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 _____ Чернуха В.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, обрабатывающую строку в соответствии с заданием варианта.

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 на ЯВУ по принципу страивания (in-line).

Ход работы.

Задание 2 варианта: формирование выходной строки только из цифр и латинских букв входной строки.

Выполнение работы.

При запуске программы сначала выводится фамилия, имя и номер группы автора программы и так же текст задания варианта. Далее выводится просьба о вводе пользователем строки. После ввода строки и записи её в input_str начинается фрагмент обработки строки на языке Ассемблера.

В фрагменте кода на Ассемблере происходит запись в output_str нужных символом — латиница и цифры. На метке checking используется lodsb для получения нового символа из строки и записи его в al. Далее происходит ряд проверок, содержимое al проверяется на вхождение в промежуток от «a» до «z», от «A» до «Z», от «0» до «9» или на равенство «\0» (в этом случае происходит переход на метку endprog и часть кода на ассемблере завершается). Проверка на вхождение осуществляется с помощью условных переходах и сравнений . В случае вхождения содержимого al в один из промежутков происходит запись символа в выходную строку(с помощью stosb) и переход на checking, в ином случае происходит переход опять на метку checking но без записи символа в строку.

После завершения части кода на ассемблере выводится выходная строка и происходит запись её в файл.

Исходный код программы см. в приложении А.

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

Таблица 1 – Тестирование программы main.exe.

№ Теста	Вввод	Вывод
1	dfgdfgdfg	dfgdfgdfg
2	fdgвапвапвап	fdg
3	ываыва234234gdfgdfg	234234gdfgdfg
4	Hdf34324ыФвasfds34fdыёG	Hdf34324asfds34fdG

Вывод.

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

Приложение А

Исходный код программы

Название файла: main.cpp

```
#include <iostream>
#include <fstream>
char input_str[81];
char output_str[81];

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

    std::cout << "Автор: Чернуха Владимир, 1303. Задание: вывести строку только из латинских букв и цифр.\nВведите строку\n";
    std::cin.getline(input_str, 81);

    __asm {
        push ds
        pop es
        mov esi, offset input_str
        mov edi, offset output_str

        checking :
            lodsb
            cmp al, '\0'
            je endprog

            cmp al, 'a'
            jge checklow

            cmp al, 'A'
            jge checkup

            cmp al, '0'
            jge checkdigit

            jmp checking

        checkdigit:
```

```
                cmp al, '9'
                jle write
                jmp checking

            checkup:
                cmp al, 'Z'
                jle write
                jmp checking

            checklow:
                cmp al, 'z'
                jle write
                jmp checking

            write:
                stosb
                jmp checking

            endprog:
        };

        std::ofstream fp;
        fp.open("text.txt");
        std::cout << "Выходная строка, только латиница и цифры:\n" <<
output_str;
        fp << output_str;
        fp.close();

        return 0;
    }
```