# 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. 1304 Басыров В.А.

Преподаватель Кирьянчиков В. А.

Санкт-Петербург,2022

**Цель работы****:** Изучить представление символьной информации с использованием  
строковых команд. Разработать программу обработки символьной информации  
на языке Ассемблер и включить в программу на языке высокого уровня – С 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. По заданию из таблицы 4 варианта:

Инвертирование введенных во входной строке цифр в восьмеричной СС и преобразование заглавных русских букв в строчные, остальные символы входной строки передаются в выходную строку непосредственно.  
- 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).

Выполнение работы.

Для реализации задачи, поставленной в лабораторной работе, был написан

программный код на языке С с использованием принципа встраивания

ассемблерной части. Программа была разработана в Visual Studio 2019.

После объявления всех необходимых библиотек си ,мы обьявляем 2 глобальных массива типа char.

char input\_string[81];

char output\_string[81];

Так как в задание сказано, что длина не превышает 80 символов, а последующая функция fgets считывает еще и символ переноса строки, то мы объявляем массив из 81 элемента.

Дальше в int main() мы проводим пункт 1 — инициализацию.

setlocale(LC\_ALL, "Russian.1251")

FILE\* f;

char name[] = "res.txt";

f=fopen(name, "w");

SetConsoleCP(1251);

printf("Студент-Басыров Владимир\nГруппа 1304\nЗадание-Инвертировать все цифры восьмизначной системы счисления, а также преобразовать все заглавные русские буквы к строчным\n");

В программе и в консоли устанавливается кодировка 1251. Это свяазно с тем, что в этой кодировки удобно представлять символы киррилицы — все символы в этой таблице расположены подряд,как строчные, так и заглавные. Также здесь создается и открывается файл для записи и выводиться текст задания.

Дальше согласно пункту 2 считывается с помощью функции fgets входная строка и обрубается символ перевода строки.

fgets(input\_string, 81, stdin);

input\_string[strlen(input\_string) - 1] = '\0';

Затем согласно пункту 3 идет ассемблерная вставка. После ключевого слова \_\_asm находится блок ассемблерного кода. Регистру ES присваиваем значение регистра DS, так как при работе со строками чтение происходит из памяти по адресу ES:ESI, а запись в ячейку с адресом ES:EDI. Далее присваем смещение input\_string ESI , а output\_string EDI.

push ds

pop es

mov esi, offset input\_string

mov edi, offset output\_string

Далее идет закомментированный текст программы. Этот код нам нужен в случае, если мы не знаем кодов символов (мы их знаем!!!). Действительно, так как мы знаем, что в кодировке 1251 все символы заглавных букв идут подряд и символы строчных букв идут подряд(исключение Ё и ё), мы можем даже не знать код конкретного символа, а просто для преобразования заглавных букв в строчные найти разность между кодами символов А и а, а затем просто прибавлять к нашему исходному символу эту разность.

Начиная отсюда – текст непонятный – перехожу к программме.

Для инвертирование цифр, мы можем получить такую формулу: итоговый\_символ=код\_символа\_0+код\_символа\_7-исходный\_символ=2\*код\_символа\_0+7-исходный\_символ. Тоесть записав выражение в регистр 2\*код\_симола\_0+7 мы можем впоследствии делать исходный символ отрицательным и прибавлять к этому символу наше выражение.

; mov bl, '0'

; add bl, bl

;add bl, 7

; mov cl, 'а'

; sub cl, 'А'

Однако ,так как мы знаем коды символов, то впоследствии мы будем прибавлять уже конкретный результат тех операций, о которых было сказано выше.

