

Отчёта по лабораторной работе 8

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

Элина Майзингер НММбд-02-22

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	24
	Список литературы	25

Список иллюстраций

4.1	Файл lab8-1.asm:	9
4.2	Программа lab8-1.asm:	10
4.3	Файл lab8-1.asm:	11
4.4	Программа lab8-1.asm:	12
4.5	Файл lab8-1.asm	13
4.6	Программа lab8-1.asm	14
4.7	Файл lab8-2.asm	15
4.8	Программа lab8-2.asm	16
4.9	Файл листинга lab8-2	17
4.10	ошибка трансляции lab8-2	18
4.11	файл листинга с ошибкой lab8-2	19
4.12	Файл lab8-3.asm	20
4.13	Программа lab8-3.asm	21
4.14	Файл lab8-4.asm	22
4.15	Программа lab8-4.asm	23

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 Задание

1. Изучите примеры программ.
2. Изучите файл листинга.
3. 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
4. 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.

3 Теоретическое введение

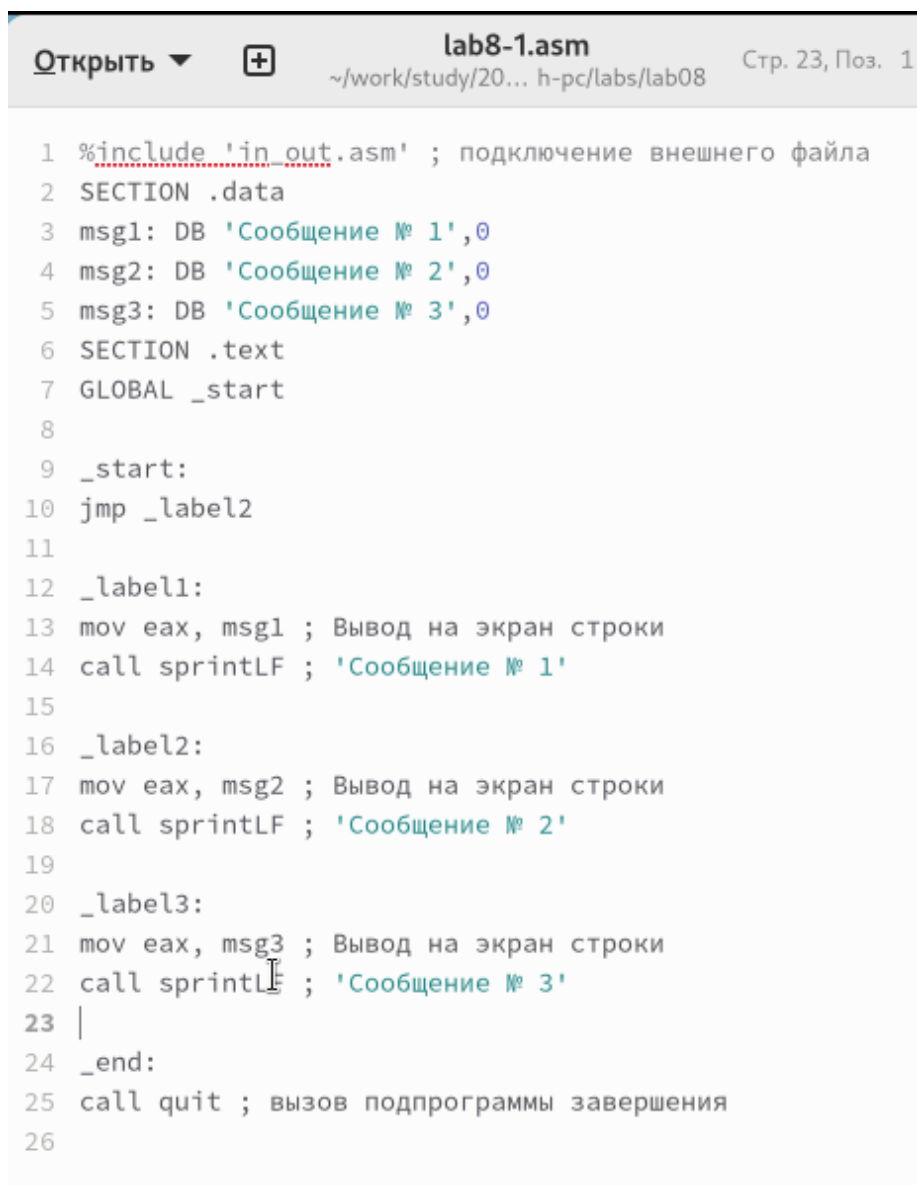
Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов:

- условный переход – выполнение или не выполнение перехода в определенную точку программы в зависимости от проверки условия.
- безусловный переход – выполнение передачи управления в определенную точку программы без каких-либо условий.

