**ЛАБОРАТОРНА РОБОТА № 5**

**ОБЧИСЛЕННЯ СКЛАДНИХ ЦІЛОЧИСЛЕНИХ**

**ВИРАЗІВ НА МОВІ ASSEMBLER**

***Мета:*** ознайомитися з основними командами мови Assembler для обчислення складних цілочисельних виразів; набути практичних навичок в написанні програм для обчислення складних цілочисельних виразів на мові Assembler.

**Хід роботи:**

**Завдання 1**: Написати програму для обчислення заданого цілочисленого виразу для початкових даних в знаковому форматі довжиною 8 біт, використовуючи арифметичні операції ADD, ADC, INC, SUB, SBB, DEC, NEG, IMUL, IDIV, CBW, CWD. Провести тестові перевірки, відмітити нормальні та аномальні результати. Виконати покрокове виконання асемблерного коду та навести значення регістрів при їх виконанні.

**24.**  (4\*a/b+1-e)/(c\*b-18+a-f)

Лістинг програми:

#include "pch.h"

#include <iostream>

#include <cstdlib>

int main()

{

signed char a, b, c, e, f, k, res\_c, res\_asm;

printf("\n(4 \* a / b + 1 - e) / (c \* b - 18 + a - f)\n");

printf("Enter the values of the range [-128...127]\n");

printf("a = "); scanf\_s("%d", &a);

printf("b = "); scanf\_s("%d", &b);

printf("c = "); scanf\_s("%d", &c);

printf("e = "); scanf\_s("%d", &e);

printf("f = "); scanf\_s("%d", &f);

res\_c = (4 \* a / b + 1 - e) / (c \* b - 18 + a - f);

printf("Result C = %d\n", res\_c);

\_\_asm

{ mov al, 4; // <al> = 4

mov bl, a; // <bl> = a

imul bl; // <ax> = 4 \* a

mov cl, b; // <cl> = b

idiv cl; // <al> = 4 \* a / b

inc al; // <al> = 4 \* a / b + 1

sub al, e; // <al> = 4 \* a / b + 1 - e

mov k, al; // k = 4 \* a / b + 1 - e

mov al, c; // <al> = c

imul b; // <ax> = c \* b

mov cl, 1; // <cl> = 1

idiv cl; // <al>

sub al, 18; // <al> = c \* b - 18

add al, a; // <al> = c \* b - 18 + a

sub al, f; // <al> = c \* b - 18 + a - f

mov bl, al; // <bl> = <al>

mov al, k; // <al> = k

cbw; // <ax>

idiv bl; // <ax> = (4 \* a / b + 1 - e) / (c \* b - 18 + a - f)

idiv cl; // <al>

mov res\_asm, al; // res = <al>

}

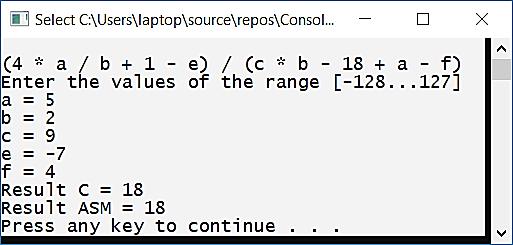
printf("Result ASM = %d\n", res\_asm);

system("pause");

return 0;

}

Результат виконання програми:



**Значення регістрів при покроковому виконанні**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Крок** | **Команда** | **Значення регістра** | | | | **EFLAGS/FLAGS (CF, OF)** |
| **al** | **ax** | **bl** | **cl** |
| **1** | mov al, 4 | 4 | н/в | н/в | н/в | **-** |
| **2** | mov bl, a | н/в | н/в | 5 | н/в | **-** |
| **3** | imul bl | 20 | н/в | н/в | н/в | **-** |
| **4** | mov cl, b | н/в | н/в | н/в | 2 | **-** |
| **5** | idiv cl | 10 | н/в | н/в | н/в | **-** |
| **6** | inc al | 11 | н/в | н/в | н/в | **-** |
| **7** | sub al, e | 18 | н/в | н/в | н/в | **-** |
| **8** | mov k, al | н/в | н/в | н/в | н/в | **-** |
| **9** | mov al, c | 9 | н/в | н/в | н/в | **-** |
| **10** | imul b | 18 | н/в | н/в | н/в | **-** |
| **11** | mov cl, 1 | н/в | н/в | н/в | 1 | **-** |
| **12** | idiv cl | 18 | н/в | н/в | н/в | **-** |
| **13** | sub al, 18 | 0 | н/в | н/в | н/в | **-** |
| **14** | add al, a | 5 | н/в | н/в | н/в | **-** |
| **15** | sub al, f | 1 | н/в | н/в | н/в | **-** |
| **16** | mov bl, al | н/в | н/в | 1 | н/в | **-** |
| **17** | mov al, k | 18 | н/в | н/в | н/в | **-** |
| **18** | cbw | 18 | н/в | н/в | н/в | **-** |
| **19** | idiv bl | 18 | н/в | н/в | н/в | **-** |
| **20** | idiv cl | 18 | н/в | н/в | н/в | **-** |
| **21** | mov res\_asm, al | 18 | н/в | н/в | н/в | **-** |

**Завдання 2**: Написати програму для обчислення заданого цілочисленого виразу для початкових даних в знаковому форматі довжиною 16 біт, використовуючи арифметичні операції ADD, ADC, INC, SUB, SBB, DEC, NEG, IMUL, IDIV, CBW, CWD. Провести тестові перевірки, відмітити нормальні та аномальні результати. Виконати покрокове виконання асемблерного коду та навести значення регістрів при їх виконанні.

