МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпропетровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №4**

**з дисципліни «архітектура комп’ютера»**

**на тему: «Реалізація розгалужень мовою Асемблер.»**

Виконав: студент гр. ПЗ1911

Сіньков Г.О.

Прийняла: ас. каф. КІТ

Нежуміра О. І.

Дніпро, 2019

**Лабораторна робота № 4**

Тема. Реалізація циклічних алгоритмів мовою Асемблера.

Мета. Набути практичних навичок розробки програм циклічної структури.

**1.Постановка задачи**

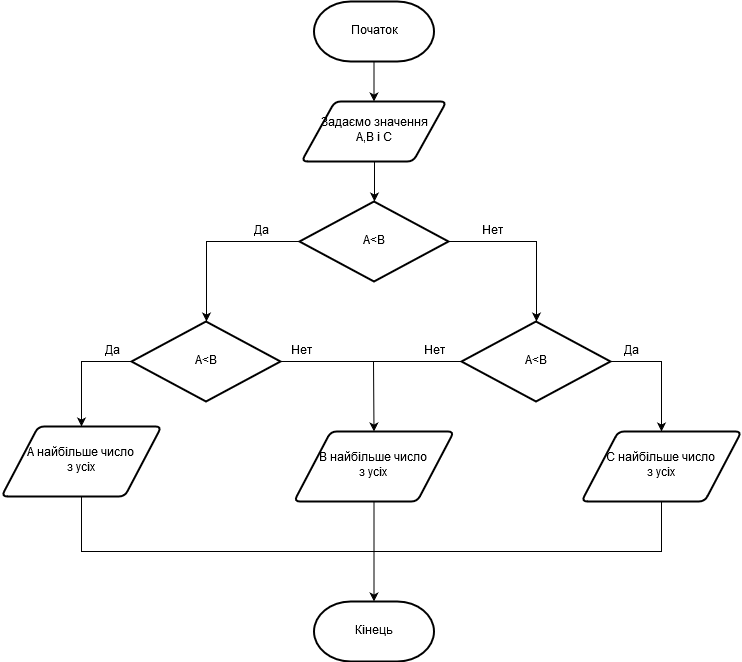
1.1 Загальне завдання

Завдання наведено в таблиці. Вибір завдання здійснюється згідно з порядковим номером у списку підгрупи. Вхідні дані задаються у сегменті даних.

1.2. Індивідуальне завдання

|  |  |
| --- | --- |
| № п/п | Зміст завдання |
| 7 | Для трьох довільних чисел вказати найменше. Результат подається у вигляді повідомлення. |

**2. Алгоритм розв’язання задачі**



**3. Лістинг програми**

Turbo Assembler Version 3.0 11/04/19 22:20:32 Page 1

lab4.asm

1 ; сегмент даних

2 0000 data segment para public 'data'

3 0000 0D 0A 41 2D E1 EE EB+ A\_answer db 13,10,'A-больше всех',13,10,13,10,'$'

4 FC F8 E5 20 E2 F1 E5+

5 F5 0D 0A 0D 0A 24

6 0014 0D 0A 42 2D E1 EE EB+ B\_answer db 13,10,'B-больше всех',13,10,13,10,'$'

7 FC F8 E5 20 E2 F1 E5+

8 F5 0D 0A 0D 0A 24

9 0028 0D 0A 43 2D E1 EE EB+ C\_answer db 13,10,'C-больше всех',13,10,13,10,'$'

10 FC F8 E5 20 E2 F1 E5+

11 F5 0D 0A 0D 0A 24

12 003C 39 A db 57

13 003D 19 B db 25

14 003E 9A C db 154

15 003F data ends

16 ; сегмент стека

17 0000 stk segment stack

18 0000 0100\*(??) db 256 dup (?) ;поле для стека розміром 256 байт

19 0100 stk ends

20 ; сегмент команд

21 0000 code segment para public 'code'

22 assume cs:code, ds:data, ss:stk

23 0000 start:

24 0000 B8 0000s mov ax, data ;ініціалізація сегментного регістра ds

25 0003 8E D8 mov ds, ax ; на програмний сегмент даних data

26 0005 B8 0000 mov ax, 0

27

28 0008 8A 26 003Cr mov ah, A

29 000C A0 003Dr mov al, B

30

31 000F 3A E0 cmp ah, al ; A<B

32 0011 7C 14 jl \_c1

33

34 0013 8A 26 003Er mov ah, C

35 0017 3A 26 003Er cmp ah, C ; A<C

36 001B 7C 1C jl \_else

37

38 001D BA 0000r lea dx, A\_answer

39 0020 B4 09 mov ah, 09h

40 0022 CD 21 int 21h

41 0024 EB 1D 90 jmp \_exit

42

43

44 0027 \_c1:

45 0027 8A 26 003Er mov ah, C

46 002B 3A C4 cmp al, ah ; B<C

47 002D 7C 0A jl \_else

48

49 002F BA 0014r lea dx, B\_answer

50 0032 B4 09 mov ah, 09h

51 0034 CD 21 int 21h

52

53 0036 EB 0B 90 jmp \_exit

54

55 0039 \_else:

56

57 0039 BA 0028r lea dx, C\_answer

58 003C B4 09 mov ah, 09h

59 003E CD 21 int 21h

60

61 0040 EB 01 90 jmp \_exit

62

63 0043 \_exit:

64 0043 B8 4C00 mov ax, 4C00h

65 0046 CD 21 int 21h

66 0048 code ends

67 end start

Turbo Assembler Version 3.0 11/04/19 22:20:32 Page 3

Symbol Table

Symbol Name Type Value

??DATE Text "11/04/19"

??FILENAME Text "lab4 "

??TIME Text "22:20:32"

??VERSION Number 0300

@CPU Text 0101H

@CURSEG Text CODE

@FILENAME Text LAB4

@WORDSIZE Text 2

A Byte DATA:003C

A\_ANSWER Byte DATA:0000

B Byte DATA:003D

B\_ANSWER Byte DATA:0014

C Byte DATA:003E

C\_ANSWER Byte DATA:0028

START Near CODE:0000

\_C1 Near CODE:0027

\_ELSE Near CODE:0039

\_EXIT Near CODE:0043

Groups & Segments Bit Size Align Combine Class

CODE 16 0048 Para Public CODE

DATA 16 003F Para Public DATA

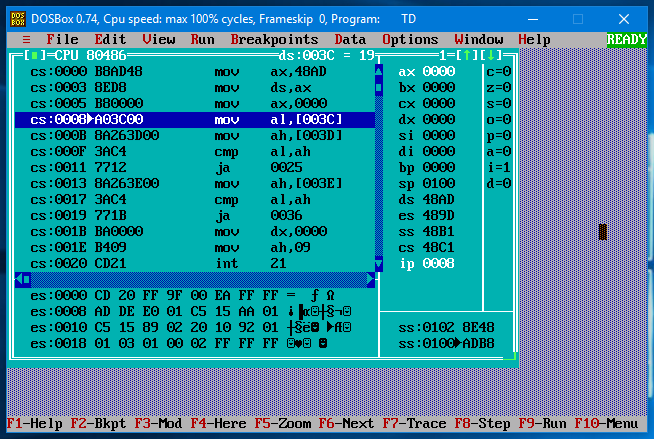
STK 16 0100 Para Stack

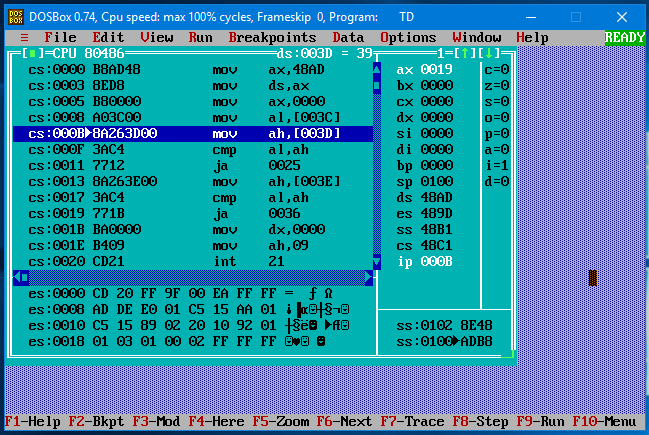
**4. Тести**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Вхідні дані | Умови переходу | Очікувані результати |
| 1 | A = 25, B = 57, C = 154 | A<B(да)  A<C(да) | A – менше всіх |
| 2 | A = 57, B = 25, C = 154 | A<B(ні)  B<C(да) | B – менше всіх |
| 3 | A = 57, B = 154, C = 25 | A<B(ні, але по умові в програмі так)  B<C(ні) | C – менше всіх |
| 4 | A = 154, B = 57, C = 25 | A<B(ні)  B<C(ні) | C – менше всіх |

