

Отчёта по лабораторной работе 8

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

Цвелев С.А. НПИбд-02-22

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	22

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:	10
2.5	Файл lab8-1.asm	11
2.6	Программа lab8-1.asm	12
2.7	Файл lab8-2.asm	13
2.8	Программа lab8-2.asm	14
2.9	Файл листинга lab8-2	15
2.10	ошибка трансляции lab8-2	16
2.11	файл листинга с ошибкой lab8-2	17
2.12	Файл lab8-3.asm	18
2.13	Программа lab8-3.asm	18
2.14	Файл lab8-4.asm	20
2.15	Программа lab8-4.asm	21

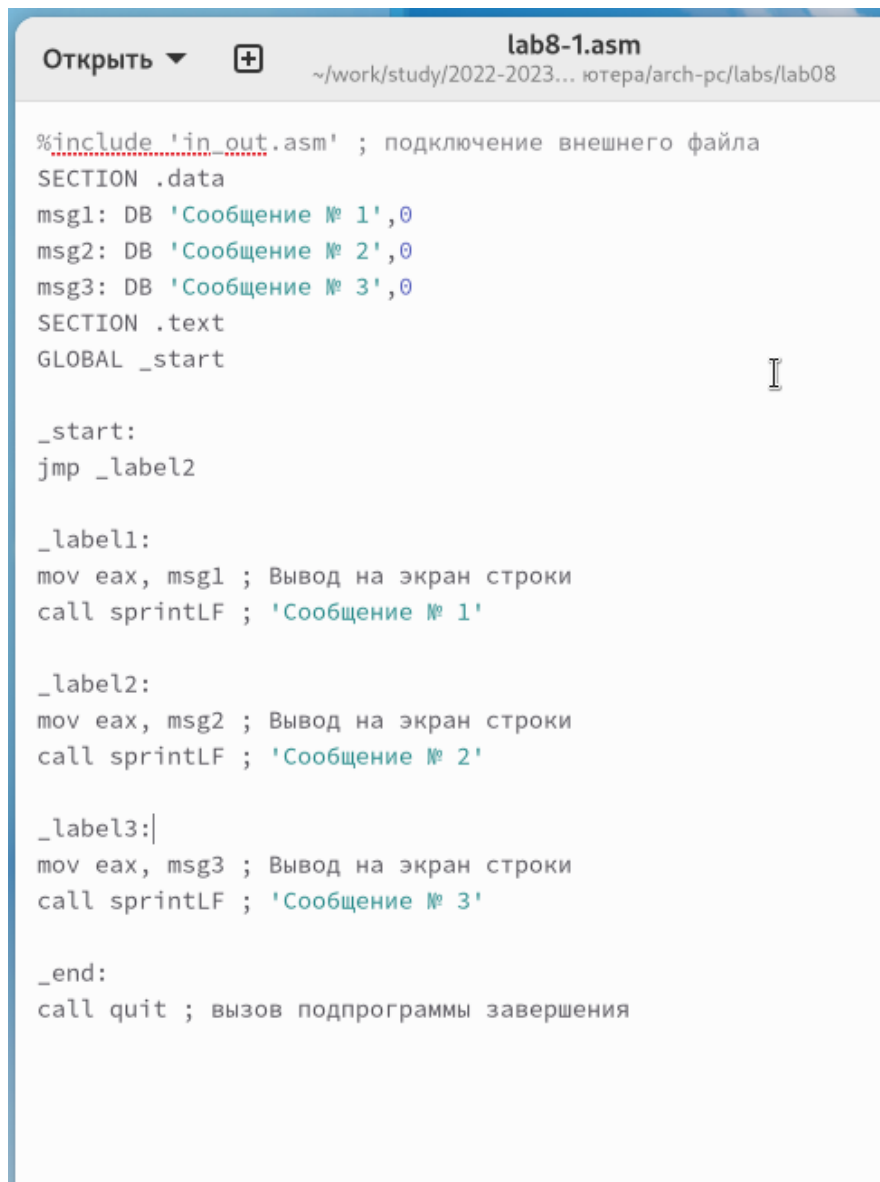
Список таблиц

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)