Далее идет наш основный цикл line. Сначала с помощью команды lodsb в регистр al записывается нащ исходный символ. Идет 2 последовательных сравнения, которые проверяют, находится ли символ в прмоежутке от 0 до 7. Стоит отметить, что так как коды символов априори неотрицательны, то можно использовать операторы беззнакового сравнения(ja,jb).Стоит отметить, что прыдок к метке F3 есть переход к следующей проверке( является ли смвол заглавной буквой киррилицы). Как было сказано выше, используя полученную ранее формулу зная , что симвод\_0=48 в таблице, получаем, что 2\*48+7=103, а исходное выражение 103-исходный\_символ.

line :

lodsb

cmp al, '0'

jae F1

jmp F3

F1 :

cmp al, '8'

jb F2

jmp F3

F2 :

neg al

add al, 103;add al,bl

Далее идет метка F3 , в которой будет проверять , является ли символ большой буквой киррилицы. Для начала сравним, является ли символ буквой Ё, если нет, зная, что код\_символа Ё=168,а код\_символа\_ё=184, то можно к исходному символу прибавить 16.Если это символ не Ё , будет осуществлен переход к метке F4.После сравнения с кодом сивола А и Я , зная код символа А=192 и а=224, посчитав их разность получим 32, что и прибавим к коду исходного символа.Затем переход будет перейден к финальной метке.

F3 :

cmp al,'Ё'

jne F4

add al,16

jmp Final

F4:

cmp al, 'А'

jae T1

jmp Final

T1 :

cmp al, 'Я'

jbe T2

jmp Final

T2 :

add al,32 ; add al, cl

jmp Final

В метке final из регистра al записывается получившееся буква с помощью команды stosb. Далее идет проверка, является ли символ нулевым(\0). Если да, происходит переход к окончанию нашей ассемблерной вставки (lineEnd). Иначе происходит переход к начальной метке line.

Final :

stosb

mov dl, '\0'

cmp dl, [esi]

je lineEnd

jmp line

lineEnd :};

Затем в соответсвии с пунктом 4, в консоль выводится итоговая строка , а также она записывается в файл, после чего файл закрывается.

printf("%s", output\_string);

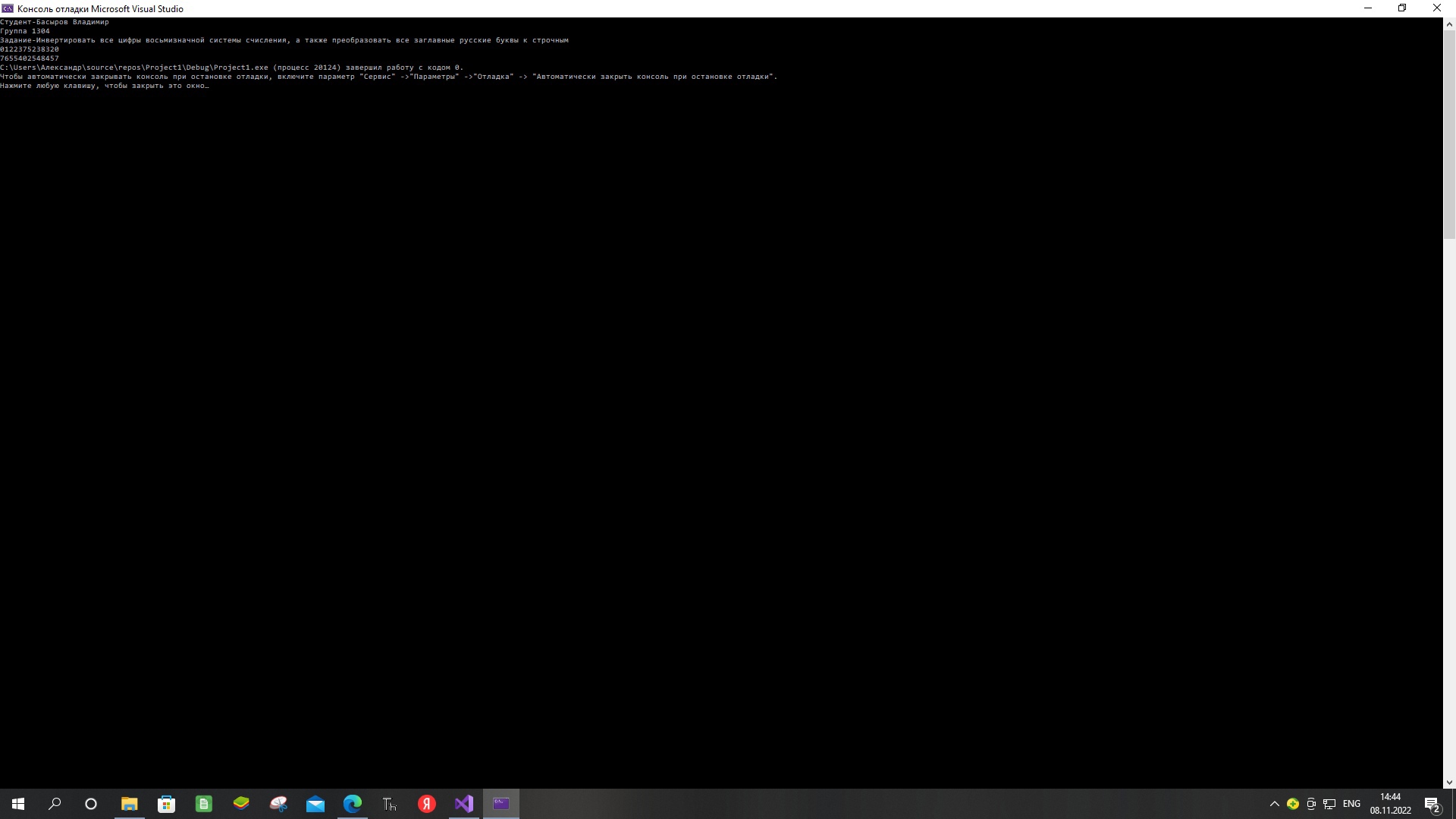
fprintf(f,"%s", output\_string);

