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. 9383 _____ Ноздрин В.Я.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2020

Цель работы.

Изучить представление и обработку символьной информации. Написать программу, обрабатывающую строку по определенному принципу на языке высокого уровня (C++) с включением фрагмента на языке Ассемблер по принципу встраивания (in-line).

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).

Реализовать программу, выполняющую преобразование всех заглавных латинских букв входной строки в строчные, а десятичных цифр в инверсные, остальные символы входной строке передаются в выходную строку непосредственно.

Ход работы.

Разработана программа на языке C++ и Ассемблер, которая выполняет преобразование латинских заглавных букв в строчные, а десятичных цифр в инверсные.

Замена заглавных букв происходит следующим образом: так как по таблице ASCII заглавные латинские буквы все имеют номер на 20 меньший, чем их строчные версии, для выполнения операции lower необходимо уменьшить значение байта на 20.

Для инверсии цифры необходимо вычесть ее код из кода цифры 9 (он равен 0x39). Таким образом получается значение инверсной цифры. Далее прибавляя код цифры 0 (он равен 0x30), получаем код инверсной цифры. Для оптимизации данные операции выполняются следующим образом: сначала меняется знак кода цифры, затем к нему прибавляется 0x69, что является

суммой кодом цифр 0 и 9. И в результате в коде инверсной цифры запишется 0x69-{код}, что равно 0x39-{код}+0x30.

Преобразования выполняются на ассемблере в функции **task** с встраиванием кода in-line.

Также в программе есть функции **init**, **readline** и **save**.

init выполняет инициализацию по заданию

readline считывает строку, игнорируя символы, введенные после 80ого.

save выводит результирующую строку на экран и сохраняет ее в файл.

Исходный код программы представлен в приложении А.

Тестирование.

Входные данные	Результат вычислений
0123456789AbCdEfG	9876543210abcdefg
Hello World!	hello world!
Привет Мир!	Привет Мир!
abcABCабвАБВ123	abcabcабвАБВ876

Выводы.

Было изучено представление символьной информации и ее обработка на языке Ассемблер. Была написана программа, обрабатывающая строку по определенному правилу на языке высокого уровня C++ с включением фрагмента на языке Ассемблер по принципу встраивания (in-line).

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Текст файла main.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstring>

#define NMAX 80

void init();
std::string readLine();
void save(std::string result);
std::string task(char* str_in);

int main() {
  init();
  std::cout << "Enter line\n";
  auto input = readLine();
  char* str = new char[80];
  strcpy(str, input.data());
  auto result = task(str);
  save(result);
  return 0;
}

void init(){
  setlocale(LC_ALL, "Russian");
  std::cout << "\\Задание: преобразовать все заглавные латинские буквы   \\\n"
               "\\входной строки в строчные, а десятичные цифры в        \\\n"
               "\\инверсные, остальные символы входной строки передаются \\\n"
               "\\во выходную строку непосредственно                     \\\n"
               "\\Author: BasilNozdrin group 9383                        \\\n";
};

std::string readLine(){
  std::string line;
  std::getline(std::cin, line);
  line = line.substr(0, NMAX);
  return line;
};

void save(std::string result){
```

```
  std::ofstream output("output.txt");
  std::cout << result << "\n";
  if(output.is_open()) {
    output << result << "\n";
    output.close();
  }
};

std::string task(char* str_in) {
  char* str_out = new char[NMAX];
  asm(
    "mov r8, %0         \n"// r8 = %0
    "mov rdi, %1        \n"// rdi = %1
"getChar:               \n"//=================================================
    "mov al, [rdi]      \n"// al = getChar(rdi)
    "inc rdi            \n"// rdi++
    "cmp al, 0          \n"// if (al == '\0')
    "je exit            \n"// break
    "cmp al, 0x80       \n"// if(c < 0x80)
    "jb Char            \n"//
"WChar:                 \n"//=================================================
    "mov ah, al         \n"// ah = al
    "mov al, [rdi]      \n"// al = getChar(rdi) // read second byte of wchar
    "inc rdi            \n"// rdi++
    "jmp writeWChar     \n"// print
"Char:                  \n"//=================================================
    "cmp al, 0x5a       \n"// if(c > 'Z')
    "jg writeChar       \n"// print
    "cmp al, 0x30       \n"// if(c < '0')
    "jl writeChar       \n"// print
    "cmp al, 0x41       \n"// if(c < 'A')
    "jl Number          \n"//
"Lower:                 \n"//=================================================
    "add al, 0x20       \n"// lower(c)
    "jmp writeChar      \n"// print
"Number:                \n"//=================================================
    "cmp al, 0x39       \n"// if(c > '9')
    "jg writeChar       \n"// print
    "neg al             \n"// al = -al           // 0x30 ix '0'
    "add al, 0x69       \n"// 0x69 = 0x39 + 0x30 // 0x39 is '9'
"writeChar:             \n"//=================================================
    "mov [r8], al       \n"// *r8 = c       // 1 byte
    "inc r8             \n"// r8++
    "jmp getChar        \n"// loop
"writeWChar:            \n"//=================================================
    "xchg ah, al        \n"// swap(ah, al)
    "mov [r8], ax       \n"// *r8 = c       //  2 bytes
    "inc r8             \n"// r8++
```

```
    "inc r8             \n"// r8++
    "jmp getChar        \n"// loop
"exit:                  \n"
    "mov [r8], al       \n"// *r8 = c
    :"=m"(str_out)// %0
    :"m"(str_in)  // %1
  );
  return std::string(str_out);
}
```