Лабораторна робота № 5
ОРГАНІЗАЦІЯ УМОВНИХ ТА БЕЗУМОВНИХ ПЕРЕХОДІВ

Мета заняття: ознайомитися з основними командами мови Assembler для організації переходів; набути практичних навичок в написанні програм з організацією умовних та безумовних переходів на мові Assembler.

Теоретичні відомості

Основні команди мови Assembler для організації умовних переходів

Команди передачі управління дозволяють змінити природню послідовність виконання команд шляхом зміни вмісту регістра IP.

Базові команди передачі управління – це команди безумовного переходу на відповідну адресу в пам'яті комп'ютера, команди умовного переходу в залежності від результату перевірки умови і команди організації циклічних обчислень. Команди передачі управління не змінюють значення прапорців.

1. Команди безумовної передачі управління

Найбільш поширена команда безумовного переходу – команда JMP. Вона є аналогом відомого в алгоритмічних мовах оператора: goto мітка**.**

Мнемокод команди JMP отриманий в результаті скорочення слова JuMP – стрибок. Ця команда містить один операнд і має наступний формат:

JMP мітка

Мітка, як і в алгоритмічних мовах, відмічає потрібну команду, на яку потрібно передати управління, і може мати атрибут NEAR (по замовчуванню) або FAR. Якщо мітка має атрибут NEAR, безумовний перехід здійснюється в межах даного сегмента. В цьому випадку логіка роботи команд наступна:

<IP> = <IP> + зміщення_до_потрібної_команди

Таким чином, автоматично відбувається виконання команди, відміченої міткою.

Оскільки в даному випадку перехід здійснюється в межах сегменту зміщення_до_потрібної_команди не може перевищувати двох байтів. Необхідна нам команда може розміщуватися в сторону більших адрес, тоді говорять, що перехід здійснюється вперед. В цьому випадку зміщення_до_потрібної_команди – додатнє. Якщо навпаки, необхідна нам команда розміщується в сторону менших адрес, говорять про перехід назад. В цьому випадку зміщення_до _потрібної_команди – від'ємне.

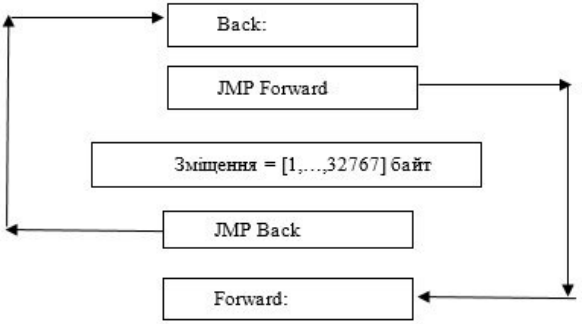

Рис.6.1 – Схема роботи команд JMP Forward (перехід вперед) і JMP Back (перехід назад) в межах одного сегменту

Можна зекономити один байт при реалізації команди безумовної передачі управління, якщо зробити перехід коротким (SHORT), тобто зміщення в межах [-128…127]. В цьому випадку перехід вперед потрібно записати таким чином: JMP SHORT Forward.

Для передачі управління назад слово SHORT можна не писати, тому що компілятор при побудові машинного коду сам визначить, яке вийде зміщення, оскільки на момент компіляції команда переходу JMP Back адреса, на яку вказує мітка Back, йому уже відома (перегляд початкового модуля відбувається зверху вниз). По замовчуванню компілятор робить одне проходження.

2. Команди умовної передачі управління Jcc

Базових команд умовного переходу всього 17, але вони можуть мати різну мнемоніку, тому в загальному виходить 31 команда. Читати їх досить просто, якщо знаєш ключ.

Перша буква команди J від уже відомого нам слова (Jump - стрибок). Інші букви (cc) в скороченому вигляді описують умову переходу. По цій ознаці всі команди умовного переходу можна розбити на три групи.

Перша група команд умовного переходу:

1. **E** – Equal (дорівнює).
2. **N** – Not (не, заперечення).
3. **G** – Greater (більше) – застосовується для чисел із знаком.
4. **L** – Less (менше) – застосовується для чисел із знаком.