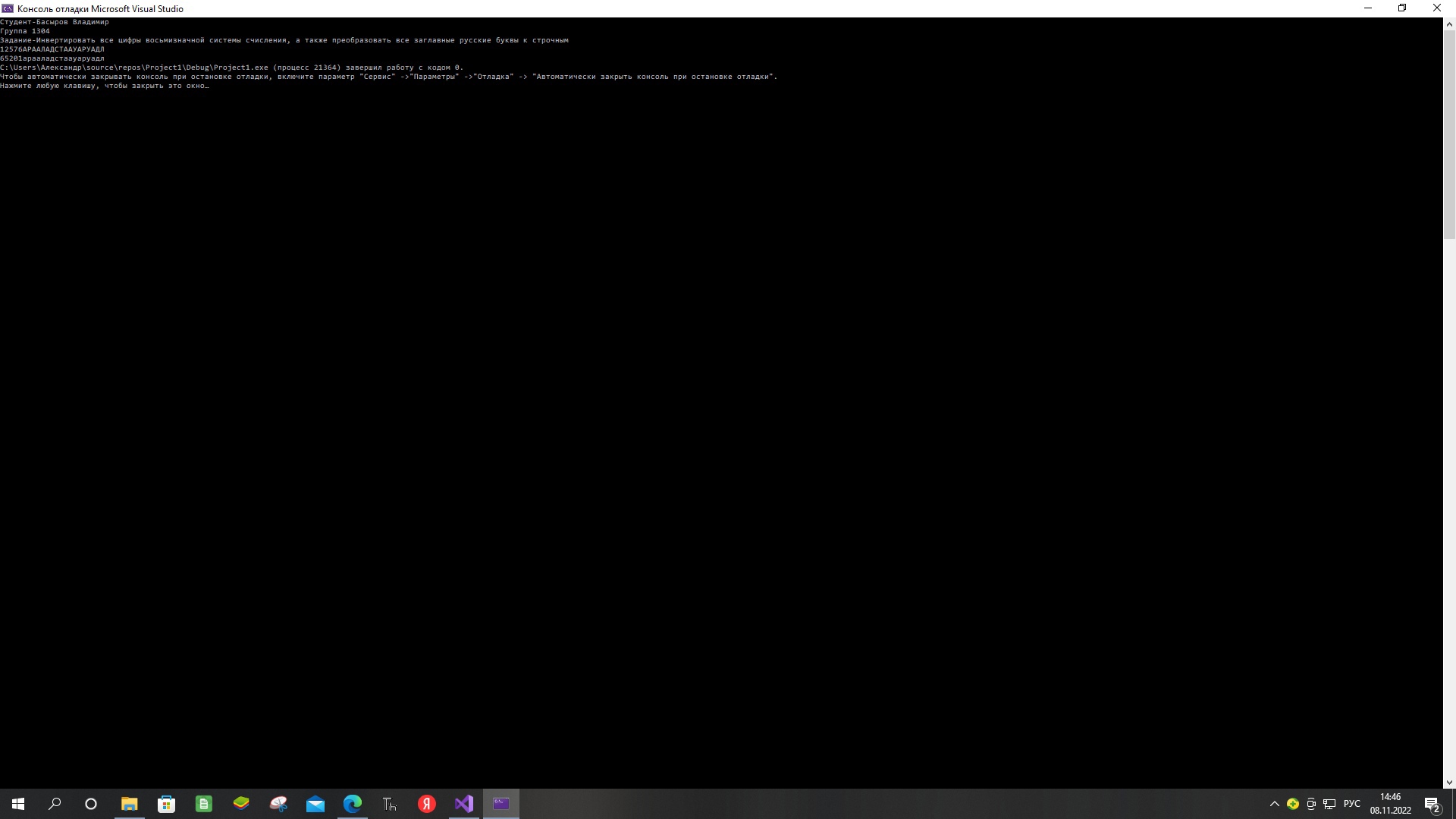
