**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**

Студент гр. 1304 Новицкий М.Д.

Преподаватель Кирьянчиков В. А.

Санкт-Петербург

2023

**Цель работы.**

Изучить команды обработки строк на ассемблере и применить на практике.

**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 нa ЯВУ по принципу встраивания (in-line).

Вариант 20:

Заменить введенные во входной строке русские буквы на числа, соответствующие их номеру по алфавиту, представленному в шестнадцатиричной СС, остальные символы входной строки передать в выходную строку непосредственно.

**Выполнение работы.**

После настройки ввода символов в формате расширенной таблицы ASCII посредством стандартных функций C++ system и setlocale происходит считывание входной строки в переменную input, после чего идет ассемблерная вставка, где и обрабатывается входная строка.

char input[81];

char output[162];

int main()

{

system("chcp 1251 > nul");

setlocale(LC\_CTYPE, "rus");

cin.getline(input, 81);

\_\_asm {

mov esi, offset input

mov edi, offset output

После записи смещений входной и выходной строки в соответствующие регистры объявляется метка read. В ней в регистр al загружается очередной символ входной строки. Если он равен нулевому, то происходит прыжок в метку exit\_prog.

read :

lodsb

cmp al, '\0'

je exit\_prog

В exit\_prog записывается символ из al в выходную строку, после чего выполнение ассемблерной ставки завершается.

exit\_prog :

stosb

После проверки на нулевой символ выполняется проверка на букву “ё” – строчную и заглавную.

cmp al, 'ё'

je yo

cmp al, 'Ё'

je yo

Если очередной символ является данной буквой, то записывается ее шестнадцатеричный код в выходную строку и выполняется безусловный

переход обратно к метке read.

yo :

mov ax, 3730h

stosw

jmp read

Далее выполняется проверка на то, является ли очередной символ русской буквой. Если нет, то он записывается. В противном случае буква делается заглавной.

cmp al, 'А'

jb write

cmp al, 'а'

jb check\_yo

sub al, 32

Также, если она идет после “Ё”, то к ее коду прибавляется один.

check\_yo:

cmp al, 'Ж'

jb process

add al, 1

После этого из кода буквы вычитается код заглавной буквы “А”, и прибавляется один для получения десятичного представления числа, соответствующего номеру буквы по алфавиту.

process :

sub al, 'А'

add al, 1

mov bl, al

С помощью команды and и побитового сдвига в al записывается старший разряд шестнадцатеричного представления числа, а в bl – младший. Регистр dl инициализируется нулем. Впоследствии он будет использован для идентификации того, был ли записан старший разряд.

and bl, 15; lower digit

and al, 240; higher digit

shr al, 4

mov dl, 0

Для обработки обоих разрядов используется метка writedigit. В ней происходит добавление к номеру буквы кода символа ‘0’. Если при сложении получился код цифры, то она записывается в выходную строку. В противном случае к коду добавляется число 7 (метка hexletter) (именно столько символов в таблице Win1251 расположено между кодом цифры ‘9’ и английской буквой ‘A’).

writedigit :

add al, '0'

cmp al, '9'

jg hexletter

cmp dl, 1

je write

stosb

jmp nextdigit

hexletter :

add al, 7

cmp dl, 1

je write

stosb

Метка nextdigit используется для перехода между разрядами.

nextdigit :

mov dl, 1

mov al, bl

jmp writedigit

**Тестирование.**

Таблица 1 – Результаты тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |  |
| 1. | абвгдеёжзийклмнопрсту | 0102030405060708090A0  B0C0D0E0F101112131415 | Программа корректно. | работает |
| 2. | АбГ900h | 010204900h | Программа корректно. | работает |
| 3. | Ёё | 0707 | Программа корректно. | работает |

**Выводы.**

Были изучены и применены на практике навыки обработки строк.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Файл main.cpp

#include <iostream>

#include <fstream>

#include <stdio.h>

#include <windows.h>

#include <tchar.h>

using namespace std;

char input[81];

char output[162];

int main()

{

system("chcp 1251 > nul");

setlocale(LC\_CTYPE, "rus");

cin.getline(input, 81);

\_\_asm {

mov esi, offset input

mov edi, offset output

read :

lodsb

cmp al, '\0'

je exit\_prog

cmp al, 'ё'

je yo

cmp al, 'Ё'

je yo

cmp al, 'А'

jb write

cmp al, 'а'

jb check\_yo

sub al, 32

check\_yo:

cmp al, 'Ж'

jb process

add al, 1

process :

sub al, 'А'

add al, 1

mov bl, al

and bl, 15; lower digit

and al, 240; higher digit

shr al, 4

mov dl, 0

writedigit :

add al, '0'

cmp al, '9'

jg hexletter

cmp dl, 1

je write

stosb

jmp nextdigit

hexletter :

add al, 7

cmp dl, 1

je write

stosb

jmp nextdigit

nextdigit :

mov dl, 1

mov al, bl

jmp writedigit

yo :

mov ax, 3730h

stosw

jmp read

write :

stosb

jmp read

exit\_prog :

stosb

};

cout << output;

return 0;

}