5. **A** – Above (вище, більше) – застосовується для чисел без знака.
6. **B** – Below (нрижче, менше) - застосовується для чисел без знака.

Наприклад, команда JL означає перехід, якщо менше. Їй еквівалентна команда-синонім JNGE – перехід, якщо не більше і не дорінює. Різниця в командах переходу для знакових і беззнакових даних пояснюється тим, що вони реагують на різні прапорці (для знакових даних суттєвий прапорець SF, а для беззнакових - CF).

Умовний перехід для цієї групи команд зазвичай реалізується в два кроки:

- Порівняння, в результаті чого формуються прапорці (регістр FLAGS).
- Умовна передача управління (Jcc Коротка_мітка) на відмічену команду в залежності від значення прапорців.

Ця група операцій найчастіше виконується в парі з командою порівняння CMP.

CMP приймач, джерело
Jcc Коротка_мітка

Команда CMP (CoMPare operands) – порівняння операндів. В результаті її виконання утворюються прапорці, які потім аналізуються командами умовного переходу. За станом прапорців зручно спостерігати під час покрокового виконання.

Таблиця 6.1

Команди умовного переходу групи 1

Умова порівняння (CMP)	Мнемокод	Стан прапорців
Для будь-яких чисел і кодів		
Приймач=джерело	**JE**	**ZF=1**
Приймач<>джерело	**JNE**	**ZF=0**
Для чисел із знаком		
Приймач<джерело	**JL/JNGE**	**SF<>OF**
Приймач<=джерело	**JLE/JNG**	**SF<>OF or ZF=1**
Приймач>джерело	**JG/JNLE**	**SF=OF and ZF=0**
Приймач>=джерело	**JGE/JNL**	**SF=OF**
Для чисел без знаку		
Приймач<джерело	**JB/JNAE**	**CF=1**
Приймач<=джерело	**JBE/JNA**	**CF=1 or ZF=1**
Приймач>джерело	**JA/JNBE**	**CF=0 and ZF=0**
Приймач>=джерело	**JAE/JNB**	**CF=0**

Друга група команд умовного переходу реагує на те чи інше значення конкретного прапорця. Тому в мнемоніці даної групи команд завжди вказується перша літера перевіряємого прапорця. Ці команди не потребують обов'язкової наявності команд порівняння перед свохм виконанням. Їм досить будь-якої команди, яка виробляє потрібний прапорець.

Таблиця 6.2

Команди умовного переходу групи 2

Мнемокод	Стан прапорців	Мнемокод	Стан прапорців
JZ/JE	**ZF=1**	**JNZ/JNE**	**ZF=0**
JS	**SF=1**	**JNS**	**SF=0**
JC/JB/JNAE	**CF=1**	**JNC/JAE/JNB**	**CF=0**
JO	**OF=1**	**JNO**	**OF=0**
JP	**PF=1**	**JNP**	**PF=0**

В третю групу команд умовного переходу входить всього одна команда JCXZ. Вона, на відміну від попередніх команд, перевіряє не прапорці, а стан регістра CX на нуль (Jump if CX is Zero).

Синтаксис:

JCXZ Коротка_мітка

Логіка роботи команди:

If (CX=0) then goto Коротка_мітка

Це дуже важлива команда, яка використовується при організації циклів.

Приклад: Написати програму для обчислення заданого умовного цілочисельного виразу для 16-бітних даних, використовуючи команди порівняння, умовного і безумовного переходів. Результат X – теж цілочисельний і його діапазон (формат) залежить від специфіки вирішуваного умовного виразу. Провести тестові перевірки, відмітити нормальні та аномальні результати. Виконати покрокове виконання асемблерного коду та навести значення регістрів при їх виконанні.