Листинг (в рамках понятийного аппарата NASM) — это один из выходных файлов, создаваемых транслятором. Он имеет текстовый вид и нужен при отладке программы, так как кроме строк самой программы он содержит дополнительную информацию.

4 Выполнение лабораторной работы

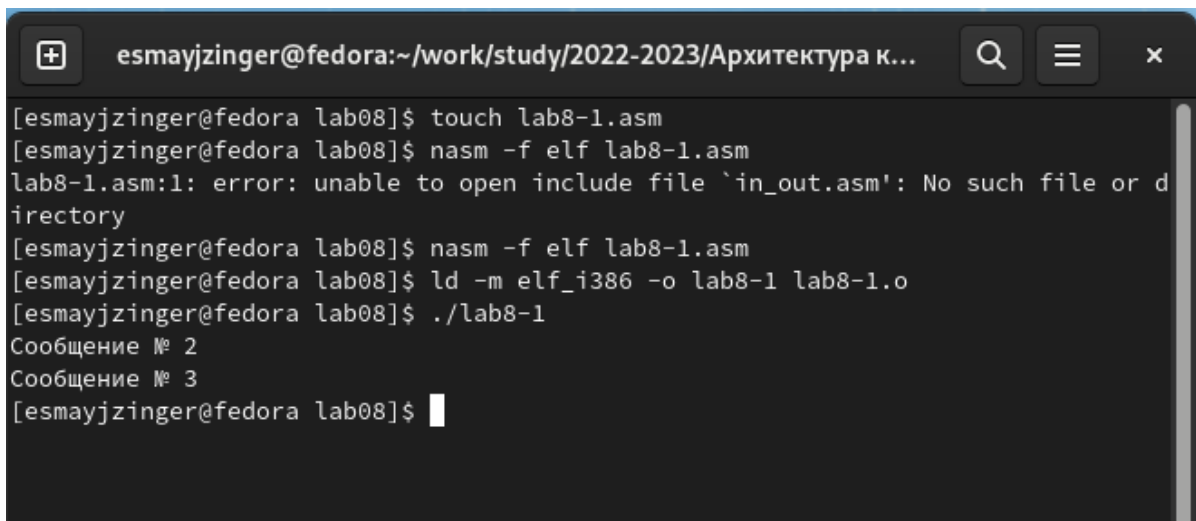
1. Создайте каталог для программ лабораторной работы № 8, перейдите в него и создайте файл `lab8-1.asm`
2. Инструкция `jmp` в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции `jmp`. Введите в файл `lab8-1.asm` текст программы из листинга 8.1. (рис. 4.1)



```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8
9 _start:
10 jmp _label2
11
12 _label1:
13 mov eax, msg1 ; Вывод на экран строки
14 call sprintLF ; 'Сообщение № 1'
15
16 _label2:
17 mov eax, msg2 ; Вывод на экран строки
18 call sprintLF ; 'Сообщение № 2'
19
20 _label3:
21 mov eax, msg3 ; Вывод на экран строки
22 call sprintLF ; 'Сообщение № 3'
23 |
24 _end:
25 call quit ; вызов подпрограммы завершения
26
```

Рис. 4.1: Файл lab8-1.asm:

Создайте исполняемый файл и запустите его. (рис. 4.2)

A terminal window with a dark background and light text. The title bar shows the user 'esmayjzinger@fedora' and the path '~/work/study/2022-2023/Архитектура к...'. The terminal contains the following text:

```
[esmayjzinger@fedora lab08]$ touch lab8-1.asm
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
lab8-1.asm:1: error: unable to open include file `in_out.asm': No such file or d
irectory
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[esmayjzinger@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 3
[esmayjzinger@fedora lab08]$
```

