
            jl symbols_check
```

```
    cmp al, '5'
    jg more_5_less_A
    mov ah, 'F'
    sub ah, al
    add ah, 48
    mov al, 'F'
    stosw
    jmp start

    more_5_less_A :
cmp al, '9'
    jg more_than_9
    mov ah, 'F'
    sub ah, al
    add ah, 41
    mov al, 'F'
    stosw
    jmp start

    more_than_9 :
cmp al, 'F'
    jg symbols_check
    mov ah, 'F'
    sub ah, al
    add ah, 48
    mov al, 'F'
    stosw
    jmp start

    symbols_check :
cmp al, 224
    jl yo_check
    cmp al, 255
    jg yo_check
    sub al, 32
    stosb
```

```
        jmp start

        yo_check :
    cmp al, 'ё'
        jne other
        mov al, 'Ё'
        stosb
        jmp start

        other :
    stosb
        jmp start

        final:
    stosb
  };
  cout << output;
  return 0;
}
```