```
Открыть ▾ + lab8-1.asm
~/work/study/2022-2023... ютера/arch-pc/labs/lab08

%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
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 sprintfLF ; 'Сообщение № 1'

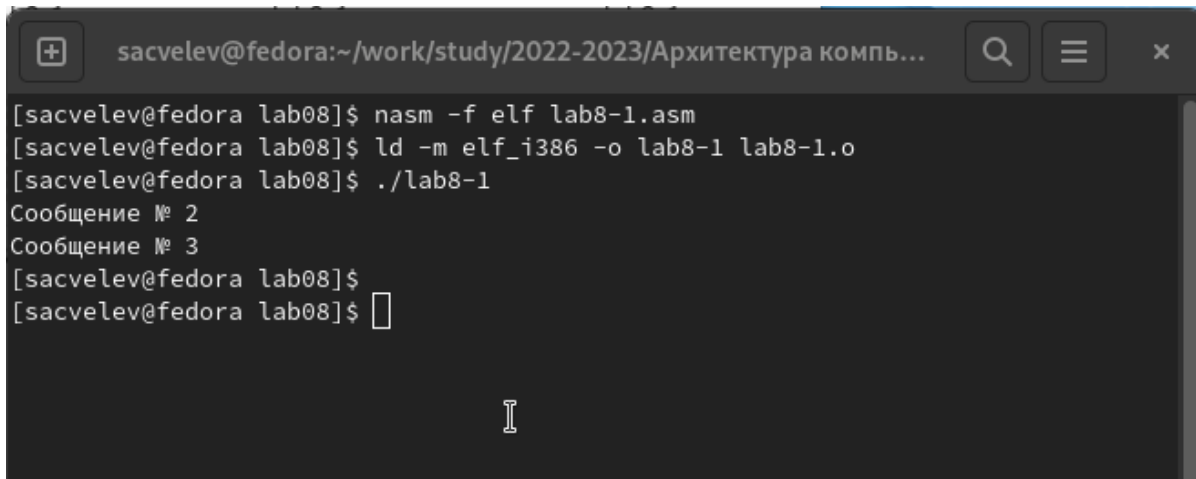
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 2'

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

_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.1: Файл lab8-1.asm:

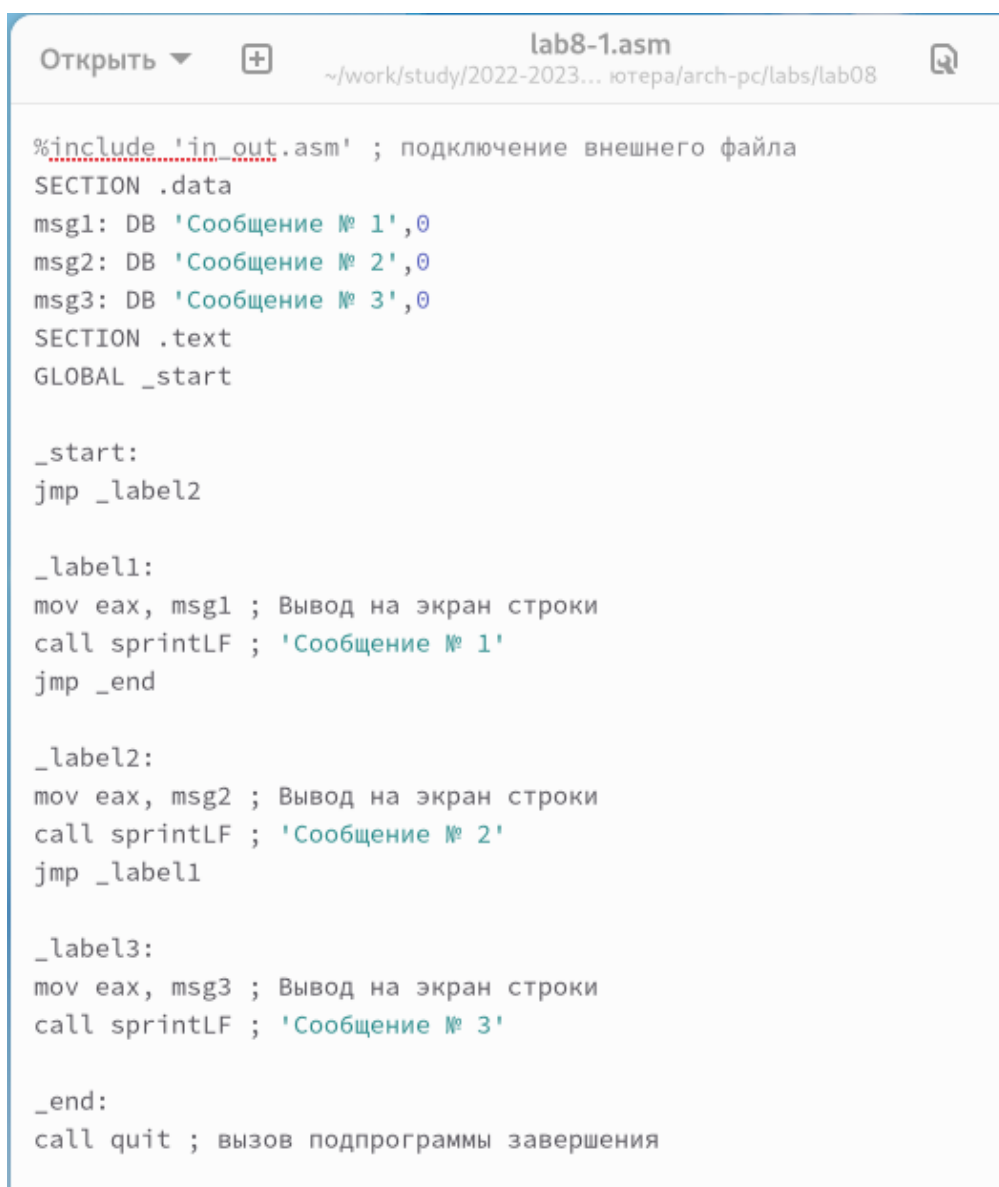
Создайте исполняемый файл и запустите его. (рис. 2.2)

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