Дано:

$$X = \begin{cases} 52 * b / a + b, & якщо \quad a > b, \\ -125, & якщо \quad a = b, \\ (a-5)/b, & якщо \quad a < b; \end{cases}$$

Лістинг програми:

```
#include <iostream>
#include <stdio.h>
#include "windows.h"
#include <stdlib.h>
#include <conio.h>

int main()
{
	SetConsoleCP(1251);
	SetConsoleOutputCP(1251);
	short int a, b, x_c, x_asm;
	printf("Введіть значення з діапазону [-32768...32767]:\n");
	printf("a = "); scanf_s("%hd", &a);
	printf("b = "); scanf_s("%hd", &b);

	if (a > b)
	{	x_c = (52 * b) / a + b;		}

	else if (a < b)
	{	x_c = (a - 5) / b;		}

	else
	{	x_c = -125;			}

	printf("Результат на мові C++ X = %hd\n", x_c);

	__asm
	{
	    mov ax, a		// <ax> = a
	    mov bx, b		// <bx> = b
	    cmp ax, bx		// порівнюємо значення регістрів <ax> та <bx>

	    jg mark1		// a > b
	    je mark2		// a = b
	    jl mark3		// a < b

	    mark1 :
		xchg ax, bx	// змінюємо вміст регістрів <ax> та <bx>
		mov cx, 52	// <cx> = 52
		imul cx		// <dx:ax>:  = 52*b
		idiv bx		// <ax> = (52*b)/a
		add ax, b	// <ax> = (52*b)/a+b
		mov x_asm, ax	// x = (52*b)/a+b
		jmp exit1	// переходимо до мітки exit1

	    mark2 :
		mov x_asm, -125 // x_asm = -125
		jmp exit1	// переходимо до мітки exit1
```

```
            mark3 :
               sub ax, 5        // <ax> = a-5
               cwd              // <dx:ax> = a-5
               idiv bx          // <ax> = (a-5)/b
               mov x_asm, ax    // x_asm = (a-5)/b
               jmp exit1        // переходимо до мітки exit1

            exit1 :
        }

        printf("Результат на мові Assembler X = %hd\n", x_asm);
        system("Pause");
        return 0;

}
```

Проведемо тестову перевірку роботи програми(рис.6.2 – рис.6.4)