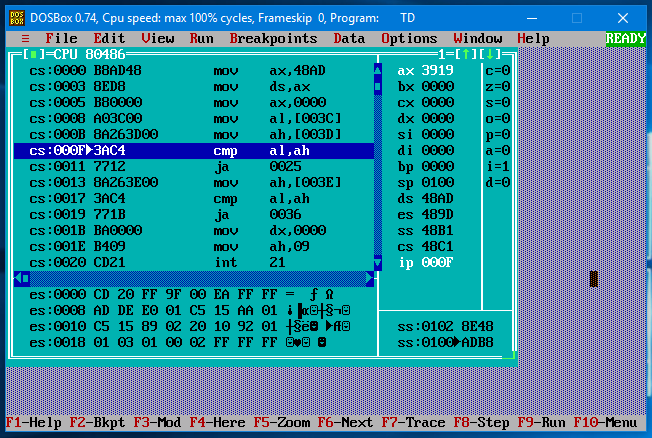
**5. Результати виконання програми**

1) Записую значення змінних A і B в регістри

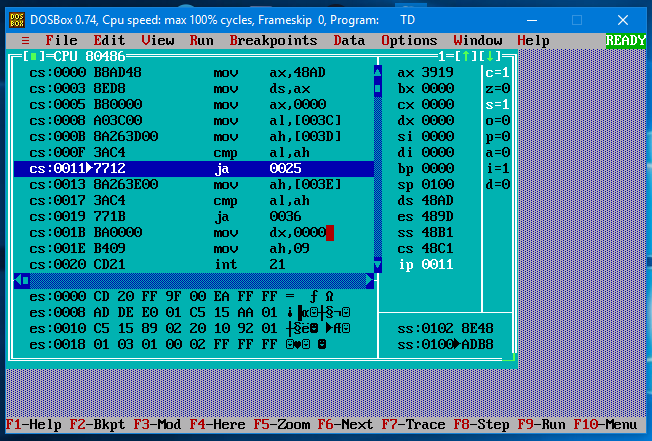




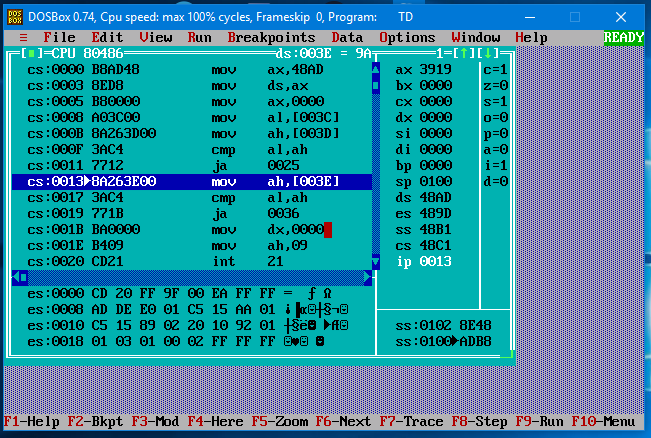
2) Порівнюю числа A і B



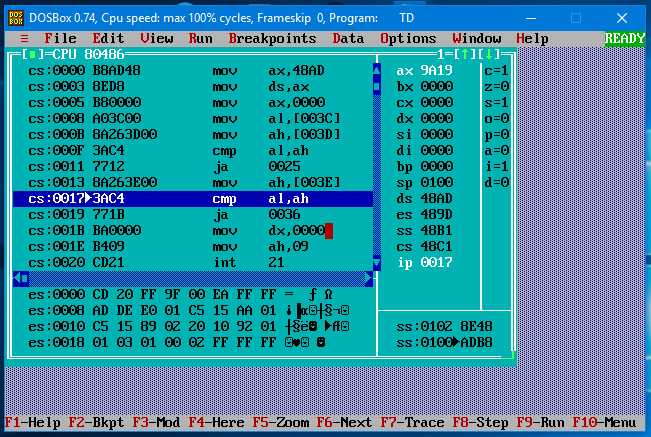
3) Переходимо на мітку \_с1 якщо умова(A<B) а для виконання використав прапор навпаки (A>B)