```
[sacvelev@fedora lab08]$ nasm -f elf lab8-1.asm
[sacvelev@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[sacvelev@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 3
[sacvelev@fedora lab08]$
[sacvelev@fedora 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
~/work/study/2022-2023... ютеpa/arch-pc/labs/lab08

%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start

_start:
jmp _label2

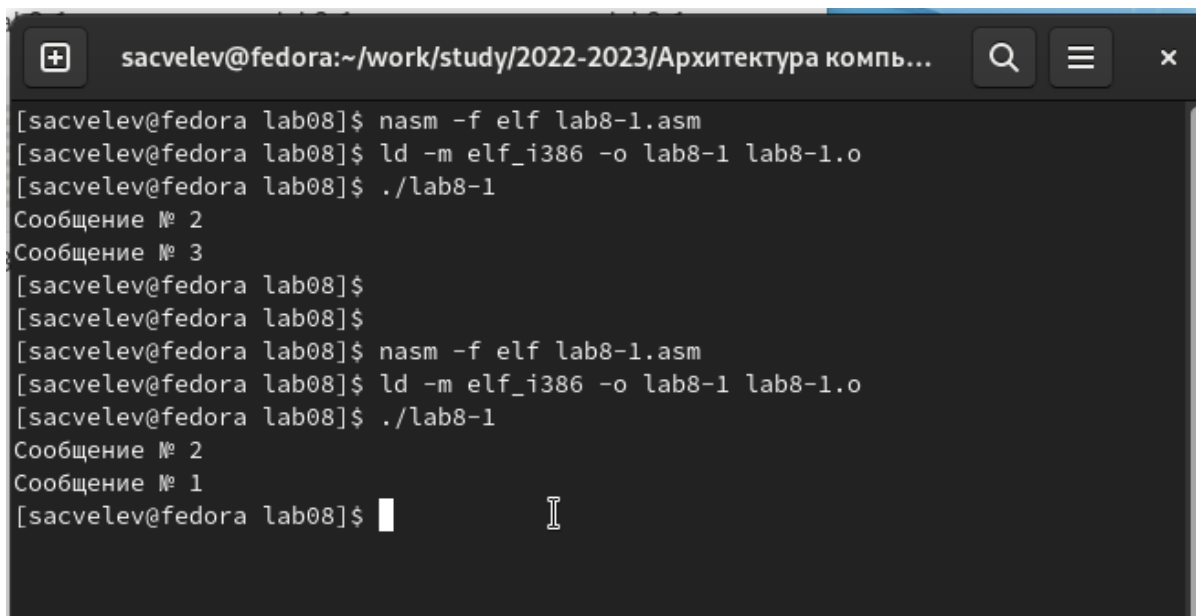
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
jmp _end

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

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

_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.3: Файл lab8-1.asm:

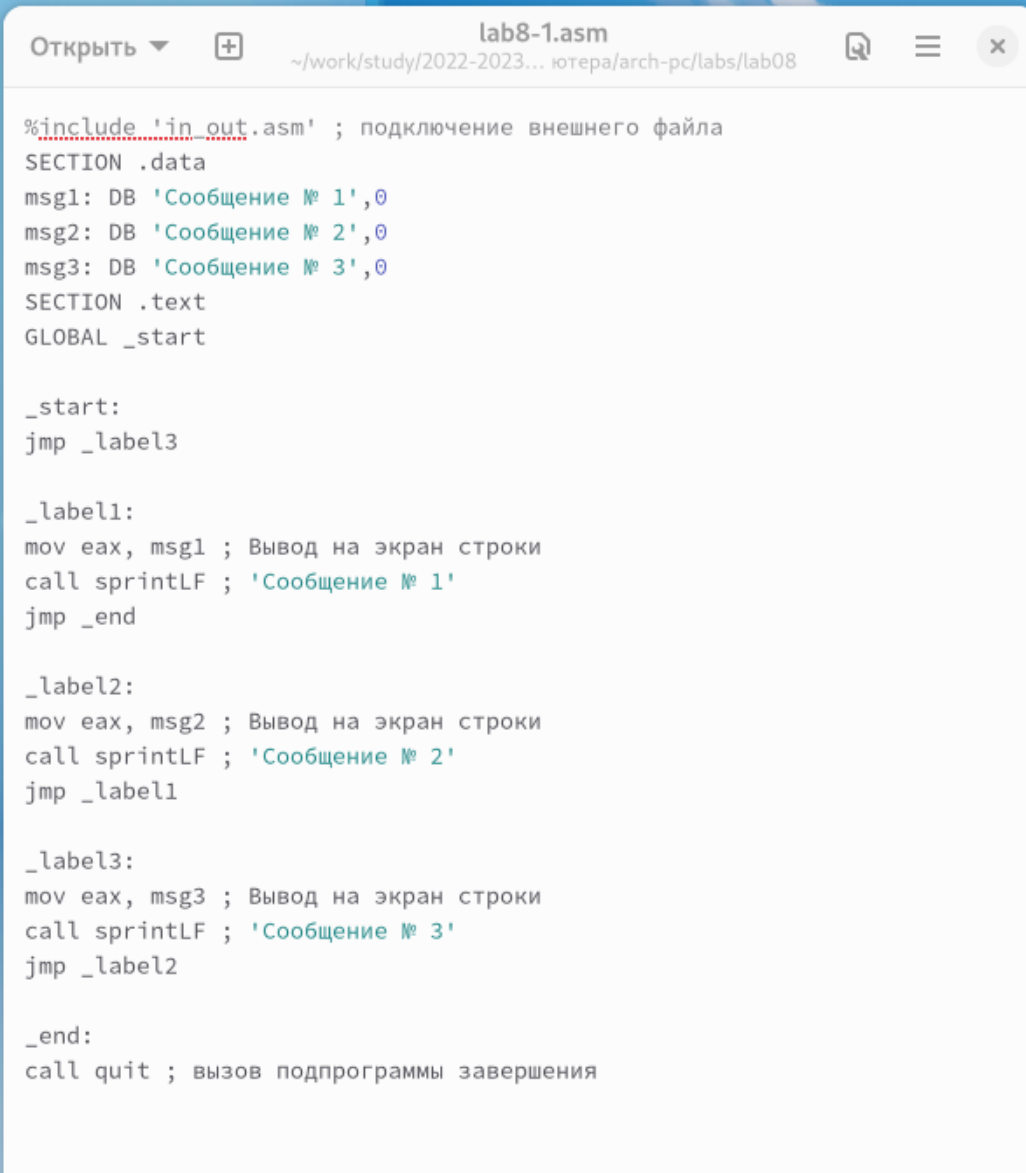
A terminal window titled 'sacvelev@fedora:~/work/study/2022-2023/Архитектура компь...' with search, menu, and close icons. The terminal shows the following commands and output:

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



```
Открыть ▾ + lab8-1.asm
~/work/study/2022-2023... ютеpa/arch-pc/labs/lab08

%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
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

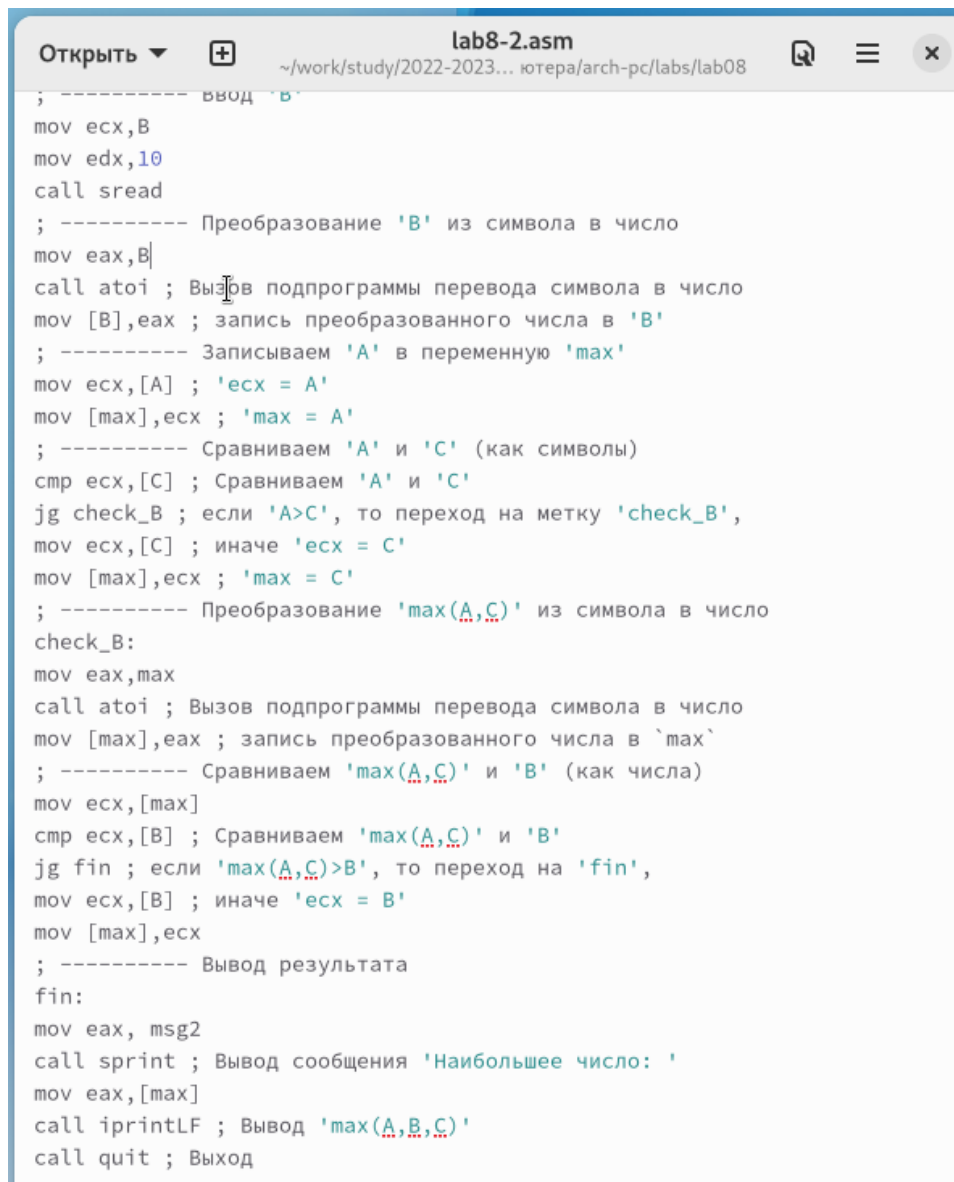
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.5: Файл lab8-1.asm

A screenshot of a terminal window with a dark background. The window title is "sacvelev@fedora:~/work/study/2022-2023/Архитектура компь...". The terminal shows a series of commands and their outputs. The commands are: `nasm -f elf lab8-1.asm`, `ld -m elf_i386 -o lab8-1 lab8-1.o`, and `./lab8-1`. The outputs are: "Сообщение № 2", "Сообщение № 3", an empty line, another empty line, "Сообщение № 2", "Сообщение № 1", "Сообщение № 3", "Сообщение № 2", and "Сообщение № 1". The cursor is at the end of the last command line.

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)



```
Открыть ▾ + lab8-2.asm
~/work/study/2022-2023... ютера/arch-pc/labs/lab08

; ----- 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]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jg fin ; если 'max(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [max],ecx
; ----- Вывод результата
fin:
mov eax,msg2
call sprint ; Вывод сообщения 'Наибольшее число: '
mov eax,[max]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход
```

Рис. 2.7: Файл lab8-2.asm

```
[sacvelev@fedora lab08]$ nasm -f elf lab8-2.asm
[sacvelev@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[sacvelev@fedora lab08]$ ./lab8-2
Введите В: 60
Наибольшее число: 60
[sacvelev@fedora lab08]$ ./lab8-2
Введите В: 50
Наибольшее число: 50
[sacvelev@fedora lab08]$ ./lab8-2
Введите В: 40
Наибольшее число: 50
[sacvelev@fedora 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)

```

Открыть  + lab8-2.lst
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08

170 000000E5 C080 <1> int 80h
171 000000E7 C3 <1> ret
2 section .data
3 00000000 D092D0B2D0B5D0B4D0- msg1 db 'Введите B: ',0h
3 00000009 B8D182D0B520423A20-
3 00000012 00
4 00000013 D09DD0B0D0B8D0B1D0- msg2 db "Наибольшее число: ",0h
4 0000001C BED0BBD18CD188D0B5-
4 00000025 D0B520D187D0B8D181-
4 0000002E D0BBD0BE3A2000
5 00000035 32300000 A dd '20'
6 00000039 35300000 C dd '50'
7 section .bss
8 00000000 <res Ah> max resb 10
9 0000000A <res Ah> B resb 10
10 section .text
11 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 000000E8 B8[00000000] mov eax,msg1
15 000000ED E81DFFFFFF call sprint
16 ; ----- Ввод 'B'
17 000000F2 B9[0A000000] mov ecx,B
18 000000F7 BA0A000000 mov edx,10
19 000000FC E842FFFFFF call sread
20 ; ----- Преобразование 'B' из символа в число
21 00000101 B8[0A000000] mov eax,B
22 00000106 E891FFFFFF call atoi ; Вызов подпрограммы перевода символа в число
23 0000010B A3[0A000000] mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
25 00000110 8B0D[35000000] mov ecx,[A] ; 'ecx = A'
26 00000116 8B0D[00000000] mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 0000011C 3B0D[39000000] cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 00000122 7F0C jg check_B ; если 'A'>'C', то переход на метку 'check_B',
30 00000124 8B0D[39000000] mov ecx,[C] ; иначе 'ecx = C'

