ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)»

Кафедра МО ЭВМ

ОТЧЁТ

по лабораторной работе № 4

по дисциплине «Организация ЭВМ и систем»

Тема: Представление и обработка символьной информации с использованием стоковых команд.

Студент гр. 1303 _____ Мусатов Д.Е.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Разработать программу на языке Ассемблера, обрабатывающую символьную информацию с использованием строковых команд.

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} 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 её запись в файл — на ЯВУ.

Ассемблерную часть программы включить в программу на ЯВУ по принципу встраивания (in-line).

Вариант 4: Преобразование всех заглавных латинских букв входной строки в строчные, а восьмеричных цифр в инверсные, остальные символы входной строки передаются в выходную строку непосредственно.

Выполнение работы.

Для выполнения лабораторной работы был использован ЯВУ Си с подключёнными библиотеками locale.h и wchar.h для возможности работы с кириллицей.

Программа создаёт массивы для хранения входной и выходной строки, считывает строку из потока ввода, а после передаёт их в ассемблерную вставку: входная строка передаётся в регистр rsi, выходная строка передаётся в регистр rdi, также мы запрещаем использовать

компилятору регистр rax, поскольку он будет использоваться для считывания букв строки.

Алгоритм программы в ассемблерной вставке следующий:

С помощью инструкции lodsd из памяти, на который указывает регистр rsi, считывается 4 байта (размер символа wchar_t), которые записываются в регистр eax, после чего значение в регистре rsi увеличивается на 4. После происходит проверка, окончилась ли строка? (является ли считанный символом нулевым). Если да, то программа записывает нулевой символ в выходную строку (запись происходит с помощью инструкции stosd, которая записывает данные из регистра eax в память, на которую указывает регистр rdi с увеличением последнего на 4).

Если же считанный символ не является концом строки, то происходит проверка: является ли считанный символ меньшим кода латинской буквы ”A” (прописная)? Если является, то программа переходитк проверке на цифру восьмеричной системы счисления. Иначе программа проверяет, меньше ли символ латинской буквы ”Z” (прописная). Если нет, то происходит запись символа в выходную строку. Если да, значит, символ является прописной латинской буквой, поэтому её значение увеличивается на 32, тем самым превращая её в строчную, после чего результат записывается в выходную строку.

Проверка на цифру восьмеричной системы счисления производится сравнением символа с границами: символом цифры ”0” и символомцифры ”7”. Если вне границ, то символ записывается, а иначе символ необходимо инвертировать: символ инвертируется по знаку, после чего к нему прибавляется сумма символов цифр ”0” и ”7”, после чего символ записывается в выходную строку.

По выходу из ассемблерной вставки в выходной строке будет находиться обработанная строка, которая будет выведена на экран.

Тестирование.

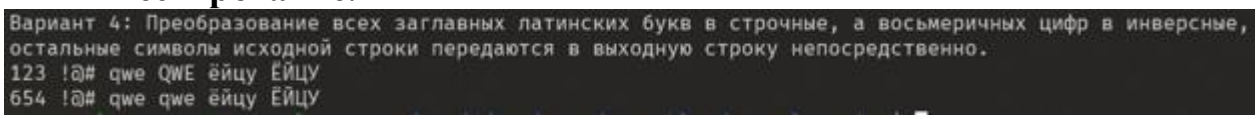

Рисунок 1 – Тестирование программы со входной строкой.

Выводы

В результате лабораторной работы была изучена обработка символьной информации с использованием языка Ассемблера; разработана программа на ЯВУ (Си) с использованием ассемблерной вставки.

Приложение А:
Файл lab4.c

```
#include <locale.h>
#include <stdio.h>
#include <wchar.h>

#define N 81

int main() {
  setlocale(LC_CTYPE, "");

  wprintf(
      L"Мусатов Дмитрий, гр. 1303.\nВариант 4:
Преобразование всех заглавных "
      L"латинских букв в строчные, а восьмеричных цифр
в инверсные, остальные "
      L"символы исходной строки передаются в выходную
строку непосредственно.");

  wchar_t in[N];
  wchar_t out[N];

  fgetws(in, N, stdin);

  asm("process_str:                \n"
      "  xor  rax, rax             \n"
      "  lodsd                     \n"
      "  cmp  eax, 0               \n"
      "  je   end_process          \n"

      "latin_check:                \n"
      "  cmp  eax, 65              \n"
```

```
        "  jl   oct_check             \n"
        "  cmp  eax, 90               \n"
        "  jg   write_ch              \n"
        "  add  eax, 32               \n"
        "  jmp  write_ch              \n"

        "oct_check:                   \n"
        "  cmp  eax, 48               \n"
        "  jl   write_ch              \n"
        "  cmp  eax, 55               \n"
        "  jg   write_ch              \n"
        "  neg  eax                   \n"
        "  add  eax, 103              \n"
        "  jmp  write_ch              \n"

        "write_ch:                    \n"
        "  stosd                      \n"
        "  jmp  process_str           \n"

        "end_process:                 \n"
        "  mov  eax, 0                \n"
        "  stosd                      \n"
        :
        : [in] "S"(in), [out] "D"(out)
        : "rax");

  wprintf(L"%ls", out);

  return 0;
}
```