**ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

**Факультет инфокоммуникационных Сетей и систем (иксс)**

**кафедра программной инженерии и вычислительной техники (пи и вт)**

Дисциплина: «Машинно-зависимые языки программирования»

Лабораторная работа №1.

**Тема: «вычисление целочисленных арифметических выражений»**

**вариант №1**

Выполнил: Студент группы ИКПИ-95

Алюшин В.В.

Подпись\_\_\_\_\_\_\_\_\_\_\_\_

Принял: Коробов С.А.

Подпись\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_»\_\_\_\_\_\_\_\_ 2020

2020г

**ЦЕЛЬ РАБОТЫ:**

Вычислить заданное целочисленное выражение для исходных данных в знаковых и беззнаковых форматах длиной 8 и 16 бит: signed char, unsigned char и signed int , unsigned int , используя арифметические операции ADD, ADC, INC, SUB, SBB , DEC, NEG, MUL, IMUL, DIV, IDIV , CBW, CWD. Исходные значения переменных вводятся пользователем с клавиатуры. Они должны быть максимально приближены к максимально-возможным для тех типов данных, с которыми решается задача. Размер и тип числителя, знаменителя и результата зависит от заданного выражения.

**(3\*c+8-d)/(a-c/4)**

**Таблица Результатов**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | a | b | c | Результат ASM & C |
| int8 | 1 | 2 | 3 | 11 |
| uint8 | 3 | 4 | 5 | 7 |
| int16 | -100 | 1 | 4 | 0 |
| uint16 | 44 | 45 | 46 | 2 |

**Код:**

**Файл main.c:**

#include <stdio.h>

#include <conio.h>

extern "C"

{

signed \_\_int8 s8\_a;

unsigned \_\_int8 u8\_a;

signed \_\_int16 s16\_a;

unsigned \_\_int16 u16\_a;

signed \_\_int8 s8\_c;

unsigned \_\_int8 u8\_c;

signed \_\_int16 s16\_c;

unsigned \_\_int16 u16\_c;

signed \_\_int8 s8\_d;

unsigned \_\_int8 u8\_d;

signed \_\_int16 s16\_d;

unsigned \_\_int16 u16\_d;

}

extern "C" signed \_\_int16 calc\_s8();

extern "C" signed \_\_int16 calc\_u8();

extern "C" signed \_\_int16 calc\_s16();

extern "C" signed \_\_int16 calc\_u16();

//1 (3\*c+8-d)/(a-c/4)

signed \_\_int16 test\_calc\_s8()

{

return (3 \* s8\_c + 8 - s8\_d) / (s8\_a - s8\_c / 4);

}

signed \_\_int16 test\_calc\_u8()

{

return (3 \* u8\_c + 8 - u8\_d) / (u8\_a - u8\_c / 4);

}

signed \_\_int16 test\_calc\_s16()

{

return (3 \* s16\_c + 8 - s16\_d) / (s16\_a - s16\_c / 4);

}

signed \_\_int16 test\_calc\_u16()

{

return (3 \* u16\_c + 8 - u16\_d) / (u16\_a - u16\_c / 4);

}

void m1()

{

int buff;

printf("Enter a: ");

scanf("%i", &buff);

s8\_a = buff;

printf("Enter c: ");

scanf("%i", &buff);

s8\_c = buff;

printf("Enter d: ");

scanf("%i", &buff);

s8\_d = buff;

int r = calc\_s8();

printf("Result: %i\n", r);

r = test\_calc\_s8();

printf("Test: %i\n", r);

}

void m2()

{

int buff;

printf("Enter a: ");

scanf("%i", &buff);

u8\_a = buff;

printf("Enter c: ");

scanf("%i", &buff);

u8\_c = buff;

printf("Enter d: ");

scanf("%i", &buff);

u8\_d = buff;

int r = calc\_u8();

printf("Result: %i\n", r);

r = test\_calc\_u8();

printf("Test: %i\n", r);

}

void m3()

