

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ "САНКТ-ПЕТЕРБУРГСКИЙ  
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М. А.  
БОНЧ-БРУЕВИЧА"

Факультет инфокоммуникационных сетей и систем

Кафедра программной инженерии и вычислительной техники

**ЛАБОРАТОРНАЯ РАБОТА №1**

«ВЫЧИСЛЕНИЕ ЦЕЛОЧИСЛЕННЫХ АРИФМЕТИЧЕСКИХ ВЫРАЖЕНИЙ»

по дисциплине «Машинно-зависимые языки программирования»

Выполнил:

студент 2 курса

дневного отделения

группы ИКПИ-23

Даненко Д. А.

Санкт-Петербург 2023

## А. Постановка задачи

Вычислить заданное целочисленное выражение для исходных данных в знаковых и беззнаковых форматах длиной 8 и 16 бит: **signed char**, **unsigned int**, используя арифметические операции **ADD**, **ADC**, **INC**, **SUB**, **SBB**, **DEC**, **NEG**, **MUL**, **IMUL**, **DIV**, **IDIV**, **CBW**, **CWD**.

$$(55 - b + 1 * a) / (-88 / c + 1)$$

Исходные значения переменных вводятся пользователем с клавиатуры. Они должны быть максимально приближены к максимально-возможным для тех типов данных, с которыми решается задача. При вводе данных рекомендуется вывести диапазон возможных значений. Размер и тип числителя, знаменателя и результата зависит от заданного выражения.

Программа на языке Си должна вывести на экран числитель, знаменатель и результат вычисления на языке Си, а также числитель, знаменатель и результат вычисления на языке ASM. Обмен данными между Си и ASM — модулем должен осуществляться через глобальные переменные, определенные в модуле Си.

## Б. Разработка алгоритма

**Все четыре формата данных будут продемонстрированы в одном проекте.**

**Входные данные** состоят из 6 чисел *A*, *B*, *C* (*signed char*, *unsigned int*).

**Выходные данные** состоят из числителя, знаменателя и результата вычисления на языке Си и ASM для каждого типа задания (*signed char*, *unsigned char*, *signed int*, *unsigned int*). Переменные числителя и результата в два раза больше размера переменной знаменателя.

**Размеры типов данных.**

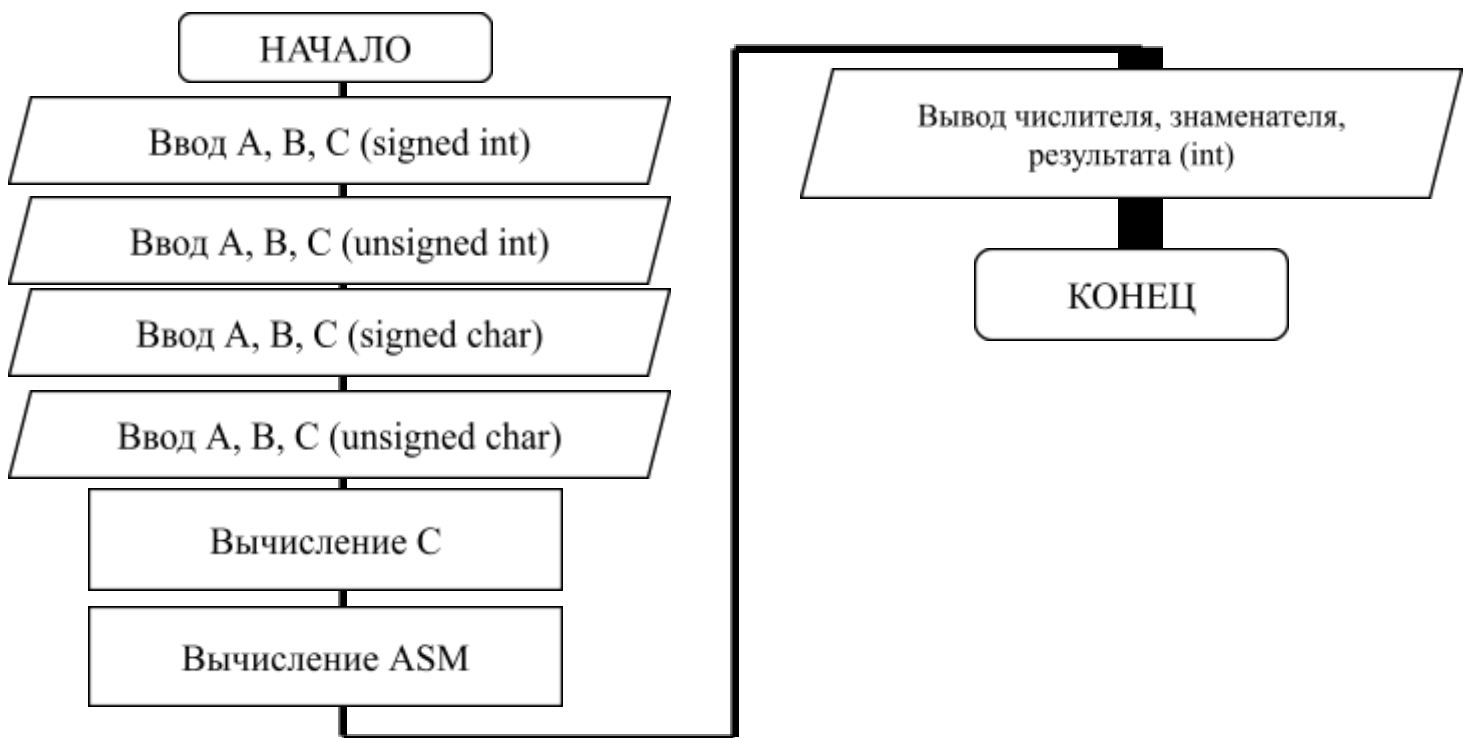
**sizeof**(signed int) = **sizeof**(unsigned int) = WORD (16 бит)