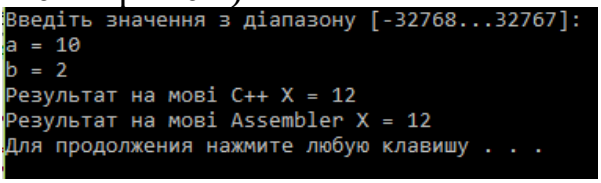

Рисунок 6.2 – Перевірка роботи програми для значення a > b

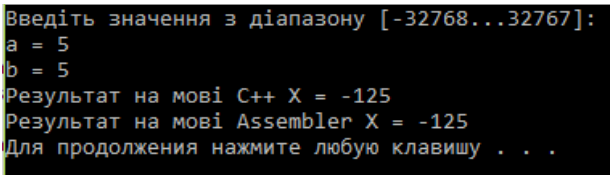

Рисунок 6.3 – Перевірка роботи програми для значення a = b

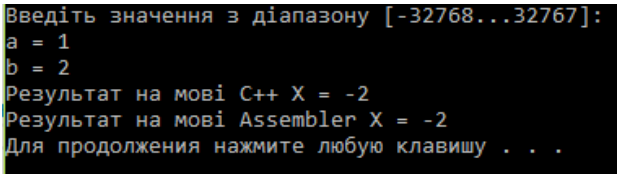

Рисунок 6.4 – Перевірка роботи програми для значення a < b

Значення регістрів при покроковому виконанні

Команда	Значення регістра				EFLAGS/FLAGS (CF, OF)
	ax	bx	cx	< dx:ax >	
mov ax, a	a	н/в	н/в	н/в	–
mov bx, b	н/в	b	н/в	н/в	–
cmp ax, bx	Порівнюються регістри ax і bx				
jg mark1	Якщо a>b, то переходимо на мітку mark1				
jg mark2	Якщо a=b, то переходимо на мітку mark2				
jg mark3	Якщо a<b, то переходимо на мітку mark3				
mark1:	Мітка mark1(код буде виконуватись в цій мітці, якщо умова – істина)				
xchg ax, bx	н/в	a	н/в	н/в	–
mov cx, 52	н/в	н/в	52	н/в	–
imul cx	н/в	н/в	н/в	52*b	1
idiv bx	(52*b)/a	н/в	н/в	н/в	–
add ax, b	(52*b)/a+b	н/в	н/в	н/в	–
mov x_asm, ax	н/в	н/в	н/в	н/в	–
jmp exit:	Перейти на мітку exit(вихід з ассемблерної вставки)				
mark2:	Мітка mark2(код буде виконуватись в цій мітці, якщо умова – істина)				
mov x_asm, -125	н/в	н/в	н/в	н/в	–
jmp exit:	Перейти на мітку exit(вихід з ассемблерної вставки)				
mark3:	Мітка mark3(код буде виконуватись в цій мітці, якщо умова – істина)				
sub ax, 5	a-5	н/в	н/в	н/в	–
Cwd	н/в	н/в	н/в	a-5	1
idiv bx	(a-5)/b	н/в	н/в	н/в	–
mov x_asm, ax	н/в	н/в	н/в	н/в	–
jmp exit:	Перейти на мітку exit(вихід з ассемблерної вставки)				
exit:	exit(вихід з ассемблерної вставки)				

Введемо вхідні данні, які виходять за межі допустимого діапазону значень(рис. 6.5).

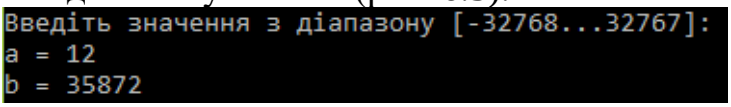

Рисунок 6.5 – Введення даних, які виходять за межі допустимого діапазону значень

В результаті виникає помилка(рис. 6.6).

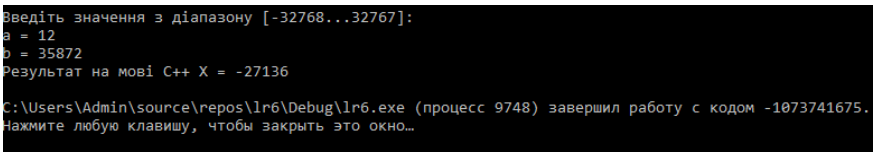

Рисунок 6.6 – Перевірка роботи програми для значень вхідних даних, які виходять за межі допустимого діапазону значень

Отже, необхідно задати умову для перевірки введених даних.

Крім того, необхідно здійснити перевірку ділення на нуль та переповнення.

В даному умовному виразі переповнення може виникнути у двох випадках: в результаті обчислення виразу 52*b/a+b, якщо a>b та в результаті обчислення чисельника виразу (a-5)/b, якщо a<b.