Лістинг програми:

#include "pch.h"

#include <iostream>

#include <cstdlib>

int main()

{

short int a, b, c, e, f, k, res\_c, res\_asm;

printf("\n(4 \* a / b + 1 - e) / (c \* b - 18 + a - f)\n");

printf("Enter the values of the range [-32768...32767]\n");

printf("a = "); scanf\_s("%hi", &a);

printf("b = "); scanf\_s("%hi", &b);

printf("c = "); scanf\_s("%hi", &c);

printf("e = "); scanf\_s("%hi", &e);

printf("f = "); scanf\_s("%hi", &f);

res\_c = (4 \* a / b + 1 - e) / (c \* b - 18 + a - f);

printf("Result C = %hi\n", res\_c);

\_\_asm

{

mov ax, 4; // <ax> = 4

mov bx, a; // <bx> = a

imul bx; // <eax> = 4 \* a

mov cx, b; // <cx> = b

idiv cx; // <ax> = 4 \* a / b

inc ax; // <ax> = 4 \* a / b + 1

sub ax, e; // <ax> = 4 \* a / b + 1 - e

mov k, ax; // k = 4 \* a / b + 1 - e

mov ax, c; // <ax> = c

imul b; // <eax> = c \* b

mov cx, 1; // <cx> = 1

idiv cx; // <ax>

sub ax, 18; // <ax> = c \* b - 18

add ax, a; // <ax> = c \* b - 18 + a

sub ax, f; // <ax> = c \* b - 18 + a - f

mov bx, ax; // <bx> = <ax>

mov ax, k; // <ax> =k

cwde; // <eax>

idiv bx; // <eax> = (4 \* a / b + 1 - e) / (c \* b - 18 + a - f)

idiv cx; // <ax>

mov res\_asm, ax; // res = <ax>

}

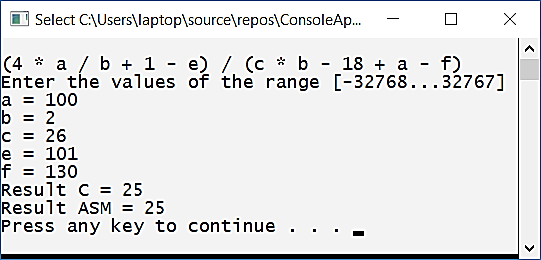
printf("Result ASM = %hi\n", res\_asm);

system("pause");

return 0;

}

Результат виконання програми:



**Значення регістрів при покроковому виконанні**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Крок** | **Команда** | **Значення регістра** | | | | **EFLAGS/FLAGS (CF, OF)** |
| **ax** | **eax** | **bx** | **cx** | **-** |
| **1** | mov ax, 4 | 4 | н/в | н/в | н/в | **-** |
| **2** | mov bx, a | н/в | н/в | 100 | н/в | **-** |
| **3** | imul bx | 400 | н/в | н/в | н/в | **-** |
| **4** | mov cx, b | н/в | н/в | н/в | 2 | **-** |
| **5** | idiv cx | 200 | н/в | н/в | н/в | **-** |
| **6** | inc ax | 201 | н/в | н/в | н/в | **-** |
| **7** | sub ax, e | 100 | н/в | н/в | н/в | **-** |
| **8** | mov k, ax | н/в | н/в | н/в | н/в | **-** |
| **9** | mov ax, c | 26 | н/в | н/в | н/в | **-** |
| **10** | imul b | 52 | н/в | н/в | н/в | **-** |
| **11** | mov cx, 1 | н/в | н/в | н/в | 1 | **-** |
| **12** | idiv cx | 52 | н/в | н/в | н/в | **-** |
| **13** | sub ax, 18 | 34 | н/в | н/в | н/в | **-** |
| **14** | add ax, a | 134 | н/в | н/в | н/в | **-** |
| **15** | sub ax, f | 4 | н/в | н/в | н/в | **-** |
| **16** | mov bx, ax | н/в | н/в | 4 | н/в | **-** |
| **17** | mov ax, k | 100 | н/в | н/в | н/в | **-** |
| **18** | cwde | 100 | н/в | н/в | н/в | **-** |
| **19** | idiv bx | 25 | н/в | н/в | н/в | **-** |
| **20** | idiv cx | 25 | н/в | н/в | н/в | **-** |
| **21** | mov res\_asm, ax | 25 | н/в | н/в | н/в | **-** |

**Завдання 3**: Написати програму для обчислення заданого цілочисленого виразу(табл. 5.5) для початкових даних в знаковому форматі довжиною 32 біт, використовуючи арифметичні операції ADD, ADC, INC, SUB, SBB, DEC, NEG, IMUL, IDIV, CBW, CWD. Провести тестові перевірки, відмітити нормальні та аномальні результати. Виконати покрокове виконання асемблерного коду та навести значення регістрів при їх виконанні.