```

Рис. 2.9: Файл листинга lab8-2

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

строка 10

- 10 - номер строки
- 00000006 - адрес
- 7403 - машинный код
- jz finished - код программы

строка 11

- 11 - номер строки
- 00000008 - адрес
- 40 - машинный код
- inc eax - код программы

строка 12

- 12 - номер строки
- 00000009 - адрес
- EBF8 - машинный код
- jmp nextchar - 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)

```
[sacvelev@fedora lab08]$
[sacvelev@fedora lab08]$ nasm -f elf lab8-2.asm -l lab8-2.lst
[sacvelev@fedora lab08]$
[sacvelev@fedora lab08]$
[sacvelev@fedora lab08]$ nasm -f elf lab8-2.asm -l lab8-2.lst
lab8-2.asm:23: error: invalid combination of opcode and operands
[sacvelev@fedora lab08]$
```

Рис. 2.10: ошибка трансляции lab8-2


```

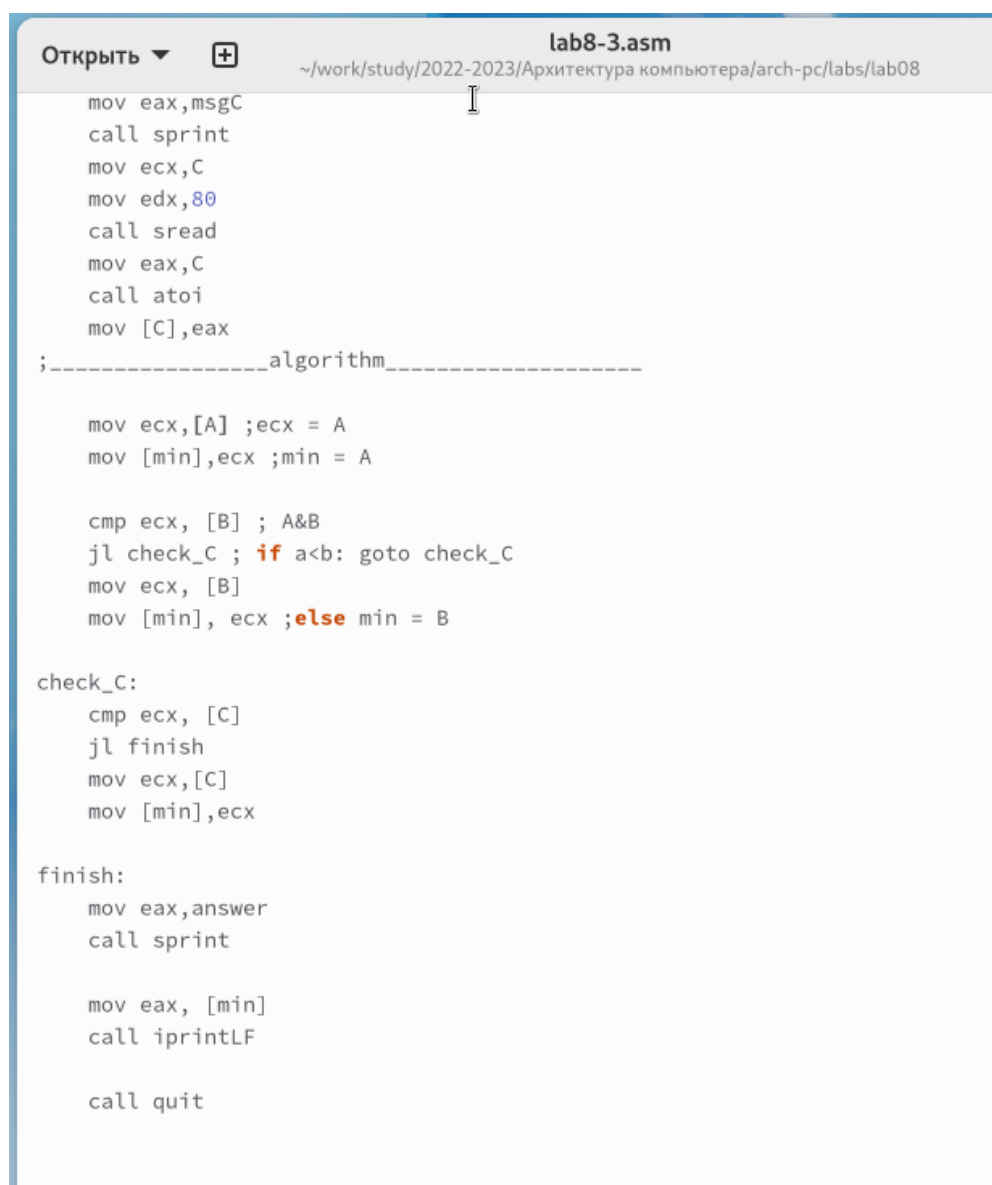
11  global _start
12  _start:
13  ; ----- Вывод сообщения 'Введите B: '
14  000000E8 B8[00000000] mov eax,msg1
15  000000ED E81DFFFFFF call sprint
16  ; ----- Ввод 'B'
17  000000F2 B9[0A000000] mov ecx,B
18  000000F7 BA0A000000 mov edx,10
19  000000FC E842FFFFFF call sread
20  ; ----- Преобразование 'B' из символа в число
21  00000101 B8[0A000000] mov eax,B
22  00000106 E891FFFFFF call atoi ; Вызов подпрограммы перевода символа в число
23  00000107 00000000 mov [B], ; запись преобразованного числа в 'B'
24  ***** error: invalid combination of opcode and operands
25  0000010B 8B0D[35000000] mov ecx,[A] ; 'ecx = A'
26  00000111 890D[00000000] mov [max],ecx ; 'max = A'
27  ; ----- Сравниваем 'A' и 'C' (как символы)
28  00000117 3B0D[39000000] cmp ecx,[C] ; Сравниваем 'A' и 'C'
29  0000011D 7F0C jg check_B ; если 'A>C', то переход на метку 'check_B',
30  0000011F 8B0D[39000000] mov ecx,[C] ; иначе 'ecx = C'
31  00000125 890D[00000000] mov [max],ecx ; 'max = C'
32  ; ----- Преобразование 'max(A,C)' из символа в число
33  check_B:
34  0000012B B8[00000000] mov eax,max
35  00000130 E867FFFFFF call atoi ; Вызов подпрограммы перевода символа в число
36  00000135 A3[00000000] mov [max],eax ; запись преобразованного числа в 'max'
37  ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
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 на 'fin',
41  00000148 8B0D[0A000000] mov ecx,[B] ; иначе 'ecx = B'
42  0000014E 890D[00000000] mov [max],ecx