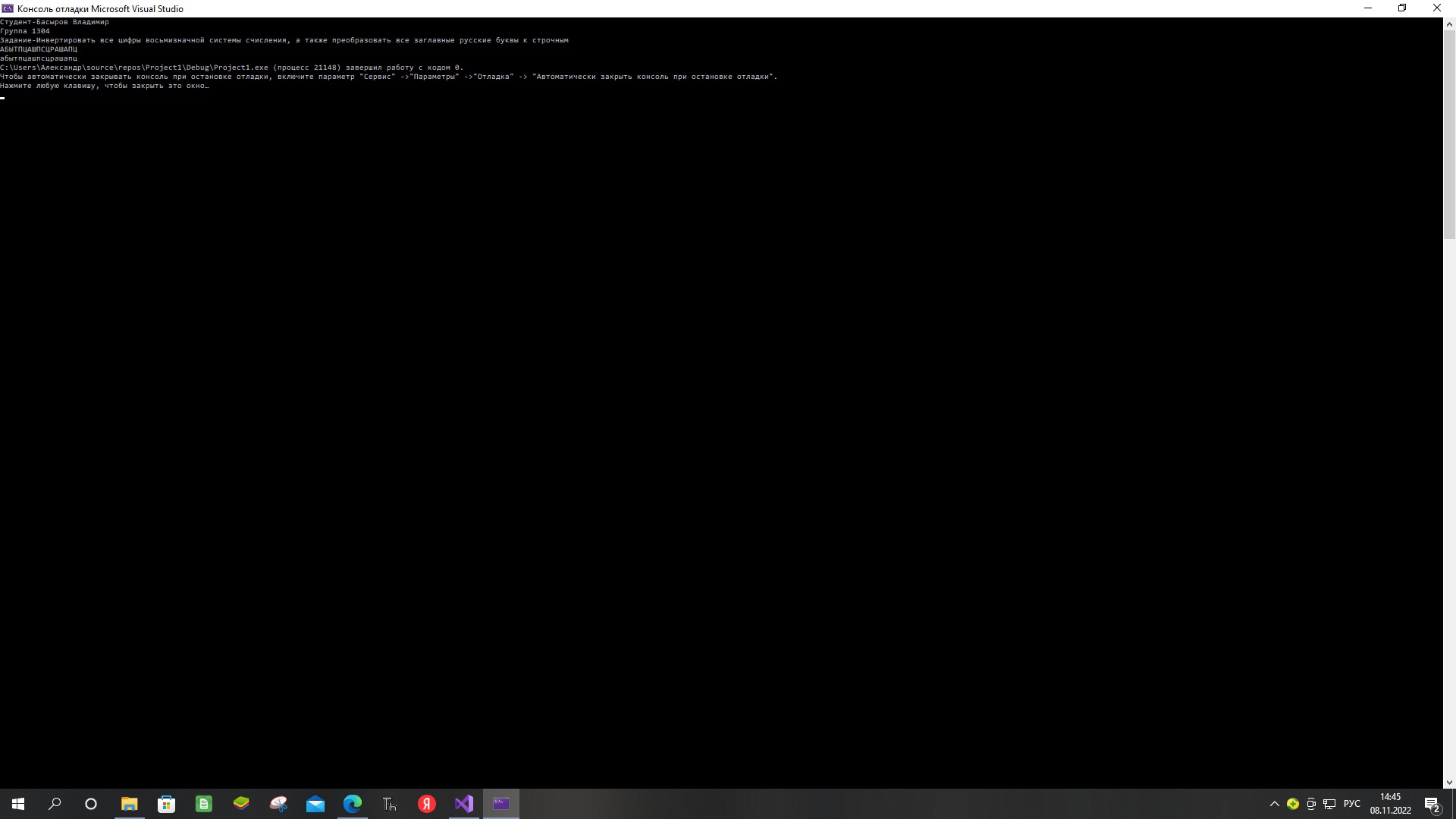
fclose(f);