**sizeof**(signed char) = **sizeof**(unsigned char) = BYTE (8 бит)

## В. Таблица идентификаторов

N	Обозначение в задаче	Идентификатор	Назначение
1	A (signed int)	sia	Входные данные
2	B (signed int)	sib	
3	C (signed int)	sic	
4	A (unsigned int)	usia	
5	B (unsigned int)	usib	
6	C (unsigned int)	usic	
7	A (signed char)	sca	
8	B (signed char)	scb	
9	C (signed char)	scc	
10	A (unsigned char)	usca	
11	B (unsigned char)	uscb	
12	C (unsigned char)	uscc	
14	Числитель (int)	Num	Выходные данные
15	Знаменатель (int)	Den	
16	Результат (int)	Res	

## Г. Схема алгоритма



## Д. Таблица результатов

Результаты вычислений приведены ниже в таблице вычислений.

Тип	A	B	C	Числитель	Знаменатель	Результат
Signed Byte	-1	-4	-5	C: 58 ASM: 58	18 18	3 3
	127	-128	-128	C: 310 ASM: 310	1 1	310 310
	-128	127	1	C: -200 ASM: -200	-87 -87	2 2
Unsigned Word	1	2	3	C: 54 ASM: 54	-28 -28	65535 65535
	256	256	256	C: 55 ASM: 55	1 1	55 55
	9	5	1	C: 59 ASM: 59	-87 -87	0 0

```
eg@eg-hplaptop15bs1xx ~/Documents/forritun/VSCode/mdpl/lab1 [19:59:26]
> $ ./c [main ●●]
sca = -1
scb = -4
scc = -5
C result:
58 18 3
Asm result:
58 18 3

eg@eg-hplaptop15bs1xx ~/Documents/forritun/VSCode/mdpl/lab1 [19:59:33]
> $ [main ●●]
```

```
> $ make [±]
main ●●]
gcc -g -gdb -c -o c.o c.c
nasm -g -f elf64 asm.asm -l asm.lst -F dwarf
gcc -g -o c c.o asm.o -fno-pie -no-pie

eg@eg-hplaptop15bs1xx ~/Documents/forritun/VSCode/mdpl/lab1 [2
0:02:12]
> $ ./c [main ●●]
sca = 127
scb = -128
scc = -128
C result:
310 1 310
Asm result:
310 1 310

eg@eg-hplaptop15bs1xx ~/Documents/forritun/VSCode/mdpl/lab1 [2
0:02:19]
> $ [main ●●]
```

```
Asm result:
310 1 310

eg@eg-hplaptop15bs1xx ~/Documents/forritun/VSCode/mdpl/lab1 [2
0:02:19]
> $ make [main ●●]
gcc -g -gdb -c -o c.o c.c
nasm -g -f elf64 asm.asm -l asm.lst -F dwarf
gcc -g -o c c.o asm.o -fno-pie -no-pie

eg@eg-hplaptop15bs1xx ~/Documents/forritun/VSCode/mdpl/lab1 [20:03:24]
> $ ./c [main ●●]
sca = -128
scb = 127
scc = 1
C result:
-200 -87 2
Asm result:
-200 -87 2

eg@eg-hplaptop15bs1xx ~/Documents/forritun/VSCode/mdpl/lab1 [20:03:34]
> $ [main ●●]
```