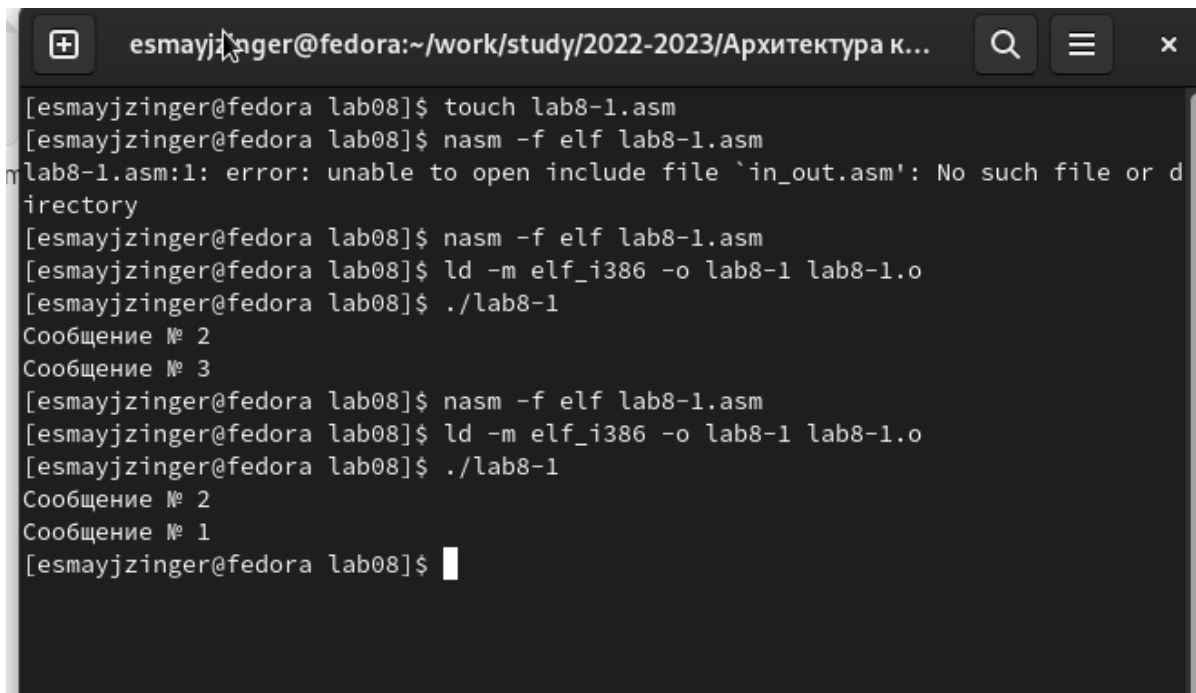
Рис. 4.2: Программа lab8-1.asm:

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. 4.3, 4.4)



```
1  %include 'in_out.asm' ; подключение внешнего файла
2  SECTION .data
3  msg1: DB 'Сообщение № 1',0
4  msg2: DB 'Сообщение № 2',0
5  msg3: DB 'Сообщение № 3',0
6  SECTION .text
7  GLOBAL _start
8  _start:
9  jmp _label2
10
11 _label1:
12 mov eax, msg1 ; Вывод на экран строки
13 call sprintf ; 'Сообщение № 1'
14 jmp _end
15
16 _label2:
17 mov eax, msg2 ; Вывод на экран строки
18 call sprintf ; 'Сообщение № 2'
19 jmp _label1
20
21 _label3:
22 mov eax, msg3 ; Вывод на экран строки
23 call sprintf ; 'Сообщение № 3'
24 |
25 _end:
26 call quit ; вызов подпрограммы завершения
```

Рис. 4.3: Файл lab8-1.asm:

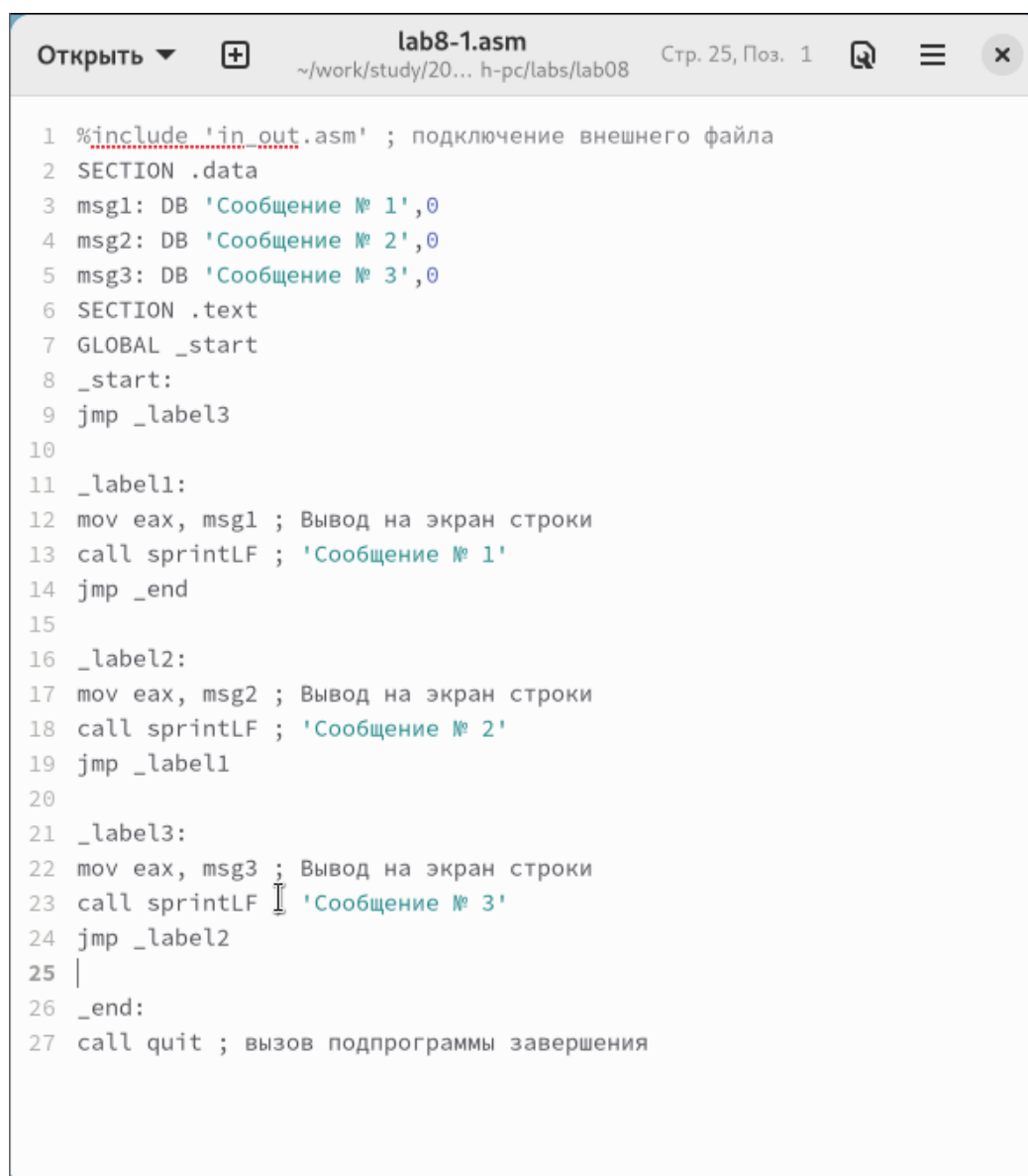
A terminal window with a dark background. The title bar shows the user 'esmayjzinger' on a 'fedora' machine, in the directory '~/work/study/2022-2023/Архитектура к...'. The terminal contains the following commands and output:

```
[esmayjzinger@fedora lab08]$ touch lab8-1.asm
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
lab8-1.asm:1: error: unable to open include file `in_out.asm': No such file or d
irectory
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[esmayjzinger@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 3
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[esmayjzinger@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 1
[esmayjzinger@fedora lab08]$
```

Рис. 4.4: Программа lab8-1.asm:

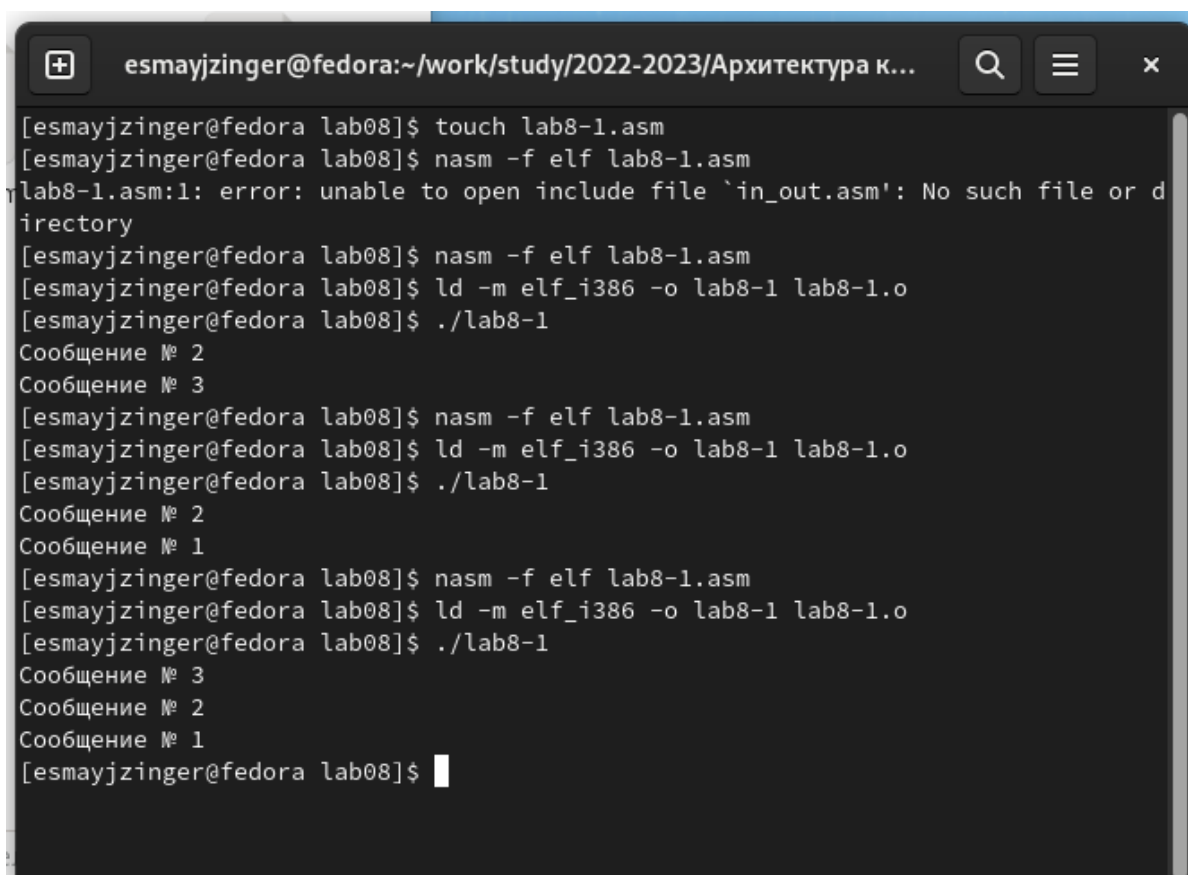
Измените текст программы добавив или изменив инструкции `jmp`, чтобы вывод программы был следующим (рис. 4.5, 4.6):

Сообщение № 3
Сообщение № 2
Сообщение № 1



```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label3
10
11 _label1:
12 mov eax, msg1 ; Вывод на экран строки
13 call sprintfLF ; 'Сообщение № 1'
14 jmp _end
15
16 _label2:
17 mov eax, msg2 ; Вывод на экран строки
18 call sprintfLF ; 'Сообщение № 2'
19 jmp _label1
20
21 _label3:
22 mov eax, msg3 ; Вывод на экран строки
23 call sprintfLF ; 'Сообщение № 3'
24 jmp _label2
25 |
26 _end:
27 call quit ; вызов подпрограммы завершения
```