Лістинг програми:

```
#include <iostream>
#include <stdio.h>
#include "windows.h"
#include <stdlib.h>
#include <conio.h>


int main()
{
        SetConsoleCP(1251);
        SetConsoleOutputCP(1251);
        short a, b, x_c, x_asm, er = 0, over = 0;
        printf("Введіть значення з діапазону [-32768...32767]:\n");
        printf("a = "); scanf_s("%hd", &a);
        printf("b = "); scanf_s("%hd", &b);

        if (a > b)
        {
           if (a == 0)
           {    printf("Помилка: ");    }

           else
           {    x_c = (52 * b) / a + b;
                if (x_c < -32768 || x_c > 32767)
                {       printf("Помилка: ");    }
                else
                {  printf("Результат на мові C++ X = %hd\n", x_c); }
            }
         }

        else if (a < b)
         {
           if (b == 0)
           {   printf("Помилка: ");    }

           else if ((a - 5) < -32768 || (a - 5) > 32767)
           {   printf("Помилка: ");    }

           else
               {       x_c = (a - 5) / b;
                       printf("Результат на мові C++ X = %hd\n", x_c);
               }
             }

        else
          {
                x_c = -125;
                printf("Результат на мові C++ X = %hd\n", x_c);
          }
```

```
__asm
{
   mov ax, a      // <ax> = a
   mov bx, b      // <bx> = b
   cmp ax, bx     // порівнюємо значення регістрів <ax> та <bx>

        jg mark1       // a > b
        je mark2       // a = b
        jl mark3       // a < b

        mark1 :
        cmp ax, 0              // порівнюємо значення <ax> з нулем
               je error1       // помилка "ділення на 0"
               xchg ax, bx   // змінюємо вміст регістрів <ax> <bx>
               mov cx, 52     // <cx> = 52
               imul cx        // <dx:ax> = 52*b
               idiv bx        // <ax> = (52*b)/a
               add ax, b      // <ax> = (52*b)/a+b
               jo error2      // помилка "переповнення"
               mov x_asm, ax  // x = (52*b)/a+b
               jmp exit1      // переходимо до мітки exit1
        mark2 :
               mov x_asm, -125 // x_asm = -125
               jmp exit1       // переходимо до мітки exit1
        mark3 :
               cmp bx, 0       // порівнюємо значення <bx> з нулем
               je error3       // помилка "ділення на 0"
               sub ax, 5       // <ax> = a-5
               jo error4       // помилка "переповнення"
               cwd             // <dx:ax> = a-5
               idiv bx         // <ax> = (a-5)/b
               mov x_asm, ax   // x_asm = (a-5)/b
               jmp exit1       // переходимо до мітки exit1

         error1 :
               mov er, 1
               jmp exit1
        error2 :
               mov over, 1
               jmp exit1
        error3 :
               mov er, 1
               jmp exit1
        error4 :
               mov over, 1
               jmp exit1



exit1 :
```

```
}

if (er == 1)
{       printf("ділення на 0!\n");     }

else if (over == 1)
{       printf("переповнення!\n");     }

else if (er == 0 && over == 0)
{       printf("Результат на мові Assembler X = %hd\n", x_asm);  }

system("Pause");
return 0;
}
```

Перевіримо виникнення помилки при діленні на 0(рис. 6.7)

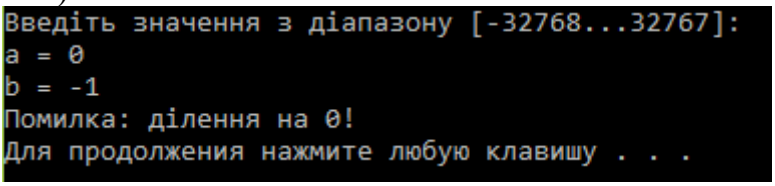

Рисунок 6.7 – Виникнення помилки при діленні на 0

Підберемо значення для перевірки виникнення переповнення(рис. 6.8)

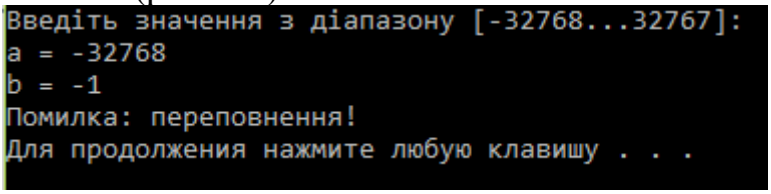

Рисунок 6.8 – Виникнення переповнення

Завдання на лабораторну роботу

1. Написати програму для обчислення заданого умовного цілочисельного виразу (табл. 6.3) для 8-бітних даних, використовуючи команди порівняння, умовного і безумовного переходів. Результат X – теж цілочисельний і його діапазон (формат) залежить від специфіки вирішуваного умовного виразу. Провести тестові перевірки, відмітити нормальні та аномальні результати. Виконати покрокове виконання асемблерного коду та навести значення регістрів при їх виконанні.

