

Отчёта по лабораторной работе 8

**Команды безусловного и условного переходов в Nasm.
Программирование ветвлений**

Мягмар Уржиндорж

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	21

Список иллюстраций

2.1	Файл lab8-1.asm:	7
2.2	Программа lab8-1.asm:	8
2.3	Файл lab8-1.asm:	9
2.4	Программа lab8-1.asm:	9
2.5	Файл lab8-1.asm	10
2.6	Программа lab8-1.asm	11
2.7	Файл lab8-2.asm	12
2.8	Программа lab8-2.asm	13
2.9	Файл листинга lab8-2	14
2.10	ошибка трансляции lab8-2	15
2.11	файл листинга с ошибкой lab8-2	16
2.12	Файл lab8-3.asm	17
2.13	Программа lab8-3.asm	18
2.14	Файл lab8-4.asm	19
2.15	Программа lab8-4.asm	20

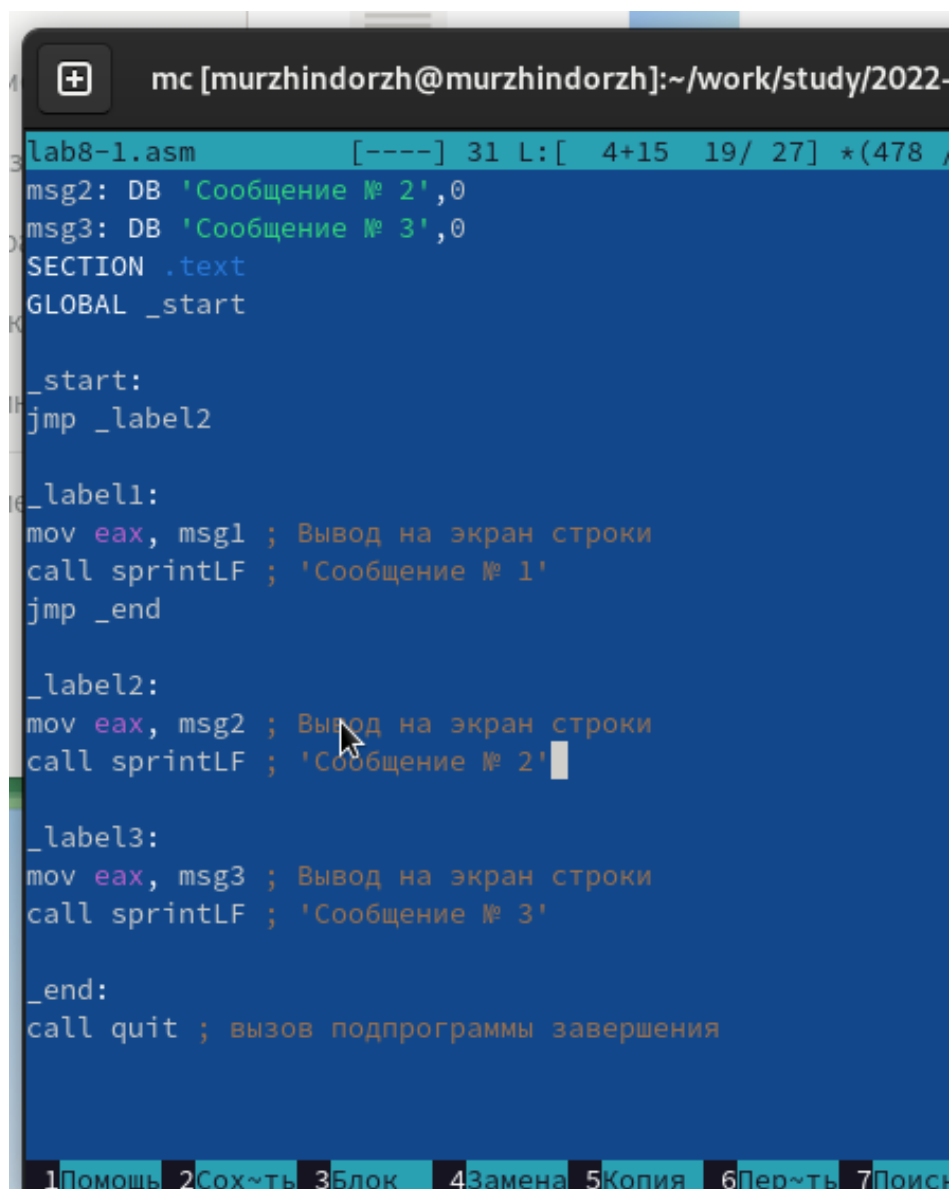
Список таблиц

1 Цель работы

Целью работы является изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

2 Выполнение лабораторной работы

1. Создайте каталог для программ лабораторной работы № 8, перейдите в него и создайте файл lab8-1.asm
2. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции jmp. Введите в файл lab8-1.asm текст программы из листинга 8.1. (рис. [2.1])



```
mc [murzhindorzh@murzhindorzh]:~/work/study/2022-  