{

int buff;

printf("Enter a: ");

scanf("%i", &buff);

s16\_a = buff;

printf("Enter c: ");

scanf("%i", &buff);

s16\_c = buff;

printf("Enter d: ");

scanf("%i", &buff);

s16\_d = buff;

int r = calc\_s16();

printf("Result: %i\n", r);

r = test\_calc\_s16();

printf("Test: %i\n", r);

}

void m4()

{

int buff;

printf("Enter a: ");

scanf("%i", &buff);

u16\_a = buff;

printf("Enter c: ");

scanf("%i", &buff);

u16\_c = buff;

printf("Enter d: ");

scanf("%i", &buff);

u16\_d = buff;

int r = calc\_u16();

printf("Result: %i\n", r);

r = test\_calc\_u16();

printf("Test: %i\n", r);

}

int main()

{

printf("Mode 1: signed 8\n");

printf("Mode 2: unsigned 8\n");

printf("Mode 3: signed 16\n");

printf("Mode 4: unsigned 16\n");

int mode;

while (1)

{

printf("Enter mode: ");

scanf("%i", &mode);

printf("Select mode %i\n", mode);

switch (mode)

{

case (1):

{

m1();

break;

}

case (2):

{

m2();

break;

}

case (3):

{

m3();

break;

}

case (4):

{

m4();

break;

}

default:

break;

}

}

\_getch();

return 0;

}

**Файл asm (ASM):**

.data

EXTERN s8\_a: byte

EXTERN u8\_a: byte

EXTERN s16\_a: word

EXTERN u16\_a: word

EXTERN s8\_c: byte

EXTERN u8\_c: byte

EXTERN s16\_c: word

EXTERN u16\_c: word

EXTERN s8\_d: byte

EXTERN u8\_d: byte

EXTERN s16\_d: word

EXTERN u16\_d: word

.code

;var 1 (3\*c+8-d)/(a-c/4)

calc\_s8 PROC

;(a-c/4)

mov al, s8\_c

cbw ;al->ax

mov bx, 4d

cwd ;копирование старшего бита ax во все байты dx

idiv bx

mov bx, ax

mov al, s8\_a

cbw ;al->ax

sub ax, bx

mov di, ax ;результат 2 части

;(3\*c+8-d)

mov al, s8\_d

cbw ;al->ax

mov bx, 8d

sub bx, ax

mov si, bx

mov al, s8\_c

cbw ;al->ax

mov bx, ax

mov ax, 3d

xor dx, dx

imul bx

add ax, si

cwd ;копирование старшего бита ax во все байты dx

idiv di

ret

calc\_s8 endp

calc\_u8 PROC

;(a-c/4)

xor ax, ax

mov al, u8\_c

mov bx, 4d

xor dx, dx

div bx

xor bx, bx

mov bl, u8\_a

sub bx, ax

mov di, bx ;результат 2 части

;(3\*c+8-d)

mov ax, 8d

xor bx, bx

mov bl, u8\_d

sub ax, bx

mov si, ax

mov ax, 3d

xor bx, bx

mov bl, u8\_c

xor dx, dx

mul bx

add ax, si

cwd ;копирование старшего бита ax во все байты dx

idiv di

ret

ret

calc\_u8 endp

calc\_s16 PROC

;(a-c/4)

mov ax, s16\_c

mov bx, 4d

cwd ;копирование старшего бита ax во все байты dx

idiv bx

mov bx, s16\_a

sub bx, ax

mov di, bx

;(3\*c+8-d)

mov ax, 8d

mov bx, s16\_d

sub ax, bx

mov si, ax

mov ax, 3

mov bx, s16\_c

xor dx, dx

imul bx

add ax, si

cwd ;копирование старшего бита ax во все байты dx

idiv di

ret

calc\_s16 endp

calc\_u16 PROC

;(a-c/4)

mov ax, u16\_c

mov bx, 4d

xor dx, dx

div bx

mov bx, u16\_a

sub bx, ax

mov di, bx

;(3\*c+8-d)

mov ax, 8d

mov bx, u16\_d

sub ax, bx

mov si, ax

mov ax, 3

mov bx, u16\_c

xor dx, dx

mul bx

add ax, si

cwd ;копирование старшего бита ax во все байты dx

idiv di

ret

calc\_u16 endp

end