МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ,

СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное образовательное бюджетное учреждение

высшего профессионального образования

«Санкт-Петербургский государственный университет телекоммуникаций

им. проф. М.А. Бонч-Бруевича»

Факультет Информационных технологий и программной инженерии

Кафедра Программной инженерии и вычислительной техники

ЛАБОРАТОРНАЯ РАБОТА №2

по дисциплине:

**«Машинно-зависимые языки программирования»**

тема: Вычисление целочисленных арифметических операций

Выполнил студент(ка):

Мананков Антон Павлович

*(Ф.И.О., № группы) (подпись)*

Дата выполнения:

«\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г.

Проверил:

Савельев Игорь Леонидович

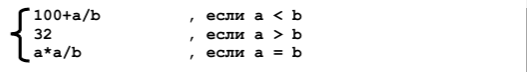
*(Ф.И.О.) (подпись)*

Санкт-Петербург

2025

**1. Задание**

Вычислить заданное условное целочисленное выражение(Рисунок 1) для данных в форматах знакового (16 бит) и беззнакового целого (16 бит), используя команды сравнения, условного и безусловного переходов.

Рисунок 1 - Условное арифметическое выражение

**2. Текст программы**

**2.1. Модуль main.c**

#include <stdio.h>

#include <stdint.h>

int16\_t a\_sign, b\_sign;

uint16\_t a\_unsign, b\_unsign;

int16\_t result\_sign;

uint16\_t result\_unsign;

extern void asm\_sign();

extern void asm\_unsign();

int calc\_sign(int16\_t a, int16\_t b) {

if (a == b) {

return a\*a/2;

} else if (a > b) {

return 32;

} else {

return (100+a/b);

}

}

int calc\_unsign(uint16\_t a, uint16\_t b) {

if (a == b) {

return a\*a/2;

} else if (a > b) {

return 32;

} else {

return (100+a/b);

}

}

int main() {

printf("Input signed (-32768 >x< 32767) a and b: \n");

printf("a-> ");

scanf("%hd", &a\_sign);

printf("b-> ");

scanf("%hd", &b\_sign);

result\_sign = calc\_sign(a\_sign, b\_sign);

printf("Result in C: %d | ", result\_sign);

asm\_sign();

printf("Result in ASM: %d\n\n", result\_sign);

printf("Input unsigned (0 >x< 65535) a and b: \n");

printf("a-> ");

scanf("%hd", &a\_unsign);

printf("b-> ");

scanf("%hd", &b\_unsign);

result\_unsign = calc\_unsign(a\_unsign, b\_unsign);

printf("Reslt in C: %u | ", result\_unsign);

asm\_unsign();

printf("Result in ASM: %u\n\n", result\_unsign);

return 0;

}

**2.2. Модуль asm16unsignbit.asm**

section .text

extern a\_unsign, b\_unsign

extern result\_unsign

global asm\_unsign

asm\_unsign:

MOV AX, [a\_unsign]

MOV BX, [b\_unsign]

CMP AX, BX

ja above

je equal

jb below

below:

; 100 + a/b

MOV AX, [a\_unsign]

MOV CX, 100

DIV BX

ADD AX, CX

jmp end

above:

; 32

MOV AX, 32

jmp end

equal:

; a\*a/b

MOV AX, [a\_unsign]

MUL AX

DIV BX

jmp end

end:

MOV [result\_unsign], AX

ret

**2.3. Модуль asm16signbit.asm**

section .text

extern a\_sign, b\_sign

extern result\_sign

global asm\_sign

asm\_sign:

MOV AX, [a\_sign]

MOV BX, [b\_sign]

CMP AX, BX

jg bigger

je equal

jl smaller

smaller:

; 100 + a/b

MOV AX, [a\_sign]

MOV CX, 100

CWD

IDIV BX

ADD AX, CX

jmp end

bigger:

; 32

MOV AX, 32

jmp end

equal:

; a\*a/b

MOV AX, [a\_sign]

IMUL AX

CWD

IDIV BX

MOV BX, AX

jmp end

end:

MOV [result\_sign], AX

ret

**2.3. Модуль makefile**

all: lr2.exe

lr2.exe: main.c asm16signbit.o asm16unsignbit.o

gcc -m64 -o lr2.exe main.c asm16signbit.o asm16unsignbit.o -no-pie -z noexecstack -Wno-format

asm16signbit.o: asm16signbit.asm

nasm -f elf64 -F dwarf -g asm16signbit.asm

asm16unsignbit.o: asm16unsignbit.asm

nasm -f elf64 -F dwarf -g asm16unsignbit.asm

clean:

find . -name "\*.exe" -delete

find . -name "\*.o" -delete

**3. Сборка проекта**

sudo apt update && sudo apt install -y nasm make gcc git

git clone <https://github.com/IglFox/MZYAP>

cd LabWork\_2 && make

**4. Выполнение программы**

**4.1. Запуск программ**

./lr2.exe

**4.2. Входные данные**

Для 8 бит:

Enter the values for 8-bit (signed char from -128 to 127):

a = 1

b = 1

c = 1

Для 16 бит:

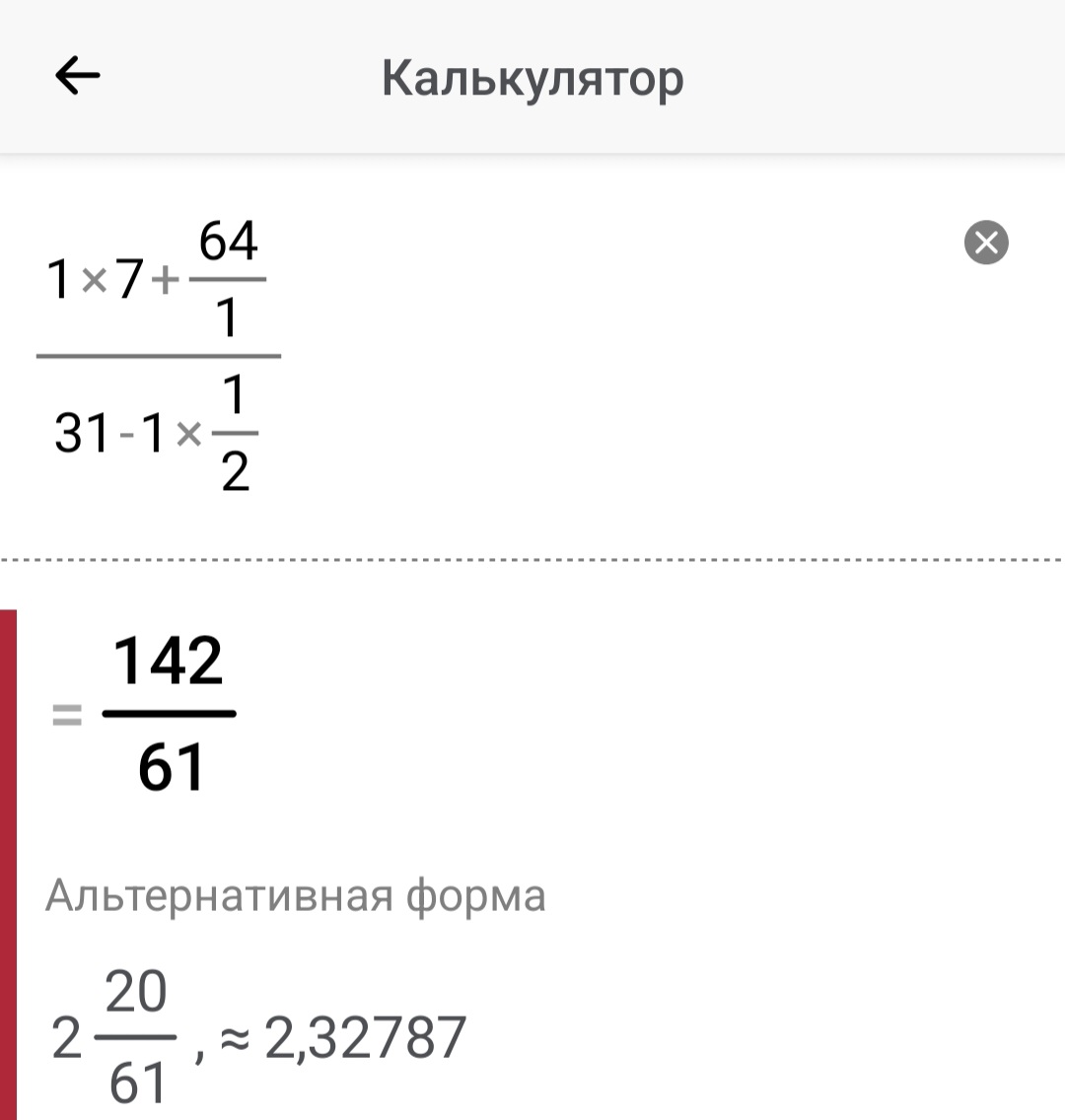
Enter the values for 16-bit (unsigned short from 0 to 65535):