lab8-1.asm [----] 31 L: [ 4+15 19/ 27] *(478  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL _start  
  
_start:  
jmp _label2  
  
_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintf ; 'Сообщение № 1'  
jmp _end  
  
_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintf ; 'Сообщение № 2'  
  
_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintf ; 'Сообщение № 3'  
  
_end:  
call quit ; вызов подпрограммы завершения
```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск

Рис. 2.1: Файл lab8-1.asm:

Создайте исполняемый файл и запустите его. (рис. [2.2])

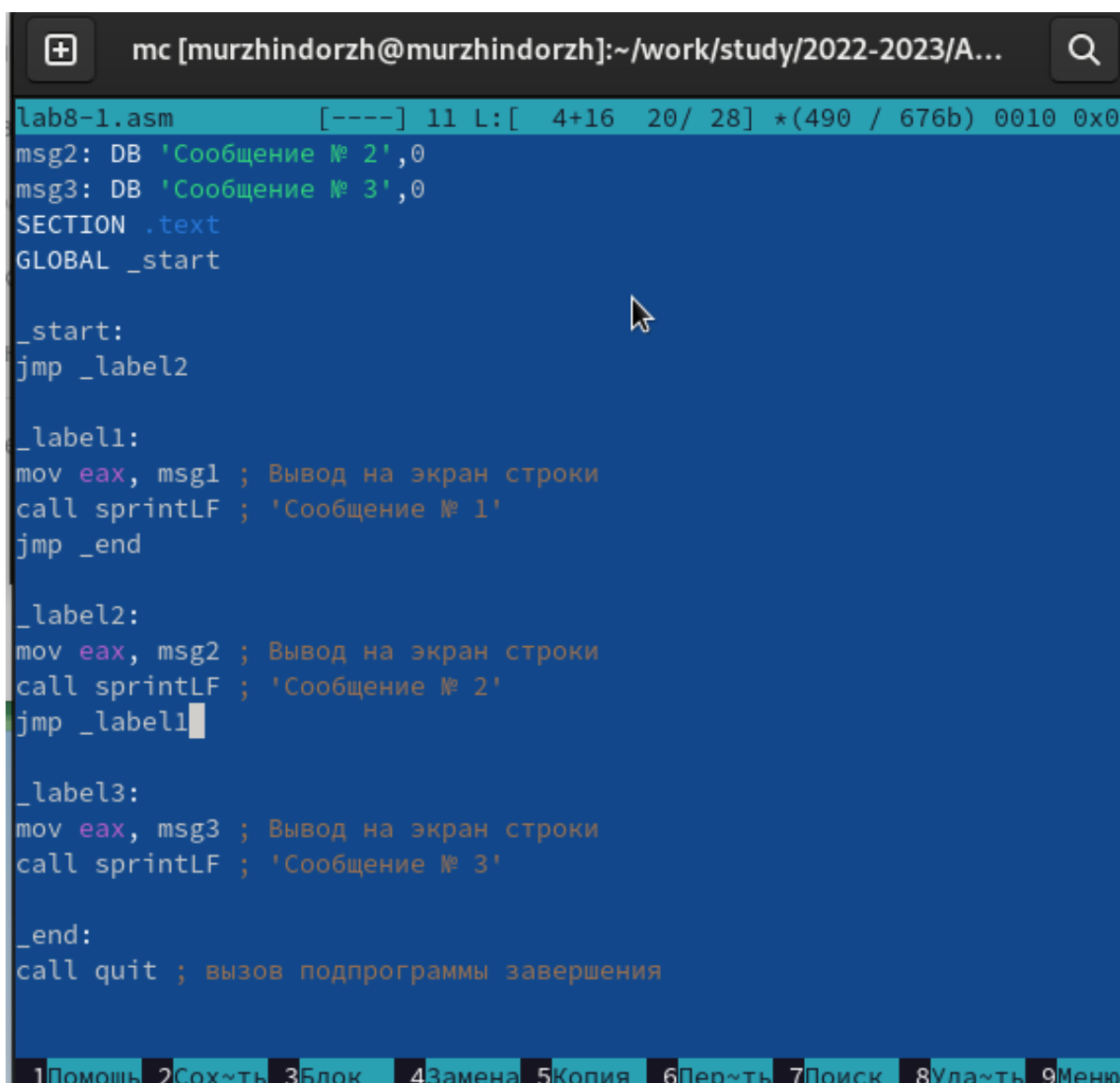
```

[murzhindorzh@murzhindorzh lab08]$
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-1.asm
[murzhindorzh@murzhindorzh lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
ld: не распознан режим эмуляции: elf
Поддерживаемые эмуляции: elf_x86_64 elf32_x86_64 elf_i386 elf_iamcu elf_llom elf
_klom i386pep i386pe elf64bpf
[murzhindorzh@murzhindorzh lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[murzhindorzh@murzhindorzh lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 3
[murzhindorzh@murzhindorzh lab08]$

