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 на 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 c++, 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, 'a'
         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], 'a'
    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;
      }
```