```

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 12 - 99, 29, 26



```
lab8-3.asm
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08

mov eax,msgC
call sprint
mov ecx,C
mov edx,80
call sread
mov eax,C
call atoi
mov [C],eax

;-----algorithm-----

mov ecx,[A] ;ecx = A
mov [min],ecx ;min = A

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

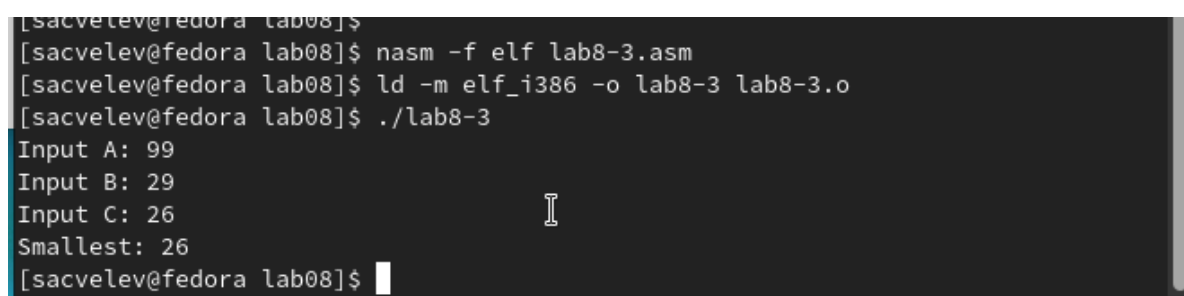
check_C:
cmp ecx, [C]
jlt finish
mov ecx,[C]
mov [min],ecx

finish:
mov eax,answer
call sprint

mov eax, [min]
call iprintLF

call quit
```

