**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Организация ЭВМ и систем»**

# **Тема:** **Представление и обработка символьной информации с использованием строковых команд**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9383 |  | Поплавский И. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2020

**Цель работы.**

Изучение представления и обработки символьной информации с использованием строковых команд на ассемблере.

**Постановка задачи.**

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 16. Преобразование введенных во входной строке русских букв в латинские в соответствие с правилами транслитерации, остальные символы входной строки передаются в выходную строку непосредственно.

**Выполнение работы.**

Сначала происходит инициализация массивов для ввод и вывода на с++, а после идет ассемблерная вставка \_\_asm:

В начале происходит подготовка под основной круг traverse(В зависимости от длины исходной строки настраиваются регистры отвечающие за индексы). После с помощью сравнений cmp проходимся по массиву и если находим русскую букву, то прыгаем на метку, обрабатываем ее и если еще остались символы в строке повторяем круг traverse до конца.

**Вывод.**

В результате выполнения лабораторной работы были изучены представления и обработки символьной информации с использованием строковых команд на ассемблере.

Приложение А

Разработанный код программы

#include <iostream>

#include <stdio.h>

#define N 80

using namespace std;

int main()

{

system("chcp 1251 > nul");

char \_str[N + 1];

cout << "ЛР4. Поплавский Иван, Вариант 16.\n";

char str\_out[N \* 4];

int i = 0;

cin.getline(\_str, N);

\_asm {

sub eax, eax;

mov al, 0; in al code of str ending symbol

mov ecx, N; ecx = N

lea edi, \_str; edi now points at start of \_str

repne scas; ecx now contains N - str.length

sub ecx, N; ecx = -str.length

neg ecx; ecx = str.length

mov edx, ecx; edx = ecx

sub edi, edi; edi == 0

sub esi, esi; esi == 0

traverse:

mov edi, edx; edi = edx

sub edi, ecx; edi - points at last element in str, when we subtracting ecx we pointing to currentIdx, as ecx decreasing every iteration

mov al, \_str[edi]; al contains currentElement

cmp al, 'а'

je writeSymbol\_1

cmp al, 'б'

je writeSymbol\_2

cmp al, 'в'

je writeSymbol\_3

cmp al, 'г'

je writeSymbol\_4

cmp al, 'д'

je writeSymbol\_5

cmp al, 'е'

je writeSymbol\_6

cmp al, 'ё'

je writeSymbol\_7

cmp al, 'ж'

je writeSymbol\_8

cmp al, 'з'

je writeSymbol\_9

cmp al, 'и'

je writeSymbol\_10

cmp al, 'й'

je writeSymbol\_11

cmp al, 'к'

je writeSymbol\_12

cmp al, 'л'

je writeSymbol\_13

cmp al, 'м'

je writeSymbol\_14

cmp al, 'н'

je writeSymbol\_15

cmp al, 'о'

je writeSymbol\_16

cmp al, 'п'

je writeSymbol\_17

cmp al, 'р'

je writeSymbol\_18

cmp al, 'с'

je writeSymbol\_19

cmp al, 'т'

je writeSymbol\_20

cmp al, 'у'

je writeSymbol\_21

cmp al, 'ф'

je writeSymbol\_22

cmp al, 'х'

je writeSymbol\_23

cmp al, 'ц'

je writeSymbol\_24

cmp al, 'ч'

je writeSymbol\_25

cmp al, 'ш'

je writeSymbol\_26

cmp al, 'щ'

je writeSymbol\_27

cmp al, 'ъ'

je writeSymbol\_28

cmp al, 'ы'

je writeSymbol\_29

cmp al, 'ь'

je writeSymbol\_30

cmp al, 'э'

je writeSymbol\_31

cmp al, 'ю'

je writeSymbol\_32

cmp al, 'я'

je writeSymbol\_33

jmp writeSymbol

writeSymbol\_1 :

mov str\_out[esi], 'а'

inc esi

loop traverse

writeSymbol\_2 :

mov str\_out[esi], 'b'

inc esi

loop traverse

writeSymbol\_3 :

mov str\_out[esi], 'v'

inc esi

loop traverse

writeSymbol\_4 :

mov str\_out[esi], 'g'

inc esi

loop traverse

writeSymbol\_5 :

mov str\_out[esi], 'd'

inc esi

loop traverse

writeSymbol\_6 :

mov str\_out[esi], 'e'

inc esi

loop traverse

writeSymbol\_7 :

mov str\_out[esi], 'J'

inc esi

mov str\_out[esi], 'e'

inc esi

loop traverse

writeSymbol\_8 :

mov str\_out[esi], 'z'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_9 :

mov str\_out[esi], 'z'

inc esi

loop traverse

writeSymbol\_10 :

mov str\_out[esi], 'i'

inc esi

loop traverse

writeSymbol\_11 :

mov str\_out[esi], 'y'

inc esi

loop traverse

writeSymbol\_12 :

mov str\_out[esi], 'k'

inc esi

loop traverse

writeSymbol\_13 :

mov str\_out[esi], 'l'

inc esi

loop traverse

writeSymbol\_14 :

mov str\_out[esi], 'm'

inc esi

loop traverse

writeSymbol\_15 :

mov str\_out[esi], 'n'

inc esi

loop traverse

writeSymbol\_16 :

mov str\_out[esi], 'o'

inc esi

loop traverse

writeSymbol\_17 :

mov str\_out[esi], 'p'

inc esi

loop traverse

writeSymbol\_18 :

mov str\_out[esi], 'r'

inc esi

loop traverse

writeSymbol\_19 :

mov str\_out[esi], 's'

inc esi

loop traverse

writeSymbol\_20 :

mov str\_out[esi], 't'

inc esi

loop traverse

writeSymbol\_21 :

mov str\_out[esi], 'u'

inc esi

loop traverse

writeSymbol\_22 :

mov str\_out[esi], 'f'

inc esi

loop traverse

writeSymbol\_23 :

mov str\_out[esi], 'k'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_24 :

mov str\_out[esi], 'c'

inc esi

loop traverse

writeSymbol\_25 :

mov str\_out[esi], 'c'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_26 :

mov str\_out[esi], 's'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_27 :

mov str\_out[esi], 'j'

inc esi

mov str\_out[esi], 's'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_28 :

mov str\_out[esi], 'h'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_29 :

mov str\_out[esi], 'i'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_30 :

mov str\_out[esi], 'j'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_31 :

mov str\_out[esi], 'e'

inc esi

mov str\_out[esi], 'h'

inc esi

loop traverse

writeSymbol\_32 :

mov str\_out[esi], 'j'

inc esi

mov str\_out[esi], 'u'

inc esi

loop traverse

writeSymbol\_33 :

mov str\_out[esi], 'j'

inc esi

mov str\_out[esi], 'a'

inc esi

loop traverse

writeSymbol :

mov str\_out[esi], al

inc esi

loop traverse

mov str\_out[esi], 0

}

cout << str\_out;

return 0;

}