return 0;}

**Тестирование:**

Тест 1: Проверка правильности инвертирования цифр восьмизначной системы счисления, а также проверка, что цифра 8 инвертирована не будет.

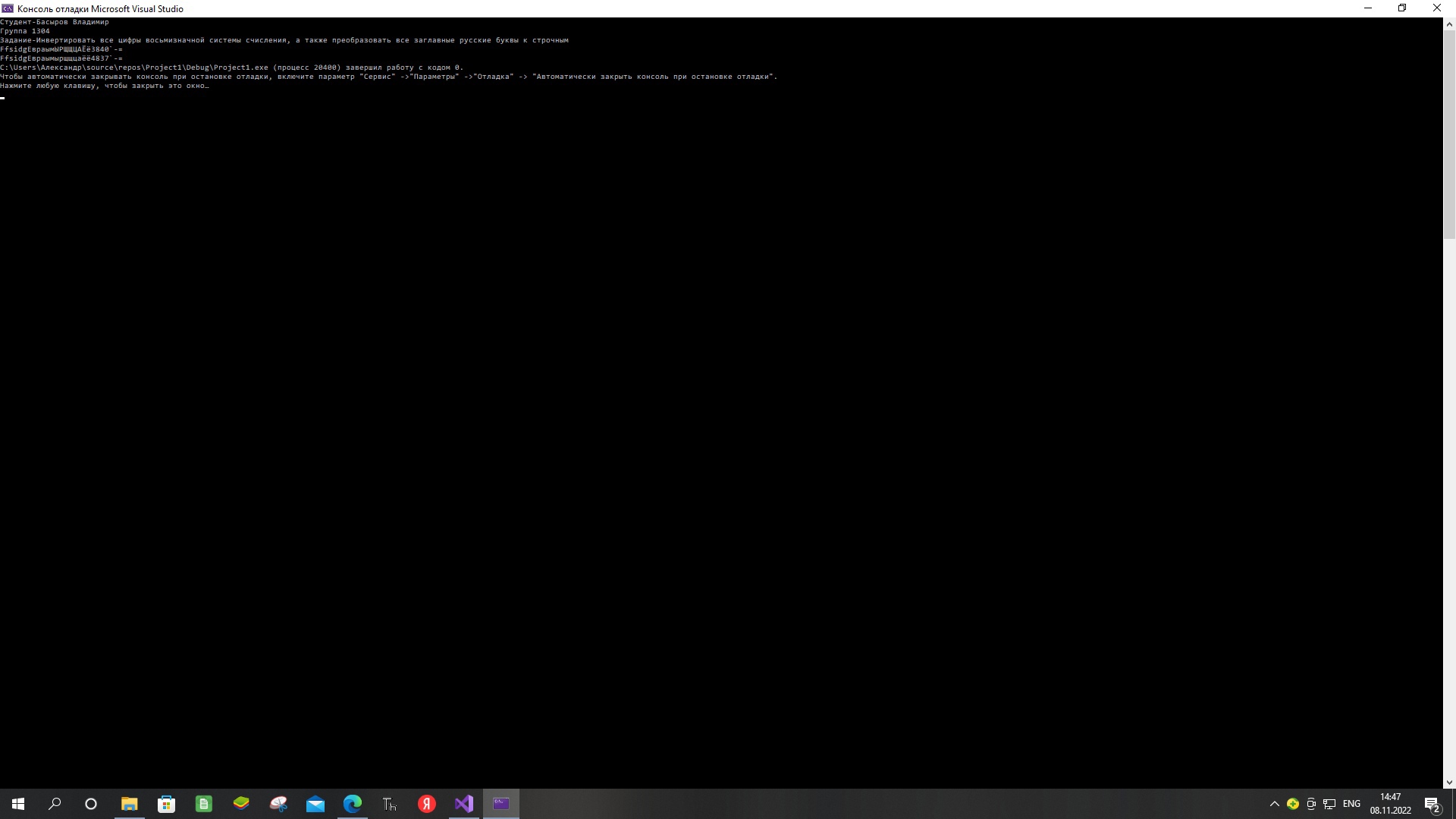
Здесь и далее все скриншоты – плохие, проверить не возможно.