Таблиця 6.3

Дані для розрахунку

№ Варіанту	Вираз
1	$X = \begin{cases} (a-b)/a-3, & якщо \quad a > b, \\ 2, & якщо \quad a = b, \\ (a^3+1)/b, & якщо \quad a < b; \end{cases}$
2	$X = \begin{cases} a/b+10, & якщо \quad a < b, \\ -51, & якщо \quad a = b, \\ (a*b-4)/a, & якщо \quad a > b; \end{cases}$
3	$X = \begin{cases} a/b-1, & якщо \quad a < b, \\ -25, & якщо \quad a = b, \\ (b^3-5)/a, & якщо \quad a > b; \end{cases}$
4	$X = \begin{cases} (a*b-1)/a, & якщо \quad a > b, \\ 125, & якщо \quad a = b, \\ (a-5)/b, & якщо \quad a < b; \end{cases}$

5	$X = \begin{cases} a/b+31, & якщо \quad a > b, \\ -25, & якщо \quad a = b, \\ (5*b-1)/a, & якщо \quad a < b; \end{cases}$
6	$X = \begin{cases} b/a+1, & якщо \quad a < b, \\ 25, & якщо \quad a = b, \\ (a^3-5)/b, & якщо \quad a > b; \end{cases}$
7	$X = \begin{cases} a/b+1, & якщо \quad a < b, \\ -2, & якщо \quad a = b, \\ (a-b)/a, & якщо \quad a > b; \end{cases}$
8	$X = \begin{cases} b/a-1, & якщо \quad a < b, \\ -105, & якщо \quad a = b, \\ (a-235)/b, & якщо \quad a > b; \end{cases}$
9	$X = \begin{cases} a/b+1, & якщо \quad a > b, \\ a+25, & якщо \quad a = b, \\ (a*b-2)/a, & якщо \quad a < b; \end{cases}$
10	$X = \begin{cases} (a*a-b)/a, & якщо \quad a > b, \\ -a, & якщо \quad a = b, \\ (a*b-1)/b, & якщо \quad a < b; \end{cases}$
11	$X = \begin{cases} a/b-1, & якщо \quad a < b, \\ 25-a, & якщо \quad a = b, \\ (b-5)/a, & якщо \quad a > b; \end{cases}$
12	$X = \begin{cases} a/b+2, & якщо \quad a > b, \\ 8, & якщо \quad a = b, \\ (b-9)/a, & якщо \quad a < b; \end{cases}$

13	$X = \begin{cases} a/b+1, & якщо \quad a<b, \\ -71, & якщо \quad a=b, \\ (a-b)/a, & якщо \quad a>b; \end{cases}$
14	$X = \begin{cases} -5+b/a, & якщо \quad a>b, \\ 45, & якщо \quad a=b, \\ (3*a-6)/b, & якщо \quad a<b; \end{cases}$
15	$X = \begin{cases} a/b-4, & якщо \quad a<b, \\ -55, & якщо \quad a=b, \\ (b-5)/a, & якщо \quad a>b; \end{cases}$
16	$X = \begin{cases} a/b+11, & якщо \quad a>b, \\ -11, & якщо \quad a=b, \\ (3*b-9)/a, & якщо \quad a<b; \end{cases}$
17	$X = \begin{cases} 2*a/b+1, & якщо \quad a>b, \\ -5, & якщо \quad a=b, \\ (a^3-9)/a, & якщо \quad a<b; \end{cases}$
18	$X = \begin{cases} b/a+1, & якщо \quad a>b, \\ -20, & якщо \quad a=b, \\ (a-45)/b, & якщо \quad a<b; \end{cases}$
19	$X = \begin{cases} a/b+2, & якщо \quad a>b, \\ -100, & якщо \quad a=b, \\ (b-100)/a, & якщо \quad a<b; \end{cases}$
20	$X = \begin{cases} a/b+1, & якщо \quad a>b, \\ -100, & якщо \quad a=b, \\ (a*b-9)/a, & якщо \quad a<b; \end{cases}$