```

Рис. 2.2: Программа lab8-1.asm:

Инструкция `jmp` позволяет осуществлять переходы не только вперед но и назад. Изменим программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию `jmp` с меткой `_label1` (т.е. переход к инструкциям вывода сообщения № 1) и после вывода сообщения № 1 добавим инструкцию `jmp` с меткой `_end` (т.е. переход к инструкции `call quit`). Измените текст программы в соответствии с листингом 8.2. (рис. [2.3], [2.4])



```
lab8-1.asm [----] 11 L: [ 4+16 20/ 28] *(490 / 676b) 0010 0x0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start

_start:
jmp _label2

_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
jmp _end

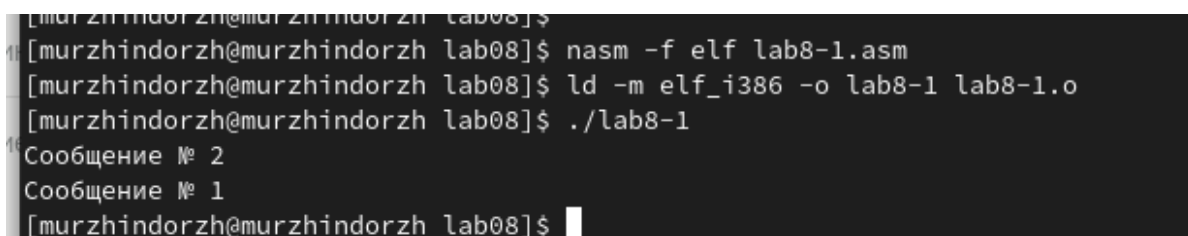
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'

_end:
call quit ; вызов подпрограммы завершения
```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить 9Меню

Рис. 2.3: Файл lab8-1.asm:



```
[murzhindorzh@murzhindorzh lab08]$
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-1.asm
[murzhindorzh@murzhindorzh lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[murzhindorzh@murzhindorzh lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 1
[murzhindorzh@murzhindorzh lab08]$
```