<pre>&gt; \$ ./c ---Input--- usia = 1 usib = 2 usic = 3 ---C part--- Num = 54 Den = -28 Res = -1 ---ASM part--- Num = 54 Den = -28 Res = -1</pre>	<pre>&gt; \$ ./c ---Input--- usia = 256 usib = 256 usic = 256 ---C part--- Num = 55 Den = 1 Res = 55 ---ASM part--- Num = 55 Den = 1 Res = 55</pre>	<pre>&gt; \$ ./c ---Input--- usia = 9 5 1 usib = usic = ---C part--- Num = 59 Den = -87 Res = 0 ---ASM part--- Num = 59 Den = -87 Res = 0</pre>
---	---	---

## ФАЙЛ C

```
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
extern void asm_signed_int(void);
extern void asm_unsigned_int(void);
extern void asm_signed_char(void);
extern void asm_unsigned_char(void);

int32_t Num, Res;
int16_t Den;
int16_t sia, sib, sic; // signed_int
uint16_t usia, usib, usic; // unsigned_int
int8_t sca, scb, scc; // signed_char
int8_t usca, uscb, uscc; // unsigned_char

void siC(int16_t sia, int16_t sib, int16_t sic) {
    Num = 0;
    Den = 0;
    Res = 0;
    Num = 55 - sib + sia;
    Den = (-88 / sic) + 1;
    Res = (Num / Den);
    printf("%d\n", Num);
    printf("%d\n", Den);
    printf("%d\n", Res);
}

void siASM(int16_t sia, int16_t sib, int16_t sic) {
    sia = sia;
    sib = sib;
    sic = sic;
    Num = 0;
    Den = 0;
    Res = 0;
    asm_signed_int();
}

void usiC(uint16_t usia, uint16_t usib, uint16_t usic) {
    Num = 0;
    Den = 0;
    Res = 0;
    Num = 55 - usib + usia;
    Den = (-88 / usic) + 1;
    Res = (Num / Den);
    printf("---C part---\n");
    printf("Num = %d\n", Num);
    printf("Den = %d\n", Den);
```

```

    printf("Res = %d\n", Res);
}

void usiASM(uint16_t usia, uint16_t usib, uint16_t usic) {
    usia = usia;
    usib = usib;
    usic = usic;
    Num = 0;
    Den = 0;
    Res = 0;
    asm_unsigned_int();
}

void scC(int8_t sca, int8_t scb, int8_t scc) {
    Num = 0;
    Den = 0;
    Res = 0;
    Num = 55 - scb + sca;
    Den = (-88 / scc) + 1;
    Res = (Num / Den);
    printf("%d\n", Num);
    printf("%d\n", Den);
    printf("%d\n", Res);
}

void scASM(int8_t sca, int8_t scb, int8_t scc) {
    sca = sca;
    scb = scb;
    scc = scc;
    Num = 0;
    Den = 0;
    Res = 0;
    asm_signed_char();
}

int main(int argc, char *argv[]) {
    // (55-b+1*a)/(-88/c+1)
    /*
    // Signed Int ()
    printf("sia = ");
    scanf("%d", &sia);

    printf("sib = ");
    scanf("%d", &sib);

    printf("sic = ");
    scanf("%d", &sic);

    siASM(sia, sib, sic);
    //siC(sia, sib, sic);
    // -----
*/
    /*
    // Unsigned Int (0<=>65535)
    printf("---Input---\n");
    printf("usia = ");
    scanf("%u", &usia);

    printf("usib = ");
    scanf("%u", &usib);

    printf("usic = ");
    scanf("%u", &usic);