Рис. 4.5: Файл lab8-1.asm



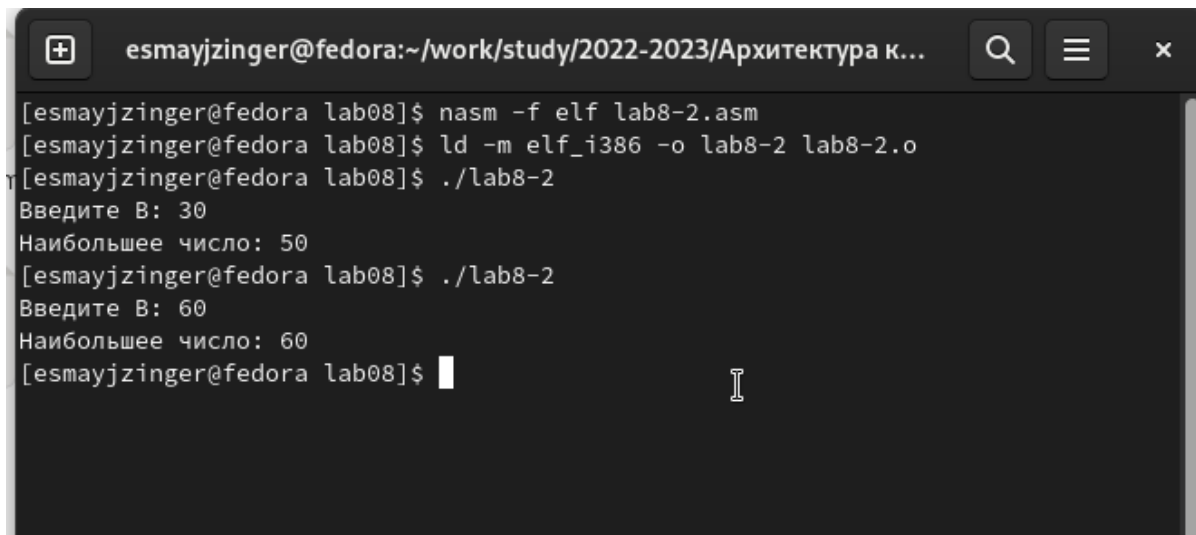
```
[esmayjzinger@fedora lab08]$ touch lab8-1.asm
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
lab8-1.asm:1: error: unable to open include file `in_out.asm': No such file or d
irectory
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[esmayjzinger@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 3
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[esmayjzinger@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 1
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-1.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[esmayjzinger@fedora lab08]$ ./lab8-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
[esmayjzinger@fedora lab08]$
```

Рис. 4.6: 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. 4.7, 4.8)

```
1 %include 'in_out.asm'
2 section .data
3 msg1 db 'Введите B: ',0h
4 msg2 db "Наибольшее число: ",0h
5 A dd '20'
6 C dd '50'
7 section .bss
8 max resb 10
9 B resb 10
10 section .text
11 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 mov eax,msg1
15 call sprint
16 ; ----- Ввод 'B'
17 mov ecx,B
18 mov edx,10
19 call sread
20 ; ----- Преобразование 'B' из символа в число
21 mov eax,B
22 call atoi ; Вызов подпрограммы перевода символа в число
23 mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
25 mov ecx,[A] ; 'ecx = A'
26 mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 ; check B : если 'A>C', то переход на метку 'check B'.
```

Рис. 4.7: Файл lab8-2.asm

A terminal window with a dark background. The title bar shows the user 'esmayjzinger@fedora' and the current directory '~/work/study/2022-2023/Архитектура к...'. The terminal contains the following text:

```
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-2.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[esmayjzinger@fedora lab08]$ ./lab8-2
Введите В: 30
Наибольшее число: 50
[esmayjzinger@fedora lab08]$ ./lab8-2
Введите В: 60
Наибольшее число: 60
[esmayjzinger@fedora lab08]$
```

Рис. 4.8: 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. 4.9)


```

4      4      <1> slen:
5      5 00000000 53      <1> push ebx
6      6 00000001 89C3      <1> mov ebx, eax
7      7      <1>
8      8      <1> nextchar:
9      9 00000003 803800      <1> cmp byte [eax], 0
10     10 00000006 7403      <1> jz finished
11     11 00000008 40      <1> inc eax
12     12 00000009 FBF8      <1> jmp nextchar
13     13      <1>
14     14      <1> finished:
15     15 0000000B 29D8      <1> sub eax, ebx
16     16 0000000D 5B      <1> pop ebx
17     17 0000000E C3      <1> ret
18     18      <1>
19     19      <1>
20     20      <1> ;----- printf -----
21     21      <1> ; Функция печати сообщения
22     22      <1> ; входные данные: mov eax, <message>
23     23      <1> printf:
24     24 0000000F 52      <1> push edx
25     25 00000010 51      <1> push ecx
26     26 00000011 53      <1> push ebx
27     27 00000012 50      <1> push eax
28     28 00000013 E8E8FFFFFF      <1> call slen
29     29      <1>
30     30 00000018 89C2      <1> mov edx, eax
31     31 0000001A 58      <1> pop eax
32     32      <1>