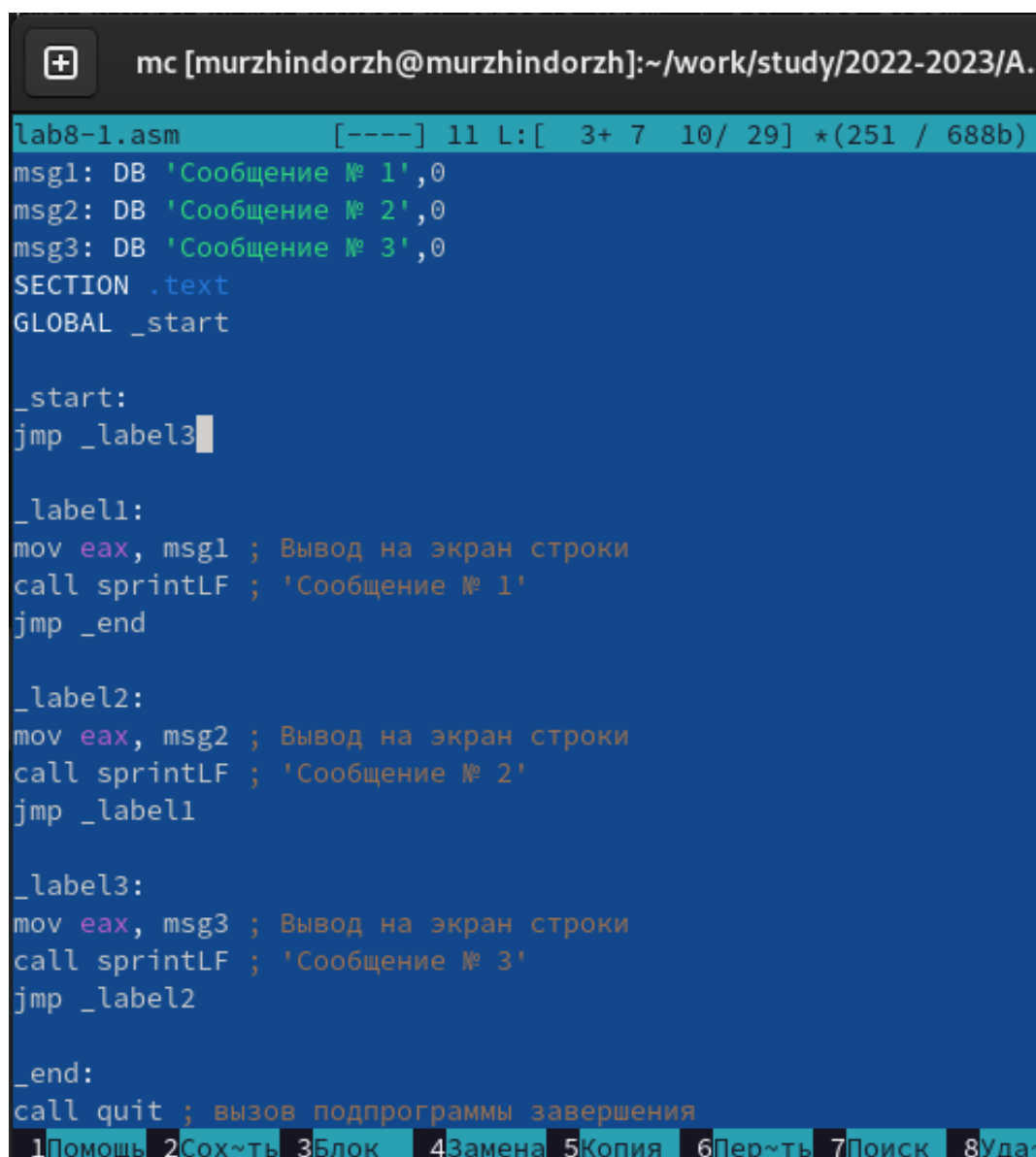
Рис. 2.4: Программа lab8-1.asm:

Измените текст программы добавив или изменив инструкции `jmp`, чтобы вывод программы был следующим (рис. [2.5], [2.6]):

