

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра МОЭВМ**

**ОТЧЕТ**  
**по лабораторной работе №4**  
**по дисциплине «Организация ЭВМ и систем»**  
**Тема: Представление и обработка символьной информации с**  
**использованием строковых команд.**

Студент гр. 1303

Беззубов Д.В.

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  $N_{\max}$  ( $\leq 80$ ), с клавиатуры в заданную область памяти - на ЯВУ; если длина строки превышает  $N_{\max}$ , 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.

### **Ход работы.**

#### **Выполнение работы.**

Разработана программа на языке C++ с использованием ассемблерных вставок. При ее запуске в консоль выводится строка, содержащее имя, фамилию, номер группы, а также задание. Затем выводится сообщение с просьбой ввести входную строку. С помощью метода *getline()* считывается не более 81 символа с учетом нуля-терминатора. *Setlocale* и *system* дают нам возможность работать с кириллицей.

Далее объявляется ассемблерная вставка через ключевое слово `__asm`. Настраиваем расширенные сегменты ESI и EDI на входную и выходную строки соответственно. Затем создается метка *checking*, по которой будем переходить при проверке очередного символа исходной строки. С помощью

команды `lodsb` выгружается очередной символ в нижний байт регистра-аккумулятора (AX). В процессе выполнения программа проверяет каждый символ на вхождение в промежутки 'А'- 'я' и 'А' – 'z'. В случае, если очередной символ попадает в один из данных промежутков, то совершается переход по метке `save`, где символ записывается в выходную строку с помощью команды `stosb`, которая выгружает символ из регистра-аккумулятора в память.

Для перехода по меткам используются следующие команды условного перехода: `je`, `jg`, `jl`, `gle`, а так же команды безусловного перехода `jmp`. Если встречается символ конца строки, то совершается переход по метке `end`, после чего ассемблерная вставка оканчивается.

В конце, полученная строка выводится на экран и записывается в текстовый файл с помощью языка ВУ.

Исходный код программы см. в приложении А.

Результаты тестирования программы `lab4.exe` представлены в табл. 1.

Таблица 1 – Тестирование программы `lab4.exe`.

№ Теста	Ввод	Вывод
1	!@#%212345йцукенqwerty	йцукенqwerty
2	!@#% 1234QWERTYЙЦУКЕН	QWERTYЙЦУ КЕН
3	JSHDFKJЙЦУФЫIDSJFHSKYВЛА ОР	JSHDFKJЙЦУФ ЫIDSJFHSKYВ ЛАОР
4	1234567."':#\$@\$(*&	

### **Вывод.**

В результате лабораторной работы была изучена обработка символьной

информации с использованием языка ассемблера, а также разработана программа на языке ВУ, использующая вставку на языке ассемблера.

## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab4.cpp

```
#include <iostream>
#include <fstream>
char input_string[81];
char output_string[81];

int main() {

    system("chcp 1251 > nul");
    setlocale(LC_CTYPE, "rus");

    std::cout << "Беззубов Даниил 1303\nЗадание: Формирование выходной
строки только из русских и латинских букв входной строки\n";

    std::cout << "Введите строку\n";
    std::cin.getline(input_string, 81);

    std::ofstream file;
    file.open("result.txt");

    __asm {
        push ds
        pop es
        mov esi, offset input_string
        mov edi, offset output_string
        checking:

        lodsb
        cmp al, '\0'
        je end

        cmp al, 'A'
        jl checking

        cmp al, 'я'
        jle save

        cmp al, 'А'
        jl checking

        cmp al, 'z'
        jg checking

        save:
        stosb
        jmp checking
        end:
    };

    std::cout << "Строка из символов латиницы и кириллицы\n";
```

```
std::cout << output_string;
file << output_string;
file.close();

return 0;
}
```