**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

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

Тема «**Представление и обработка символьной информации с использованием строковых команд.**»

|  |  |  |
| --- | --- | --- |
| Студент гр. 1303 |  | Жилин И.А. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2022

## Цель работы.

Изучить представление и обработку символьной информации с использованием строковых команд на языке Ассемблера. Разработать программу, которая обрабатывает строку.

## 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).

## Выполнение работы.

Формирование выходной строки только из цифр и латинских букв входной строки.

В начале выполнения программы в консоль выводится строка с именем, номером группы и заданием, после чего следует ввести строку для обработки. С помощью getline считывается не более 81 символа с учетом символа окончания строки ‘\0’. Setlocale и system дают нам возможность работать с кириллицей.

Далее объявляется ассемблерный блок через ключевое слово \_\_asm, в котором происходит считывание каждого символа введенной строки с помощью команды lodsb. В процессе выполнения программа проверяет каждый символ на вхождения в промежутки: ‘0’ – ‘9’; ‘A’ – ‘Z’; ‘a’ – ‘z’. Если символ выходит за эти границы, то его программа игнорирует, для этого используются метки и команды перехода к меткам: jle, jge, jmp. Если же символ подходит под условие, то команда stosb записывает его в es:edi. Если встречается символ конца строки, ассемблерный блок заканчивается.

В конце, полученная строка на ЯВУ выводится на экран и записывается в текстовый файл.

Исходный код программы см. в приложении А.

Результаты тестирования программы ConsoleApplication.exe представлены в табл. 1.

Таблица 1 – Тестирование программы ConsoleApplication.exe.

|  |  |  |  |
| --- | --- | --- | --- |
| № Теста | Ввод | Вывод | Результат |
| 1 | 123abcабв !!! | 123abc | Верно |
| 2 | 55 @@ йцукен qwerty | 55qwerty | Верно |
| 3 | 789[q]{б} | 789q | Верно |

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 1 – Трансляция работы программы*

## Вывод.

В результате лабораторной работы была изучена обработка символьной информации с использованием ассемблерного блока в коде на ЯВУ.

## ПРИЛОЖЕНИЕ А

## Тексты исходных файлов программ ConsoleApplication.cpp.

#include <iostream>

#include <fstream>

using namespace std;

char inStr[81];

char outStr[81];

int main() {

system("chcp 1251 > nul");

setlocale(LC\_CTYPE, "rus");

cout << "Жилин Илья 1303\nФормирование строки только из цифр и латинских букв входной строки\n";

cin.getline(inStr, 81);

ofstream res;

res.open("result.txt", ios::out | ios::trunc);

\_\_asm {

mov esi, offset inStr

mov edi, offset outStr

symCheck:

lodsb

cmp al, '\0'

je endBlock

cmp al, '9'

jle checkDigit

cmp al, 'Z'

jle checkUpper

cmp al, 'z'

jle checkLower

jmp symCheck

checkDigit:

cmp al, '0'

jge writeSym

jmp symCheck

checkUpper:

cmp al, 'A'

jge writeSym

jmp symCheck

checkLower :

cmp al, 'a'

jge writeSym

jmp symCheck

writeSym:

stosb

jmp symCheck

endBlock:

};

cout << outStr;

res << outStr;

res.close();

return 0;

}