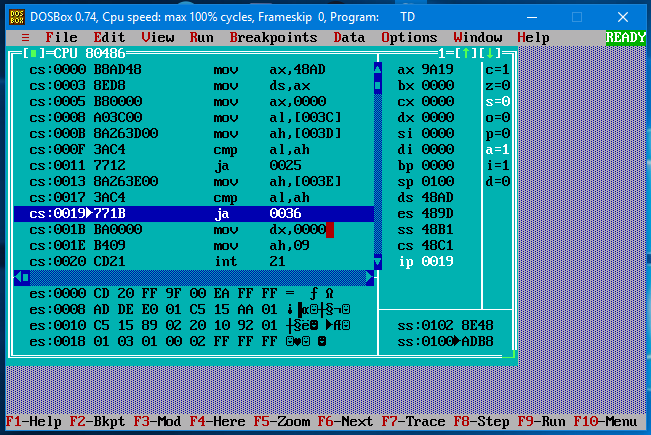
4) Записую значення змінної C в регістр



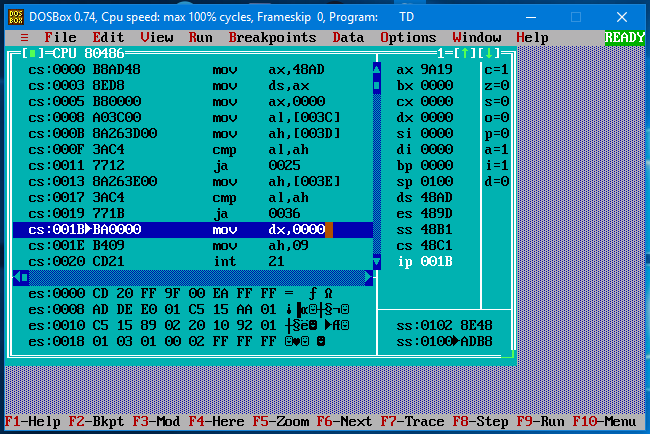
5) Порівнюємо числа A і C

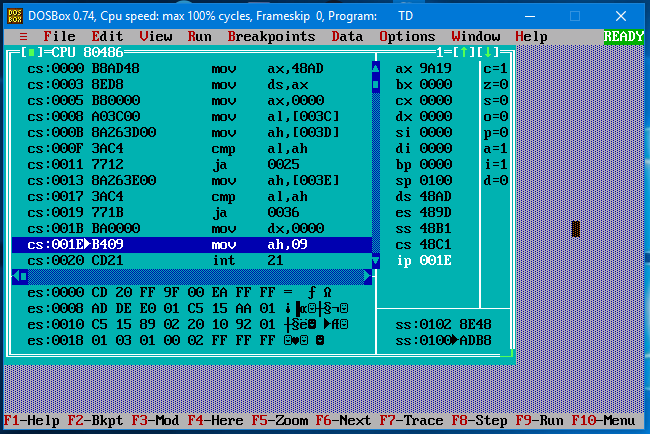


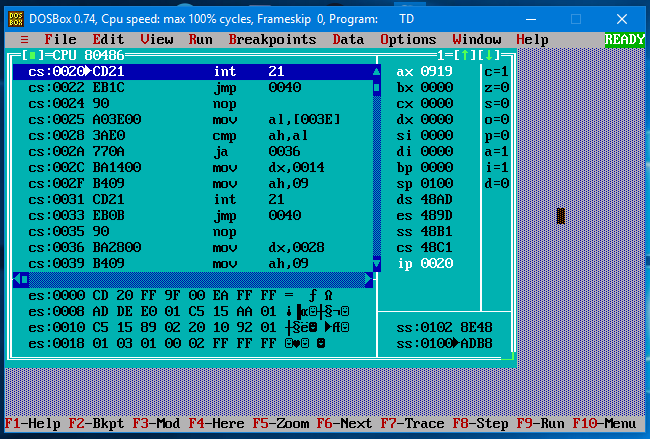
6) Переходимо на мітку \_с1 якщо умова(A<C) а для виконання використав прапор навпаки (A>C)