Рис. 2.12: Файл lab8-3.asm



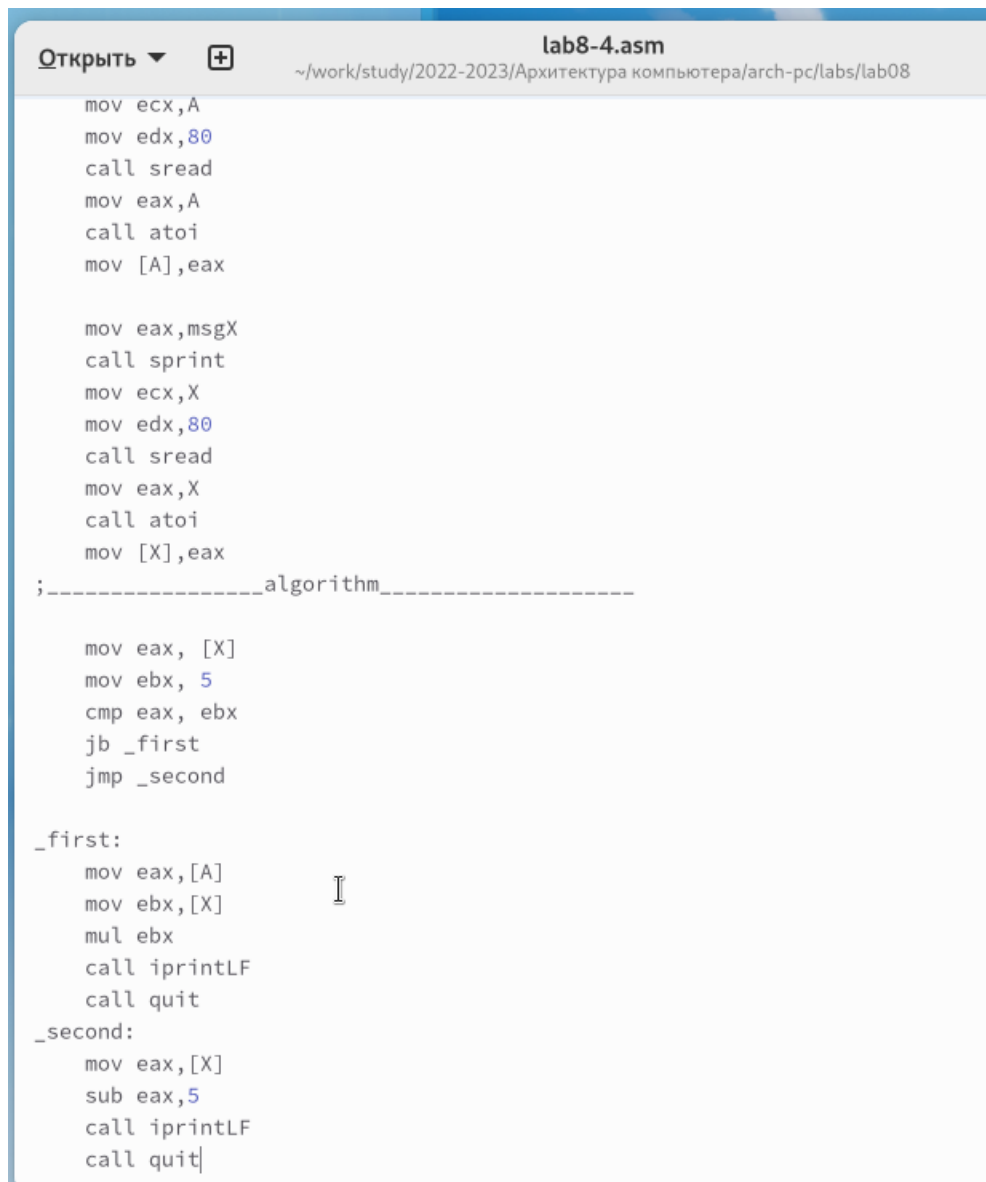
```
[sacvelev@fedora lab08]$
[sacvelev@fedora lab08]$ nasm -f elf lab8-3.asm
[sacvelev@fedora lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o
[sacvelev@fedora lab08]$ ./lab8-3
Input A: 99
Input B: 29
Input C: 26
Smallest: 26
[sacvelev@fedora lab08]$
```

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 12

$$\begin{cases} ax, & x < 5 \\ x - 5, & x \geq 5 \end{cases}$$



```
lab8-4.asm
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08

mov ecx,A
mov edx,80
call sread
mov eax,A
call atoi
mov [A],eax

mov eax,msgX
call sprint
mov ecx,X
mov edx,80
call sread
mov eax,X
call atoi
mov [X],eax

;-----algorithm-----

mov eax, [X]
mov ebx, 5
cmp eax, ebx
jb _first
jmp _second

_first:
mov eax,[A]
mov ebx,[X]
mul ebx
call iprintLF
call quit
_second:
mov eax,[X]
sub eax,5
call iprintLF
call quit
```

Рис. 2.14: Файл lab8-4.asm

```
[sacvelev@fedora lab08]$  
[sacvelev@fedora lab08]$ nasm -f elf lab8-4.asm  
[sacvelev@fedora lab08]$ ld -m elf_i386 -o lab8-4 lab8-4.o  
[sacvelev@fedora lab08]$ ./lab8-4  
Input A: 7  
Input X: 3  
21  
[sacvelev@fedora lab08]$ ./lab8-4  
Input A: 4  
Input X: 6  
1  
[sacvelev@fedora lab08]$
```

Рис. 2.15: Программа lab8-4.asm

3 Выводы

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