Лістинг програми:

#include "pch.h"

#include <iostream>

#include <cstdlib>

int main()

{

long a, b, c, e, f, k, res\_c, res\_asm;

printf("\n(4 \* a / b + 1 - e) / (c \* b - 18 + a - f)\n");

printf("Enter the values of the range [-214783648...214783647]\n");

printf("a = "); scanf\_s("%ld", &a);

printf("b = "); scanf\_s("%ld", &b);

printf("c = "); scanf\_s("%ld", &c);

printf("e = "); scanf\_s("%ld", &e);

printf("f = "); scanf\_s("%ld", &f);

res\_c = (4 \* a / b + 1 - e) / (c \* b - 18 + a - f);

printf("Result C = %ld\n", res\_c);

\_\_asm

{

mov eax, 4; // <eax> = 4

mov ebx, a; // <ebx> = a

imul ebx; // <reax> = 4 \* a

mov ecx, b; // <ecx> = b

idiv ecx; // <eax> = 4 \* a / b

inc eax; // <eax> = 4 \* a / b + 1

sub eax, e; // <eax> = 4 \* a / b + 1 - e

mov k, eax; // k = 4 \* a / b + 1 - e

mov eax, c; // <eax> = c

imul b; // <reax> = c \* b

mov ecx, 1; // <ecx> = 1

idiv ecx; // <eax>

sub eax, 18; // <eax> = c \* b - 18

add eax, a; // <eax> = c \* b - 18 + a

sub eax, f; // <eax> = c \* b - 18 + a - f

mov ebx, eax; // <ebx> = <eax>

mov eax, k; // <eax> =k

cdq; // <reax>

idiv ebx; // <reax> = (4 \* a / b + 1 - e) / (c \* b - 18 + a - f)

idiv ecx; // <eax>

mov res\_asm, eax; // res = <eax>

}

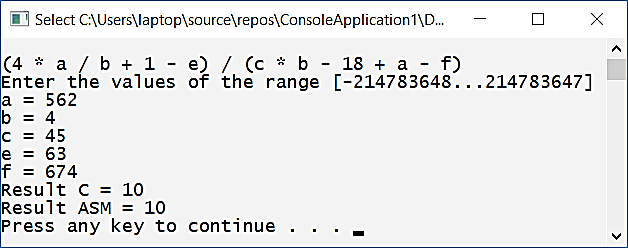
printf("Result ASM = %ld\n", res\_asm);

system("pause");

return 0;

}

Результат виконання програми:



**Значення регістрів при покроковому виконанні**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Крок** | **Команда** | **Значення регістра** | | | | **EFLAGS/FLAGS (CF, OF)** |
| **eax** | **reax** | **ebx** | **ecx** | **-** |
| **1** | mov eax, 4 | 4 | н/в | н/в | н/в | **-** |
| **2** | mov ebx, a | н/в | н/в | 562 | н/в | **-** |
| **3** | imul ebx | 2248 | н/в | н/в | н/в | **-** |
| **4** | mov ecx, b | н/в | н/в | н/в | 4 | **-** |
| **5** | idiv ecx | 562 | н/в | н/в | н/в | **-** |
| **6** | inc eax | 563 | н/в | н/в | н/в | **-** |
| **7** | sub eax, e | 500 | н/в | н/в | н/в | **-** |
| **8** | mov k, eax | н/в | н/в | н/в | н/в | **-** |
| **9** | mov eax, c | 45 | н/в | н/в | н/в | **-** |
| **10** | imul b | 180 | н/в | н/в | н/в | **-** |
| **11** | mov ecx, 1 | н/в | н/в | н/в | 1 | **-** |
| **12** | idiv ecx | 180 | н/в | н/в | н/в | **-** |
| **13** | sub eax, 18 | 162 | н/в | н/в | н/в | **-** |
| **14** | add eax, a | 724 | н/в | н/в | н/в | **-** |
| **15** | sub eax, f | 50 | н/в | н/в | н/в | **-** |
| **16** | mov ebx, eax | н/в | н/в | 50 | н/в | **-** |
| **17** | mov eax, k | 500 | н/в | н/в | н/в | **-** |
| **18** | cdqe | 500 | н/в | н/в | н/в | **-** |
| **19** | idiv ebx | 10 | н/в | н/в | н/в | **-** |
| **20** | idiv ecx | 10 | н/в | н/в | н/в | **-** |
| **21** | mov res\_asm, eax | 10 | н/в | н/в | н/в | **-** |

***Висновок:*** в ході виконання лабораторної роботи ознайомлено з основними командами мови Assembler для обчислення складних цілочисельних виразів; набуто практичних навичок в написанні програм для обчислення складних цілочисельних виразів на мові Assembler