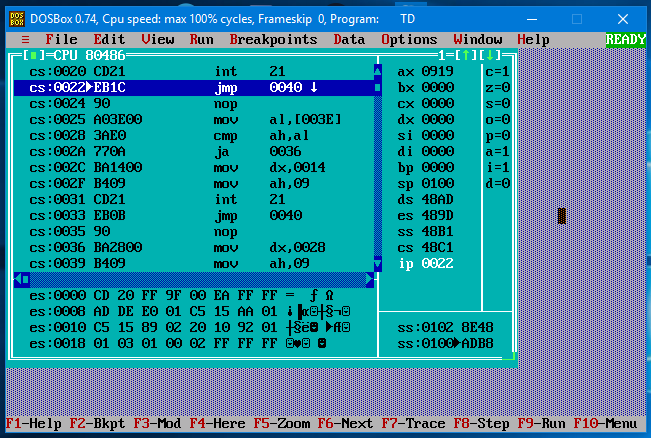
7) Блок виводу повідомлення на екран



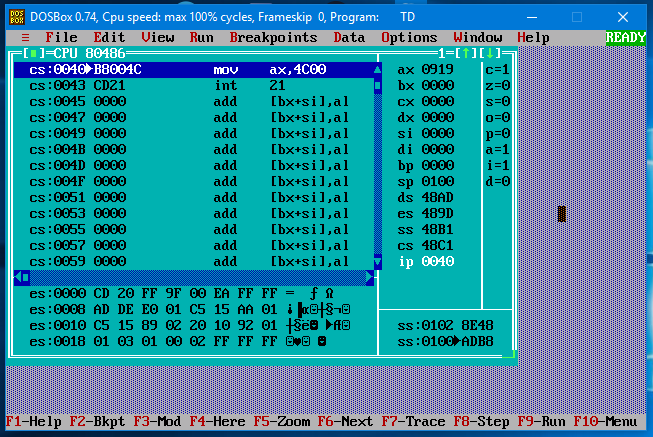


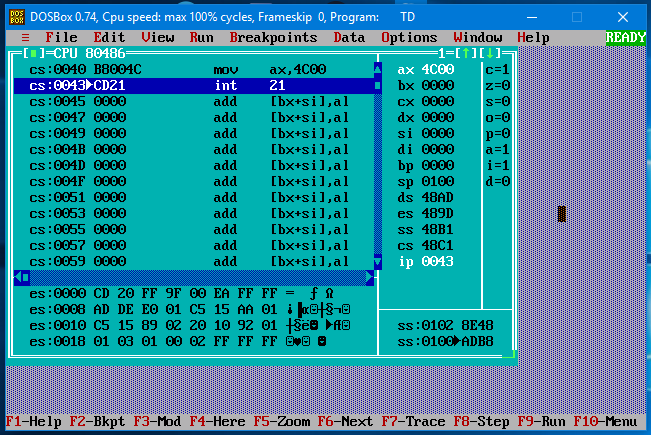


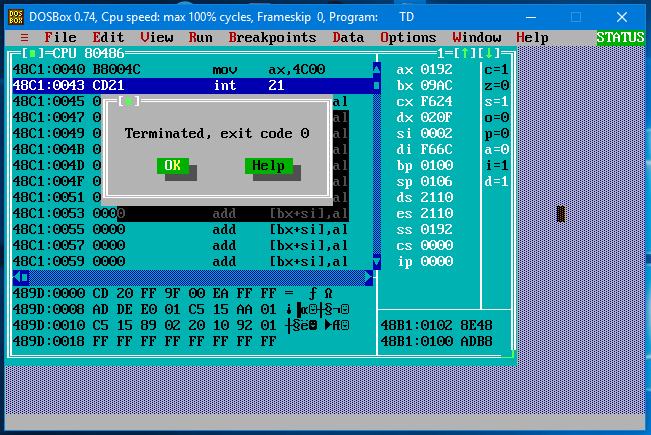
8) Переход на кінець коду



9) Блок передачі управління системі







**6. Висновок:** при створені коду с реалізацією розгалуженням потрібно спочатку правильно встановити умови та команду, яка відповідає умові заданої в задачі. В даній задачі я для власної зручності використовував команди умовного переходу протилежні для моєї умови, для того щоб при виконані умови програма не переходила на якусь мітку, а читалась поспіль команди.