Сообщение № 3

Сообщение № 2

Сообщение № 1



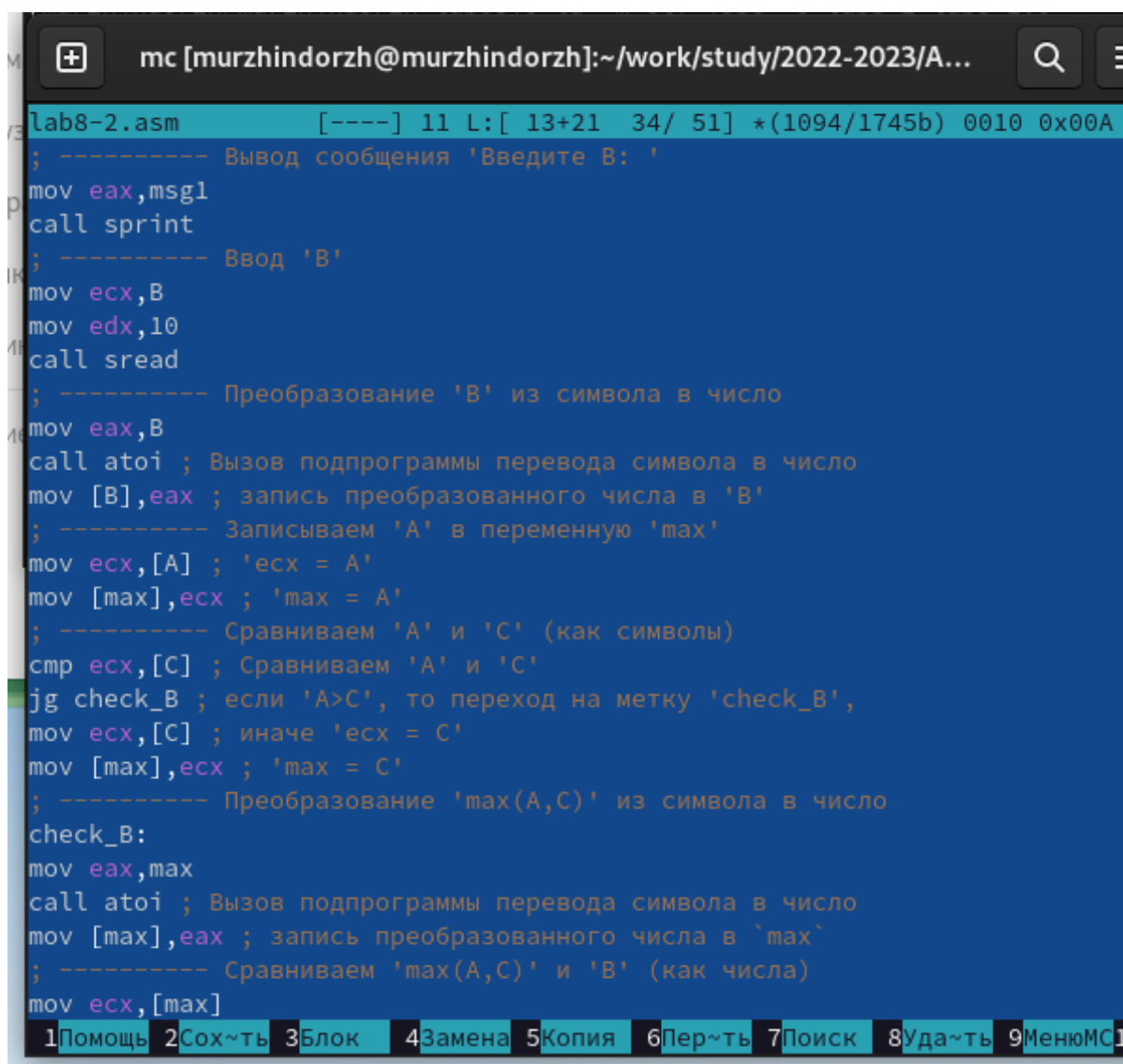
```
mc [murzhindorzh@murzhindorzh]:~/work/study/2022-2023/A.  
lab8-1.asm [----] 11 L:[ 3+ 7 10/ 29] *(251 / 688b)  
msg1: DB 'Сообщение № 1',0  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL _start  
  
_start:  
jmp _label3  
  
_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintf ; 'Сообщение № 1'  
jmp _end  
  
_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintf ; 'Сообщение № 2'  
jmp _label1  
  
_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintf ; 'Сообщение № 3'  
jmp _label2  
  
_end:  
call quit ; вызов подпрограммы завершения  
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить
```

Рис. 2.5: Файл lab8-1.asm

```
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-1.asm  
[murzhindorzh@murzhindorzh lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o  
[murzhindorzh@murzhindorzh lab08]$ ./lab8-1  
Сообщение № 3  
Сообщение № 2  
Сообщение № 1  
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$
```

Рис. 2.6: Программа lab8-1.asm

3. Использование инструкции `jmp` приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, т.е. переход должен происходить если выполнено какое-либо условие. В качестве примера рассмотрим программу, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: А, В и С. Значения для А и С задаются в программе, значение В вводится с клавиатуры. Создайте исполняемый файл и проверьте его работу для разных значений В. (рис. [2.7], [2.8])



```
lab8-2.asm [----] 11 L: [ 13+21 34/ 51] *(1094/1745b) 0010 0x00A
; ----- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
; ----- Ввод 'B'
mov ecx,B
mov edx,10
call sread
; ----- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
; ----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в 'max'
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
```