Тест 2: Проверка, что все заглавные буквы киррилицы будут преобразованы к строчным буквам.

Тест 3: Проверка, что комбинация заглавных букв и цифр восьмеричной системы счиления будут выведены корректно.

Тест 4: Проверка, что никакие посторонние символы(символы латиницы и другие) останутся исходными, а также проверка , что все цифры , буквы киррилицы и буква Ё будет преобразована корректна.



**Вывод:**Были изучены основы ассемблерной вставки в язык программирования си.

Исходный код программы.

main.c

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

#include <locale.h>

#include <stdlib.h>

#include "windows.h"

char input\_string[81];

char output\_string[81];

int main() {

setlocale(LC\_ALL, "Russian.1251");

FILE\* f;

char name[] = "res.txt";

f=fopen(name, "w");

SetConsoleCP(1251);

printf("Студент-Басыров Владимир\nГруппа 1304\nЗадание-Инвертировать все цифры восьмизначной системы счисления, а также преобразовать все заглавные русские буквы к строчным\n");

fgets(input\_string, 81, stdin);

input\_string[strlen(input\_string) - 1] = '\0';

\_\_asm { ; все команды ассемблера надо комментировать

push ds

pop es

mov esi, offset input\_string

mov edi, offset output\_string

; mov bl, '0' –что означают закомментир-е команды ?

; add bl, bl

;add bl, 7

; mov cl, 'а'

; sub cl, 'А'

line :

lodsb

cmp al, '0'

jae F1

jmp F3; зачем переходить на проверку букв. если код символа < 30 ?

F1 :

cmp al, '8'

jb F2

jmp F3

F2 :

neg al

add al, 103;add al,bl - что значит эта строка ?

jmp Final

F3 :

cmp al,'Ё'

jne F4

add al,16

jmp FInal

F4:

cmp al, 'А'

jae T1

jmp Final

T1 :

cmp al, 'Я'

jbe T2

jmp Final

T2 :

add al,32 ; add al, cl - что значит эта строка ?

jmp Final - зачем эта команда?

Final :

stosb

mov dl, '\0'

cmp dl, [esi]

je lineEnd

jmp line

lineEnd :

};

printf("%s", output\_string);

fprintf(f,"%s", output\_string);

fclose(f);

return 0;

}