21	$X = \begin{cases} a/b - a, & якщо \quad a < b, \\ -b & якщо \quad a = b, \\ (a*b-8)/a, & якщо \quad a > b; \end{cases}$
22	$X = \begin{cases} b^2/a + 1, & якщо \quad a > b, \\ 2*a, & якщо \quad a = b, \\ (a-10)/b, & якщо \quad a < b; \end{cases}$
23	$X = \begin{cases} b/a + 1, & якщо \quad a > b, \\ -107, & якщо \quad a = b, \\ (a-b)/b, & якщо \quad a < b; \end{cases}$
24	$X = \begin{cases} b/a - 5, & якщо \quad a > b, \\ -109, & якщо \quad a = b, \\ (a-6)/b, & якщо \quad a < b; \end{cases}$
25	$X = \begin{cases} a/b - 4, & якщо \quad a < b, \\ -115, & якщо \quad a = b, \\ (b^3-5)/a, & якщо \quad a > b; \end{cases}$
26	$X = \begin{cases} b/a - 141, & якщо \quad a < b, \\ -131, & якщо \quad a = b, \\ (3*a-9)/b, & якщо \quad a > b; \end{cases}$
27	$X = \begin{cases} a/4 - b, & якщо \quad a > b, \\ -57, & якщо \quad a = b, \\ (a^3-5)/b, & якщо \quad a < b; \end{cases}$
28	$X = \begin{cases} a/b + 1, & якщо \quad a > b, \\ -25, & якщо \quad a = b, \\ (b-45)/a, & якщо \quad a < b; \end{cases}$

29	$X = \begin{cases} b/a-2, & якщо \quad a > b, \\ -104, & якщо \quad a = b, \\ (a-100)/b, & якщо \quad a < b; \end{cases}$
30	$X = \begin{cases} b/a-1, & якщо \quad a > b, \\ -120, & якщо \quad a = b, \\ (a*b-9)/b, & якщо \quad a < b; \end{cases}$

2. Написати програму для обчислення заданого умовного цілочисельного виразу (табл. 6.4) для 32-бітних даних, використовуючи команди порівняння, умовного і безумовного переходів. Результат X – теж цілочисельний і його діапазон (формат) залежить від специфіки вирішуваного умовного виразу. Провести тестові перевірки, відмітити нормальні та аномальні результати. Виконати покрокове виконання асемблерного коду та навести значення регістрів при їх виконанні.

Таблиця 6.4

Дані для розрахунку

№ Варіанту	Вираз
1	$X = \begin{cases} (a^2-1)/2, & якщо\ a > 0, \\ (3*a+1)^2, & у\ всіх\ інших\ випадках; \end{cases}$
2	$X = \begin{cases} (b^2+10)/3, & якщо \quad a = 0, \\ (b-a)*a, & у\ всіх\ інших\ випадках; \end{cases}$
3	$X = \begin{cases} (a*b-1)/5, & якщо\ a = 3, \\ (a-3)*b, & у\ всіх\ інших\ випадках; \end{cases}$

4	$X = \begin{cases} (2-b)/a+3, & \text{якщо } a>0, \\ (a+7)^2, & \text{у всіх інших випадках;} \end{cases}$
5	$X = \begin{cases} a^2 - a*b+3, & \text{якщо } a>b, \\ (-b+4*a)/2, & \text{у всіх інших випадках;} \end{cases}$
6	$X = \begin{cases} a^3 - a*b, & \text{якщо } a \neq 0, \\ -a*b+5/a, & \text{у всіх інших випадках;} \end{cases}$
7	$X = \begin{cases} (a-b)^2, & \text{якщо } a>b, \\ (a+5)/b, & \text{у всіх інших випадках;} \end{cases}$
8	$X = \begin{cases} (-a*b-3)/2, & \text{якщо } a=b, \\ a*b^2+b/a, & \text{у всіх інших випадках;} \end{cases}$
9	$X = \begin{cases} -a^2+b^2, & \text{якщо } a \neq b, \\ 5*a^2/b, & \text{у всіх інших випадках;} \end{cases}$
10	$X = \begin{cases} (b+b)^2/a, & \text{якщо } b>0, \\ a*(b-3), & \text{у всіх інших випадках;} \end{cases}$
11	$X = \begin{cases} (a*b)/(a+b), & \text{якщо } a>b, \\ (b-5)/a, & \text{у всіх інших випадках;} \end{cases}$
12	$X = \begin{cases} (a-2*b)^2, & \text{якщо } a=2, \\ (3*b)/(a-2), & \text{у всіх інших випадках;} \end{cases}$
13	$X = \begin{cases} (-a+7)/(-b-4), & \text{якщо } a \neq b, \\ -a*(b+10), & \text{у всіх інших випадках;} \end{cases}$
14	$X = \begin{cases} (a-1)*a^2, & \text{якщо } a<100, \\ (b+a)/100, & \text{у всіх інших випадках;} \end{cases}$