```

Рис. 4.9: Файл листинга lab8-2

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

строка 24

- 24 - номер строки
- 0000000F - адрес
- 52 - машинный код
- push edx - код программы

строка 25

- 25 - номер строки
- 00000010 - адрес

- 51 - машинный код
- push ecx - код программы

строка 28

- 28 - номер строки
- 00000013 - адрес
- E8E8FFFFFF - машинный код
- call slen - 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. 4.10,4.11)

```

esmayjzinger@fedora:~/work/study/2022-2023/Архитектура к...
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-2.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[esmayjzinger@fedora lab08]$ ./lab8-2
Введите B: 30
Наибольшее число: 50
[esmayjzinger@fedora lab08]$ ./lab8-2
Введите B: 60
Наибольшее число: 60
[esmayjzinger@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
[esmayjzinger@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:34: error: invalid combination of opcode and operands
[esmayjzinger@fedora lab08]$
[esmayjzinger@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:34: error: invalid combination of opcode and operands
[esmayjzinger@fedora lab08]$

```

Рис. 4.10: ошибка трансляции lab8-2

```

lab8-2.lst
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08
Стр. 210, Поз. 1

lab8-1.asm      lab8-2.asm      lab8-2.lst
197 22 00000100 8B0D[00000000]      call atoi ; Вызов подпрограммы перевода символа в число
198 23 0000010B A3[0A000000]      mov [B],eax ; запись преобразованного числа в 'B'
199 24                                ; ----- Записываем 'A' в переменную 'max'
200 25 00000110 8B0D[35000000]      mov ecx,[A] ; ecx = A
201 26 00000116 890D[00000000]      mov [max],ecx ; 'max' = A
202 27                                ; ----- Сравниваем 'A' и 'C' (как символы)
203 28 0000011C 3B0D[39000000]      cmp ecx,[C] ; Сравниваем 'A' и 'C'
204 29 00000122 7F0C      jg check_B ; если 'A'>'C', то переход на метку 'check_B',
205 30 00000124 8B0D[39000000]      mov ecx,[C] ; иначе 'ecx' = C
206 31 0000012A 890D[00000000]      mov [max],ecx ; 'max' = C
207 32                                ; ----- Преобразование 'max(A,C)' из символа в число
208 33      check_B:
209 34                                mov eax,
210 34 ***** error: invalid combination of opcode and operands
211 35 00000130 E867FFFFFF      call atoi ; Вызов подпрограммы перевода символа в число
212 36 00000135 A3[00000000]      mov [max],eax ; запись преобразованного числа в 'max'
213 37                                ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
214 38 0000013A 8B0D[00000000]      mov ecx,[max]
215 39 00000140 3B0D[0A000000]      cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
216 40 00000146 7F0C      jg fin ; если 'max(A,C)>'B', то переход на 'fin',
217 41 00000148 8B0D[0A000000]      mov ecx,[B] ; иначе 'ecx' = B
218 42 0000014E 890D[00000000]      mov [max],ecx
219 43                                ; ----- Вывод результата
220 44      fin:
221 45 00000154 B8[13000000]      mov eax,msg2
222 46 00000159 E8B1FFFFFF      call sprintf ; Вывод сообщения 'Наибольшее число: '
223 47 0000015E A1[00000000]      mov eax,[max]
224 48 00000163 E81EFFFFFF      call iprintf ; Вывод 'max(A,B,C)'
225 49 00000168 E86EFFFFFF      call quit ; Выход