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить 9Меню

Рис. 2.7: Файл lab8-2.asm

```
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-2.asm  
[murzhindorzh@murzhindorzh lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o  
[murzhindorzh@murzhindorzh lab08]$ ./lab8-2  
Введите B: 140  
Наибольшее число: 140  
[murzhindorzh@murzhindorzh lab08]$ ./lab8-2  
Введите B: 10  
Наибольшее число: 50  
[murzhindorzh@murzhindorzh lab08]$
```

Рис. 2.8: Программа lab8-2.asm

4. Обычно nasm создаёт в результате ассемблирования только объектный файл. Получить файл листинга можно, указав ключ -l и задав имя файла листинга в командной строке. Создайте файл листинга для программы из файла lab8-2.asm (рис. [2.9])

```
lab8-2.lst [----] 0 L: [ 17+ 0 17/226] *(1075/14499b) 0032 0x020 [*] [X]
17 0000000E C3 <1> ret.....
18 <1> .
19 <1> .
20 <1> ;----- sprint -----
21 <1> ; Функция печати сообщения
22 <1> ; входные данные: mov eax,<message>
23 <1> sprint:
24 0000000F 52 <1> push edx
25 00000010 51 <1> push ecx
26 00000011 53 <1> push ebx
27 00000012 50 <1> push eax
28 00000013 E8E8FFFFFF <1> call slen
29 <1> ....
30 00000018 89C2 <1> mov edx, eax
31 0000001A 58 <1> pop eax
32 <1> ....
33 0000001B 89C1 <1> mov ecx, eax
34 0000001D BB01000000 <1> mov ebx, 1
35 00000022 B804000000 <1> mov eax, 4
36 00000027 CD80 <1> int 80h
37 <1> .
38 00000029 5B <1> pop ebx
39 0000002A 59 <1> pop ecx
40 0000002B 5A <1> pop edx
41 0000002C C3 <1> ret
42 <1> .
1Помощь 2Сох~ть 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюМС 10Выход
```

Рис. 2.9: Файл листинга lab8-2

Внимательно ознакомиться с его форматом и содержимым. Подробно объяснить содержимое трёх строк файла листинга по выбору.

строка 51

- 51 - номер строки
- 00000033 - адрес
- B80A000000 - машинный код
- mov eax, 0AH - код программы

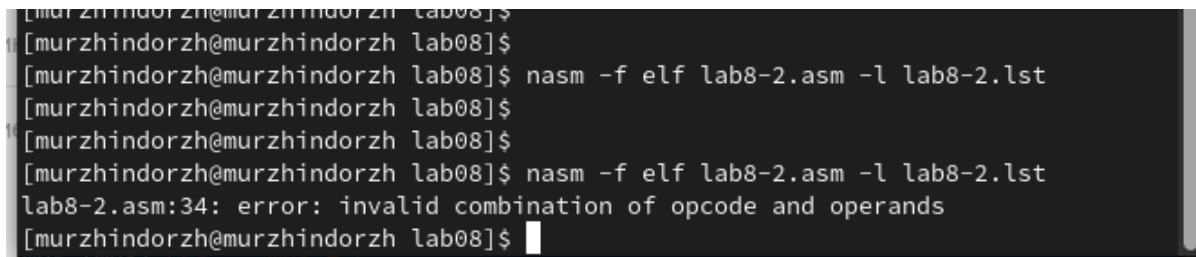
строка 52

- 52 - номер строки
- 00000038 - адрес
- 50 - машинный код
- push eax- код программы

строка 53

- 53 - номер строки
- 00000039 - адрес
- 89E0 - машинный код
- mov eax, esp - код программы

Откройте файл с программой lab8-2.asm и в любой инструкции с двумя операндами удалить один операнд. Выполните трансляцию с получением файла листинга (рис. [2.10],[2.11])



```
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-2.asm -l lab8-2.lst  
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-2.asm -l lab8-2.lst  
lab8-2.asm:34: error: invalid combination of opcode and operands  
[murzhindorzh@murzhindorzh lab08]$
```