15	$X=\begin{cases}5*a*b-b^2, & \textit{якщо} \quad a\neq b,\\ (a+100)/b, & \textit{у всіх інших випадках};\end{cases}$
16	$X=\begin{cases}b-10*a, & \textit{якщо} \quad b=80,\\ -a^2/b+20 & \textit{у всіх інших випадках};\end{cases}$
17	$X=\begin{cases}a^2*b+5*b, & \textit{якщо} \quad a<5,\\ a*(b+9)/2, & \textit{у всіх інших випадках};\end{cases}$
18	$X=\begin{cases}b/2+a/4, & \textit{якщо} \quad (a,b)>0,\\ (a*b+5)/(a*b+5), & \textit{у всіх інших випадках};\end{cases}$
19	$X=\begin{cases}5*a^2+b/5, & \textit{якщо} \quad a\neq b,\\ (a^2+b^2+10)/(a*b), & \textit{у всіх інших випадках};\end{cases}$
20	$X=\begin{cases}(a^2+4*a*b+2)/b, & \textit{якщо} \quad a<4,\\ a^2/(b-2)+4*b, & \textit{у всіх інших випадках};\end{cases}$
21	$X=\begin{cases}a^3/10+4*b, & \textit{якщо} \quad a<0,\\ (a+2*b)/a+25, & \textit{у всіх інших випадках};\end{cases}$
22	$X=\begin{cases}(a+b)^2/(2*a*b), & \textit{якщо} \quad a=b,\\ a*b^2+5*a+3, & \textit{у всіх інших випадках};\end{cases}$
23	$X=\begin{cases}(10*a-5*b)/2, & \textit{якщо} \quad a\neq b,\\ a^2/(2*b)+7, & \textit{у всіх інших випадках};\end{cases}$
24	$X=\begin{cases}(7*a*b)/(a+b), & \textit{якщо} \quad a>b,\\ 100/(a^2+2*b), & \textit{у всіх інших випадках};\end{cases}$
25	$X=\begin{cases}a*b^2-a*b+5, & \textit{якщо} \quad a<7,\\ (a*b)/5+a, & \textit{у всіх інших випадках};\end{cases}$

26	$X = \begin{cases} a^2 + 7 - 2*b, & \textit{якщо} \quad b > 10, \\ (a*b)/(b/2), & \textit{у всіх інших випадках}; \end{cases}$
27	$X = \begin{cases} -a^3 + b^2, & \textit{якщо} \quad a < 0, \\ -(a/b)/25, & \textit{у всіх інших випадках}; \end{cases}$
28	$X = \begin{cases} (b+7)^2 - a, & \textit{якщо} \quad a < 25, \\ -a/b + 1, & \textit{у всіх інших випадках}; \end{cases}$
29	$X = \begin{cases} (a*b)/(3+b), & \textit{якщо} \quad a < b, \\ (a^3 - 7)/a, & \textit{у всіх інших випадках}; \end{cases}$
30	$X = \begin{cases} (7*a)/(a-b), & \textit{якщо} \quad a > 0, \\ (-a-9)^2/b, & \textit{у всіх інших випадках}; \end{cases}$

Контрольні питання

1. Команда порівняння CMP. Призначення та синтаксис.
2. Команди умовного переходу для будь-яких чисел.
3. Команди умовного переходу для чисел без знаку. Призначення та синтаксис.
4. Команди умовного переходу для чисел зі знаком. Призначення та синтаксис.
5. Команди умовного переходу для ознаки ZF. Призначення та синтаксис.
6. Команди умовного переходу для ознаки CF. Призначення та синтаксис.
7. Команди умовного переходу для ознаки SF. Призначення та синтаксис.
8. Команди умовного переходу для ознаки OF. Призначення та синтаксис.
9. Команди умовного переходу для ознаки PF. Призначення та синтаксис.
10. Команда умовного переходу JCXZ. Призначення та синтаксис.