```

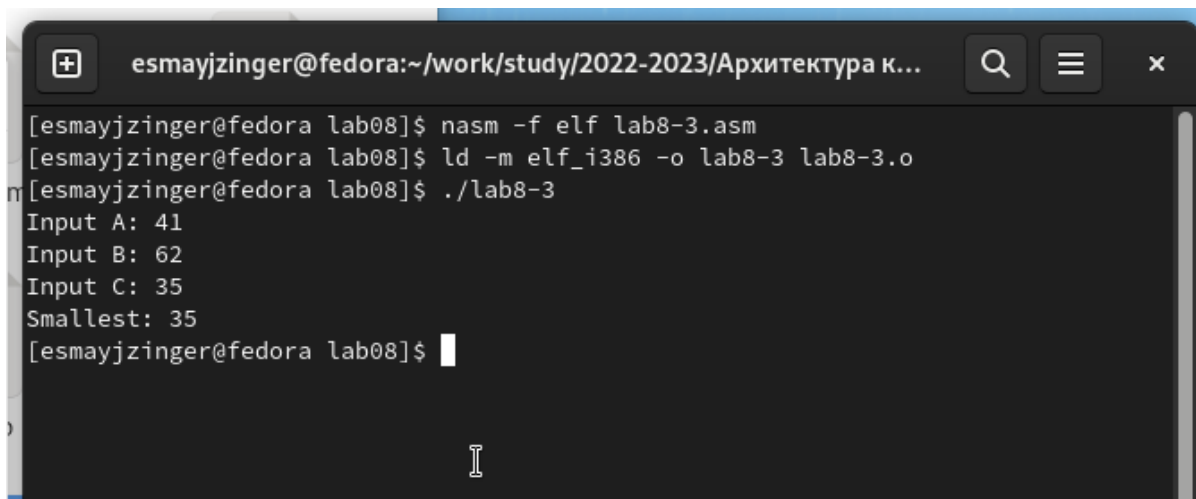
Рис. 4.11: файл листинга с ошибкой lab8-2

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. 4.12,4.13)

для варианта 10 - 41,62,35

```
11 C: RESB 80
12 result: RESB 80
13 min: RESB 80
14
15 SECTION .text
16 GLOBAL _start
17
18 _start:
19     mov eax,msgA
20     call sprint
21     mov ecx,A
22     mov edx,80
23     call sread
24     mov eax,A
25     call atoi
26     mov [A],eax
27
28     mov eax, msgB
29     call sprint
30     mov ecx,B
31     mov edx,80
32     call sread
33     mov eax,B
34     call atoi
35     mov [B],eax
36
37     mov eax,msgC
38     call sprint
39     mov ecx,C
40     mov edx,80
41     call sread
42     mov eax,C
```

Рис. 4.12: Файл lab8-3.asm



```
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-3.asm
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o
[esmayjzinger@fedora lab08]$ ./lab8-3
Input A: 41
Input B: 62
Input C: 35
Smallest: 35
[esmayjzinger@fedora lab08]$
```

Рис. 4.13: 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. 4.14,4.15)

для варианта 10

$$\begin{cases} x - 2, & x > 2 \\ 3a, & x \leq 2 \end{cases}$$

```
Открыть ▾ + lab8-4.asm Стр. 41, Поз. 14
~/work/study/20... h-pc/labs/lab08
21      call atoi
22      mov [A],eax
23
24      mov eax,msgX
25      call sprint
26      mov ecx,X
27      mov edx,80
28      call sread
29      mov eax,X
30      call atoi
31      mov [X],eax
32      ;-----algorithm-----
33
34      mov ebx, [X]
35      cmp ebx, 2
36      ja first
37      jmp second
38
39 first:
40      mov eax,[X]
41      sub eax,2
42      call iprintLF
43      call quit
44 second:
45      mov eax, [A]
46      mov ebx,3
47      mul ebx
48      call iprintLF
49      call quit
50
51
```

Рис. 4.14: Файл lab8-4.asm

```
[esmayjzinger@fedora lab08]$  
[esmayjzinger@fedora lab08]$  
[esmayjzinger@fedora lab08]$ nasm -f elf lab8-4.asm  
[esmayjzinger@fedora lab08]$ ld -m elf_i386 -o lab8-4 lab8-4.o  
[esmayjzinger@fedora lab08]$ ./lab8-4  
Input A: 0  
Input X: 3  
1  
[esmayjzinger@fedora lab08]$ ./lab8-4  
Input A: 2  
Input X: 1  
6  
[esmayjzinger@fedora lab08]$
```

Рис. 4.15: Программа lab8-4.asm

5 Выводы

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

Список литературы

1. Расширенный ассемблер: NASM
2. MASM, TASM, FASM, NASM под Windows и Linux