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

ОТЧЕТ

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

Студент гр. 1383 _____ Федорова О.В.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Разработать программу обработки символьной информации с использованием строковых комманд.

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 встраивания (in-line).

Вариант 21.

Заменить введенные во входной строке латинские буквы на числа, соответствующие их номеру по алфавиту, представленному в шестнадцатиричной СС, остальные символы входной строки передать в выходную строку непосредственно.

Выполнение работы.

В качестве ЯВУ использовался ЯП С.

В начале программы в теле функции main происходит вывод строки с именем и фамилией студента и индивидуальным заданием. После чего происходит считывание строки с консоли, длина которой не превосходит 80 символов по условию гарантированно. Так как в условии задания требуется все латинские буквы преобразовать в число - номер в алфавите, было принято решение игнорировать регистр при записи символов во входную строку. После считывания строки, смены регистра на нижний создается массив для вывода ответа и выделяется память под него.

В ассемблерной вставке перед обработкой строки происходит смещение регистров rdi и rsi на смещение строки для данных и строки для ответа(
для использования stosb и lodsb) . Далее, если символ al не 0, значит строка не кончилась, программа переходит а метку find, чтобы понять, как стоит записать полученный символ. Если al лежит в диапазоне от 97 до 122, значит, в нем записана какая-то буква латинского алфавита, если нет, то просто происходит переход на метку print, которая выводит в строку ответа 1 символ, после чего переходит на следующую итерацию цикла.

Если же полученный символ - буква, то в строку будет выведено 2 символа, так как номера в алфавите лежат в диапазоне от 0 до 25 в 10 СС, в 16СС это будут числа от 00 до 19. Если выяснилось, что символ - буква, то в регистр bl записывается число 16 и вызывается функция div 16, которая по правилам работы языка ассемблер в данном случае запишет в старший байт регистра ax остаток от деления ax на 16, а в младший - целую часть.

Так как целая часть в данном случае может быть только 0 либо 1, дополнительных проверок для вывода этого байта не требуется, потому к результату добавляется 48(чтобы получить либо символ 1 либо символ 0) и выводится в строку ответа. Далее идет обработка остатка от деления на 16. в младший байт регистра ax записывается его старший байт, значение которого может лежать в диапазоне от 0 до 16. Если результат меньше 10, то просто происходит переход на метку print, в противном случае в

младшем байте лежит число, большее 10, которое в 16СС должно быть записано буквой. Для этого происходит переход на метку letter , в которой из символа вычитается 10, и добавляется 65(код символа A), после чего происходит переход на метку print.

Таблица 1- Тестирование программы

№	Входная строка	Выходная строка	Комментарий
1	Q	10	Нумерация с 0, поэтому q 10я по счету
2	P	0F	Нумерация с нуля, 15-я по счету буква, 15 в 16с-с = 0F
3	1,2,3,A,B,C	1,2,3,00,01,02	Запятые, цифры без изменений, A - первая буква, нумерация с 0
4	p,P,q,Q,W,w,r,R	0F,0F,10,10,16,16,11,11	Вне зависимости от регистра ответ правильный
5	q,r,s,t,u,v,w	10,11,12,13,14,15,16	еще проверка
	k,L,m,N,o,p	0A,0B,0C,0D,0E,0F	10й,11й и т.д. символ алфавита

Программный код см. в приложении А.

Выводы.

Разработана программу обработки символьной информации с использованием строковых команд.

Приложение А

Исходный код программы

Название файла: lb4.c

```
    #include <stdio.h>
    #include <string.h>
    #include <stdlib.h>
    #include <ctype.h>

    int main() {
            printf("Fdorova  Oksana  -  1383\nconvert
letter into number in order\n");

           char  *input_str  =  (char  *)  calloc(81,
sizeof(char));
        fgets(input_str, 80, stdin);
        input_str[80] = '\0';
        for(int i = 0; i < 80; i++) {
         input_str[i] = tolower(input_str[i]);
        }

              char   *res   =   (char   *)   calloc(81,
sizeof(char));

        asm volatile(
         "mov rsi, %[str]              \n\t"//lodsb
будет считывать с str
         "mov rdi, %[res]              \n\t"//stosb
для заиси из ax в res
         "mov rcx, 80                    \n\t"
         "for:                           \n\t"
         "    mov rax, 0                 \n\t"
         "    lodsb                      \n\t"
         "    cmp al, 0                  \n\t"
         "    jne find                   \n\t"
         "    mov rcx, 1                 \n\t"
         "    jmp print                  \n\t"
         "find:                          \n\t"
         "    cmp al, 122                \n\t"
         "    ja print                   \n\t"
```

```
            "    cmp al, 97                    \n\t"
            "    jb print                      \n\t"
            "    sub ax, 97                    \n\t"
            "    mov bl, 16                    \n\t"
            "    div bl                        \n\t"
            "    add al, 48                    \n\t"
            "    stosb                         \n\t"
            "    mov al, ah                    \n\t"
            "    cmp al, 9                     \n\t"
            "    ja letter                     \n\t"
            "    add al, 48                    \n\t"
            "    jmp print                     \n\t"
            "letter:                        \n\t"
            "    sub al, 10                        \n\t"
            "    add al, 65                        \n\t"
            "    jmp print                         \n\t"
            "print:                                \n\t"
            "    stosb                             \n\t"
            "    loop for                          \n\t"
            :[res] "=m"(res)
            :[str] "r"(input_str)
             :"rax", "rbx", "rcx", "rdx", "rdi", "rsi",
"memory", "cc"
        );
        res[80] = '\0';
        printf("%s\n", res);

        free(input_str);
        free(res);
        return 0;
    }
```