# 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 № 7**

**по дисциплине «Организация ЭВМ и систем»**

**Тема: Преобразование целых чисел. Использование процедур в Ассемблере.**

Студент гр. 1303 Бутыло Е.А.

Преподаватель Ефремов М.А.

Санкт-Петербург 2022

# Цель работы.

Разработать программу, реализующую преобразование чисел из одной системы счисления в другую.

# Задание.

Разработать на языке Ассемблер IntelX86 две процедуры: одна — прямого и другая — обратного преобразования целого числа, заданного в регистре AX.

Представление числа: с учетом знака. Система счисления для изображения числа — десятичная. Связь данных между основной программой и подпрограммами осуществляется через сегмент стека.

Строка должна храниться в памяти, а также выводиться на экран для индикации.

# Выполнение работы.

С помощью стека передаём значение регистра AX, внесённое в процедуре Main, в процедуру CAST\_TO\_DEC. Данная процедура реализует проверку числа на знак в случае, если число окажется отрицательным, то в заранее отведённую строку DEC\_STR добавится символ минуса, иначе выполняется деление числа на 10, соответствующее сохранение остатков в стеке. После чего извлекаются остатки и записываются в строку DEC\_STR, также добавляется символ конца строки и выход из процедуры. Далее строка DEC\_STR печатается на экран.

Далее DEC\_STR передаем в процедуру GO\_TO\_REG с помощью стека. В процедуре GO\_TO\_REG сперва идет проверка на знак минус, если он есть, то в DI помещаем соответствующий флаг. Дальше в BX передаем систему счисления, после чего идет проверка на конец строки. После чего символ преобразуем в 16-ю систему и записываем в регистр AX.

Если в DI был флаг 1, то берется противоположное со знаком число в AX. После окончание процедуры переход в процедуру CAST\_TO\_HEX. В процедуре CAST\_TO\_HEX в CL заносится количество бит для считывания из строки, то есть вывод по 4 бита. Далее в AL заносится цифра в соответствие с 16-ой системой, после чего в AL получаем символ цифры и происходит проверка на цифру, если это цифра, то переходим к Digit, где записываем в строку и если CL еще больше или равно 0, то повторяем действия. После окончание процедуры выводим строку HEX\_STR на экран.

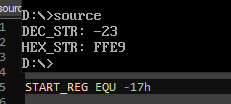
**Проверка работы программы.**

Рис. 1 – корректное выполнение программы.

# Выводы

Разработана программа преобразования число из 16-ой системы в 10-ую систему в строчном представлении, а также из строкового вида 10-ой системы в строковое представление 16-ой системы.

# ПРИЛОЖЕНИЕ A КОД ПРОГРАММ

Название файла: source.asm

AStack SEGMENT STACK

DB 1024 DUP(?)

AStack ENDS

START\_REG EQU -17h

DATA SEGMENT

HEX\_NAME DB 'HEX\_STR: ', '$'

DEC\_NAME DB 'DEC\_STR: ', '$'

DEC\_STR DB ' ','$'

HEX\_STR DB ' ', '$'

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA, SS:AStack

WriteMsg PROC NEAR

mov AH, 9

int 21h

ret

WriteMsg ENDP

CAST\_TO\_DEC proc

pop cx

pop di

pop ax

push bx

push cx

push dx

mov bx, 10

xor cx, cx

or ax, ax

jns div1

neg ax

push ax

mov dl, '-'

mov [di], dl

inc di

pop ax

div1:

xor dx, dx

div bx

push dx

inc cx

or ax, ax

jnz div1

sto:

pop dx

add dl, '0'

mov [di], dl

inc di

loop sto

mov dl, '$'

mov [di], dl

inc di

pop dx

pop cx

pop bx

push cx

ret

CAST\_TO\_DEC endp

GO\_TO\_REG proc

pop cx

pop dx

push cx

push si

push di

mov si, OFFSET DEC\_STR

cmp byte ptr [si], "-"

jnz l1

mov di, 1

inc si

l1:

xor ax, ax

mov bx, 10

l2:

mov cl, [si]

cmp cl, '$'

jz endin

sub cl, '0'

mul bx

add ax, cx

inc si

jmp l2

endin:

cmp di, 1

jnz l3

neg ax

l3:

pop di

pop si

pop cx

push ax

push cx

ret

GO\_TO\_REG endp

CAST\_TO\_HEX proc

pop cx

pop di

pop ax

push cx

push dx

mov cl, ((16-1)/4)\*4

xchg dx, ax

Repeat:

mov ax, dx

shr ax, cl

and al, 0Fh

add al, '0'

cmp al, '9'

jbe Digit

add al, 'A'-('9'+1)

Digit:

push dx

mov dl , al

mov [di], dl

inc di

pop dx

sub cl, 4

jnc Repeat

mov dl, '$'

mov [di], dl

inc di

pop dx

ret

CAST\_TO\_HEX endp

Main PROC FAR

push ds

sub ax,ax

push ax

mov ax, DATA

mov ds, ax

mov ax, START\_REG

push ax

mov di, OFFSET DEC\_STR

push di

call CAST\_TO\_DEC

mov dx, OFFSET DEC\_NAME

call WriteMsg

mov dx, OFFSET DEC\_STR

call WriteMsg

push dx

push ax

mov ah,2

mov dl, 10

int 21h

pop ax

pop dx

mov dx, OFFSET DEC\_STR

push dx

call GO\_TO\_REG

mov di, OFFSET HEX\_STR

push di

call CAST\_TO\_HEX

mov dx, OFFSET HEX\_NAME

call WriteMsg

mov dx, OFFSET HEX\_STR

call WriteMsg

ret

Main ENDP

CODE ENDS

END Main