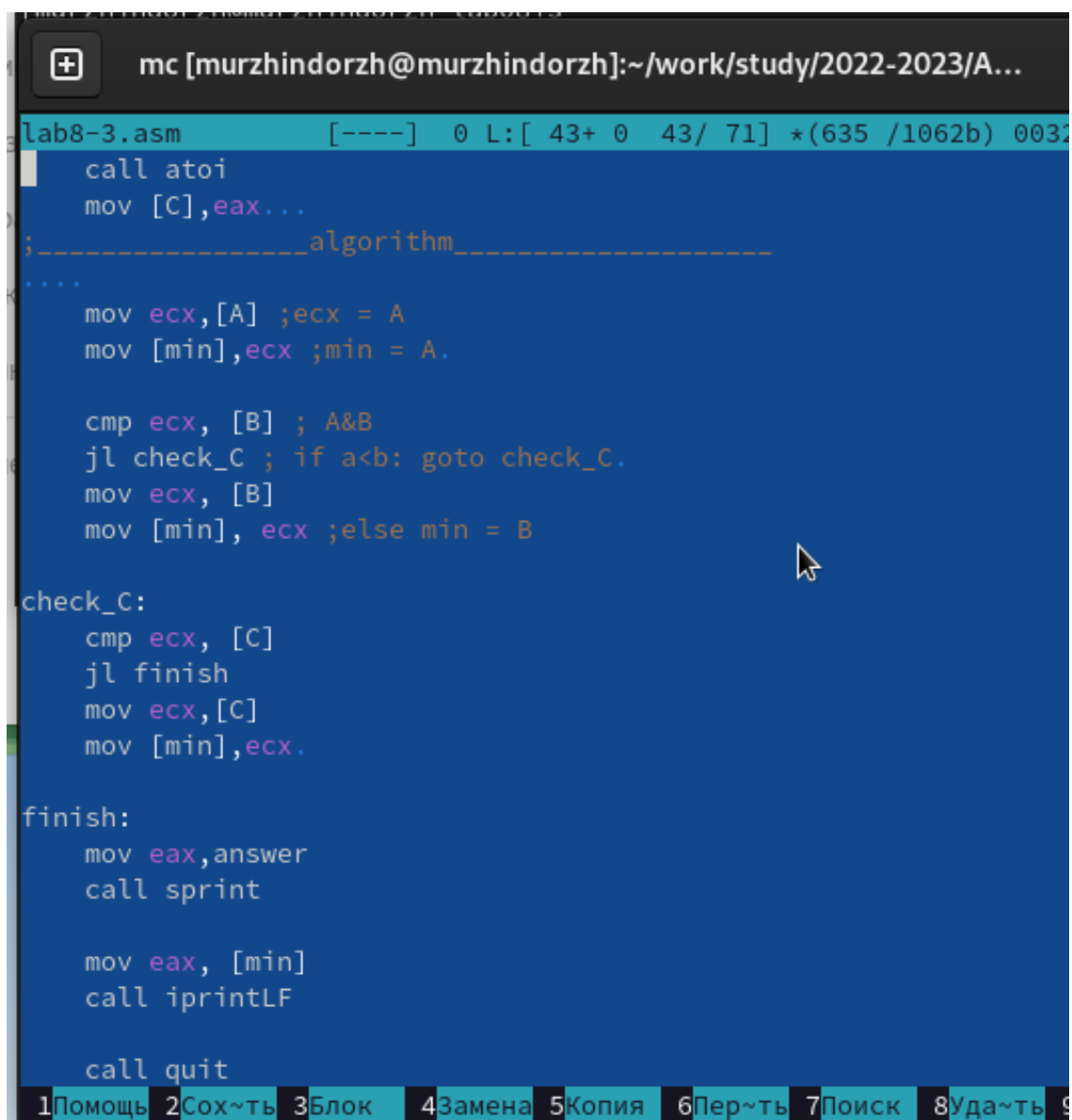
Рис. 2.10: ошибка трансляции lab8-2

```
mc [murzhindorzh@murzhindorzh]:~/work/study/2022-2023/A...
lab8-2.lst      [----]  0 L:[197+12 209/227] *(13157/14587b) 0032 0x020[*][X]
22 00000106 E891FFFFFF      call atoi ; Вызов подпрограммы перевода
23 0000010B A3[0A000000]     mov [B],eax ; запись преобразованного чи
24                               ; ----- Записываем 'A' в переменную
25 00000110 8B0D[35000000]     mov ecx,[A] ; 'ecx = A'
26 00000116 890D[00000000]     mov [max],ecx ; 'max = A'
27                               ; ----- Сравниваем 'A' и 'C' (как с
28 0000011C 3B0D[39000000]     cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 00000122 7F0C              jg check_B ; если 'A>C', то переход на м
30 00000124 8B0D[39000000]     mov ecx,[C] ; иначе 'ecx = C'
31 0000012A 890D[00000000]     mov [max],ecx ; 'max = C'
32                               ; ----- Преобразование 'max(A,C)' и
33                               check_B:
34                               mov eax,
34                               *****
35 00000130 E867FFFFFF      call atoi ; Вызов подпрограммы перевода
36 00000135 A3[00000000]     mov [max],eax ; запись преобразованного
37                               ; ----- Сравниваем 'max(A,C)' и 'B'
38 0000013A 8B0D[00000000]     mov ecx,[max]
39 00000140 3B0D[0A000000]     cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
40 00000146 7F0C              jg fin ; если 'max(A,C)>B', то переход н
41 00000148 8B0D[0A000000]     mov ecx,[B] ; иначе 'ecx = B'
42 0000014E 890D[00000000]     mov [max],ecx
43                               ; ----- Вывод результата
44                               fin:
45 00000154 B8[13000000]     mov eax, msg2
46 00000159 E8B1FEFFFF      call sprint ; Вывод сообщения 'Наибольше
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить 9МенюМС10Выход
```