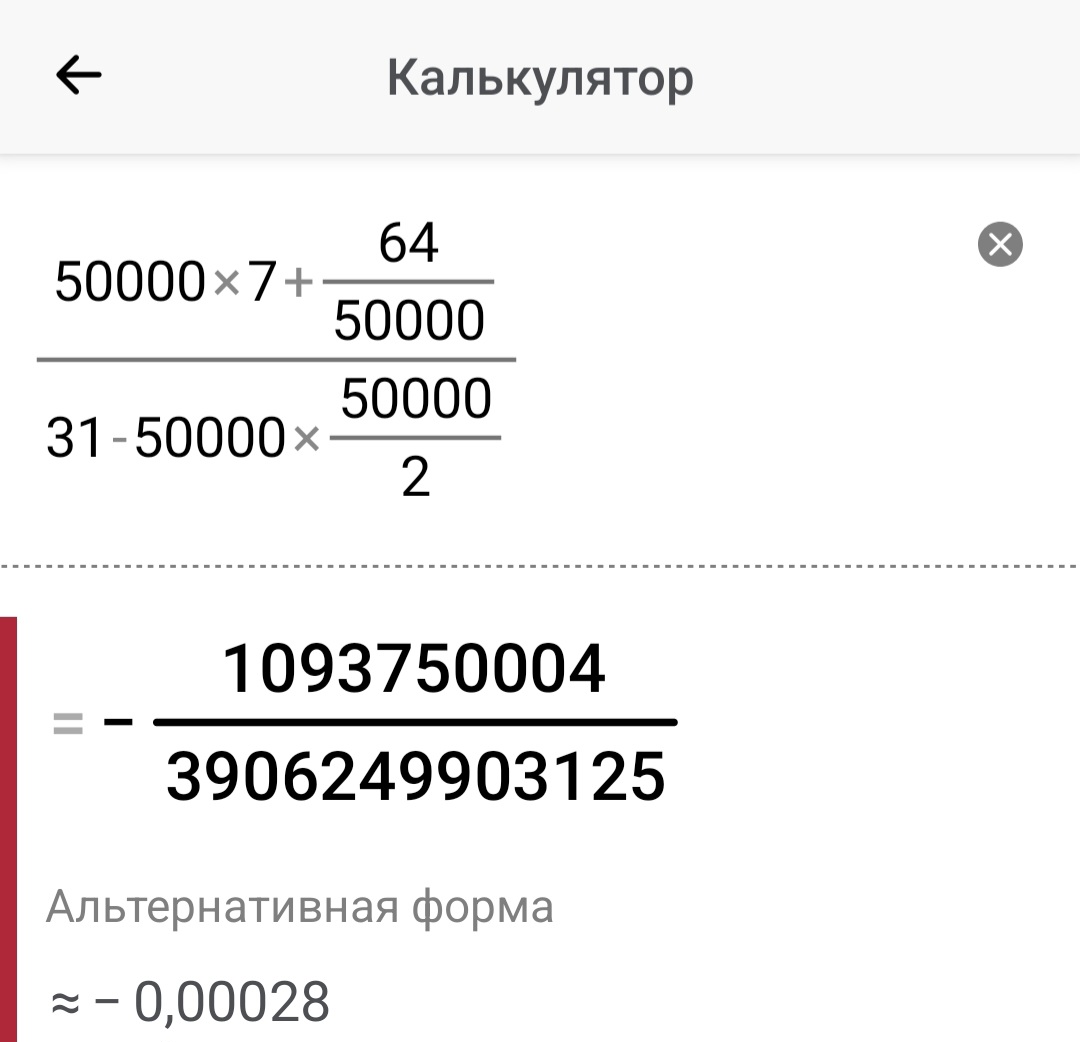
a = 65535

b = 65535

c = 65535

**4.3. Ожидаемый результат выполнения**

Рисунок 2 - Ожидаемый результат 8бит

Рисунок 3 - Ожидаемый результат 16бит

**4.4. Результат выполнения**

Для 8 бит:

Calculation on C 8-bit (signed char)

Numerator is 71

Denominator is 31

Result is 2

Calculation on ASM 8-bit (signed char)

num=71

den=31

res=2

Для 16 бит:

Calculation on C 16-bit (unsigned short)

Numerator is 458745

Denominator is -2147418081

Result is 0

Calculation on ASM 16-bit (unsigned int)

num= 458745

den= -2147418081

res= 0

**5. Вывод**

Результат выполнения программы соответствует ожидаемому результату. Работа выполнена в полном объеме.