```

```

    if (usic == 0) {
        printf("error; c = 0");
        return 0;
    }

    usiC(usia, usib, usic);
    usiASM(usia, usib, usic);
    printf("---ASM part---\n");
    printf("Num = %d\n", Num);
    printf("Den = %d\n", Den);
    printf("Res = %d\n", Res);
    // -----
    */
// Signed Char (-128<->127)
printf("sca = ");
scanf("%d", &sca);

printf("scb = ");
scanf("%d", &scb);

printf("scc = ");
scanf("%d", &scc);

scC(sca, scb, scc);
scASM(sca, scb, scc);
printf("%d\n", Num);
printf("%d\n", Den);
printf("%d\n", Res);
// -----

return 0;
}

```

#### ФАЙЛ ASM

```

section .data
    extern Num , Den , Res , var
    extern sia , sib , sic
    extern usia , usib , usic
    extern sca , scb , scc
    extern usca , uscb , uscc

section .text
    global asm_signed_int
    global asm_unsigned_int
    global asm_signed_char
    global asm_unsigned_char

asm_signed_int:
    ; numerator
    mov     bx,     [sib] ; bx = b
    mov     ax,     55     ; ax = 55
    sub     ax,     bx     ; ax = 55 - b
    mov     bx,     [sia] ; bx = a
    add     ax,     bx     ; ax = 55 - b + a
    mov     [Num],   ax     ; Num = 55 - b + a

    ; denominator
    mov     ax,     -88     ; ax = -88
    cwd     ; ax:dx = -88
    mov     bx,     [sic] ; bx = c
    idiv    bx          ; ax = -88 / c

```

```

inc      ax                ; ax = -88 / c + 1
mov      [Den], ax         ; Den = 55 / c + 1

; result
mov      ax, [Num] ; ax = 55 - b + a
cwd      ; ax:dx = 55 - b + a
mov      bx, [Den] ; bx = -88 / c + 1
idiv     bx              ; ax = num / den
mov      [Res], ax       ; Res = ax = Num / Den
ret

```

asm\_unsigned\_int:

```

; cleaning
xor      eax, eax
xor      ebx, ebx
xor      ecx, ecx
xor      edx, edx

; numerator
mov      bx, [usib] ; bx = b
mov      ax, 55     ; ax = 55
sub      eax, ebx   ; ax = 55 - b
mov      bx, [usia] ; bx = a
add      eax, ebx   ; ax = 55 - b + a
mov      [Num], eax ; Num = 55 - b + a

; denominator
xor      eax, eax
mov      ax, -88    ; ax = -88
cwd      ; ax:dx = -88
mov      bx, [usic] ; bx = c
idiv     bx         ; ax = -88 / c
inc      ax          ; ax = -88 / c + 1
mov      [Den], ax  ; Den = ax = -88 / c + 1

; result
xor      eax, eax
xor      ebx, ebx
mov      ax, [Num] ; ax = 55 - b + a
mov      dx, [Num+2]
mov      bx, [Den] ; bx = -88 / c + 1
idiv     bx         ; ax = num / den
mov      [Res], ax  ; Res = ax = Num / Den
ret

```

asm\_signed\_char:

```

; cleaning
xor      eax, eax
xor      ebx, ebx
xor      ecx, ecx
xor      edx, edx

; numerator
mov      al, byte [sca] ; al = a
cbw
mov      bx, ax ; bx = a
mov      ax, 55 ; ax = 55
add      bx, ax ; ax = 55 + a
mov      al, byte [scb] ; al = b
cbw
sub      bx, ax ; ax = 55 - b + a

```



```

mov     [Num],    bx           ; Num = ax

; denominator
mov     ax,      -88          ; ax = -88
mov     bl,      [scc]        ; bx = c
idiv    bl
cbw
inc     ax                      ; ax = -88 / c + 1
mov     [Den],    ax          ; Den = ax = -88 / c + 1

; result
mov     ax,      [Num]        ; ax = 55 - b + a
cwd                      ; ax:dx = 55 - b + a
mov     bx,      [Den]        ; bx = -88 / c + 1
idiv    bx                ; ax = num / den
mov     [Res],    ax          ; Res = ax = Num / Den
ret

```

## Ж. Выводы

Тот факт, что результаты, выполненные на ASM, достаточно совпадают с результатами, выполненными на C, свидетельствует о том, что программа составлена правильно.