Рис. 2.11: файл листинга с ошибкой lab8-2

5. Напишите программу нахождения наименьшей из 3 целочисленных переменных a, b и c. Значения переменных выбрать из табл. 8.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу (рис. [2.12],[2.13])

для варианта 7 - 45, 67, 15



```
mc [murzhindorzh@murzhindorzh]:~/work/study/2022-2023/A...
lab8-3.asm [----] 0 L: [ 43+ 0 43/ 71] *(635 /1062b) 0032
    call atoi
    mov [C],eax...
;-----algorithm-----
....
    mov ecx,[A] ;ecx = A
    mov [min],ecx ;min = A.

    cmp ecx, [B] ; A&B
    jl check_C ; if a<b: goto check_C.
    mov ecx, [B]
    mov [min], ecx ;else min = B

check_C:
    cmp ecx, [C]
    jl finish
    mov ecx,[C]
    mov [min],ecx.

finish:
    mov eax,answer
    call sprint

    mov eax, [min]
    call iprintLF

    call quit

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить 9
```

Рис. 2.12: Файл lab8-3.asm

```
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-3.asm  
[murzhindorzh@murzhindorzh lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o  
[murzhindorzh@murzhindorzh lab08]$ ./lab8-3  
Input A: 45  
Input B: 67  
Input C: 15  
Smallest: 15  
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$
```

Рис. 2.13: Программа lab8-3.asm

6. Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ выбрать из таблицы 8.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений x и a из 8.6. (рис. [2.14],[2.15])

для варианта 7

$$\begin{cases} 6a, x = a \\ a + x, x \neq a \end{cases}$$

```
mc [murzhindorzh@murzhindorzh]:~/work/st
lab8-4.asm [----] 11 L: [ 29+14 43/ 55]
    mov eax,X
    call atoi
    mov [X],eax...
;-----_algorithm-----

    mov ebx, [X]
    mov edx, [A]
    cmp ebx, edx
    je first
    jmp second

first:
    mov eax,[A]
    mov ebx,6
    mul ebx
    call iprintLF.
    call quit
    ....
second:
    mov eax,[X]
    mov ebx,[A]
    add eax, ebx
    call iprintLF.
    call quit

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти
```

Рис. 2.14: Файл lab8-4.asm

```
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$  
[murzhindorzh@murzhindorzh lab08]$ nasm -f elf lab8-4.asm  
[murzhindorzh@murzhindorzh lab08]$ ld -m elf_i386 -o lab8-4 lab8-4.o  
[murzhindorzh@murzhindorzh lab08]$ ./lab8-4  
Input A: 1  
Input X: 1  
6  
[murzhindorzh@murzhindorzh lab08]$ ./lab8-4  
Input A: 2  
Input X: 1  
3  
[murzhindorzh@murzhindorzh lab08]$
```

Рис. 2.15: Программа lab8-4.asm

3 Выводы

Изучили команды условного и безусловного переходов, познакомились с файлом листинга.