

Отчёта по лабораторной работе 8

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

Игнатенкова В.Н

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	23
	Список литературы	24

Список иллюстраций

4.1	Файл lab8-1.asm:	8
4.2	Программа lab8-1.asm:	9
4.3	Файл lab8-1.asm:	10
4.4	Программа lab8-1.asm:	11
4.5	Файл lab8-1.asm	12
4.6	Программа lab8-1.asm	13
4.7	Файл lab8-2.asm	14
4.8	Программа lab8-2.asm	15
4.9	Файл листинга lab8-2	16
4.10	ошибка трансляции lab8-2	17
4.11	файл листинга с ошибкой lab8-2	18
4.12	Файл lab8-3.asm	19
4.13	Программа lab8-3.asm	20
4.14	Файл lab8-4.asm	21
4.15	Программа lab8-4.asm	22

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

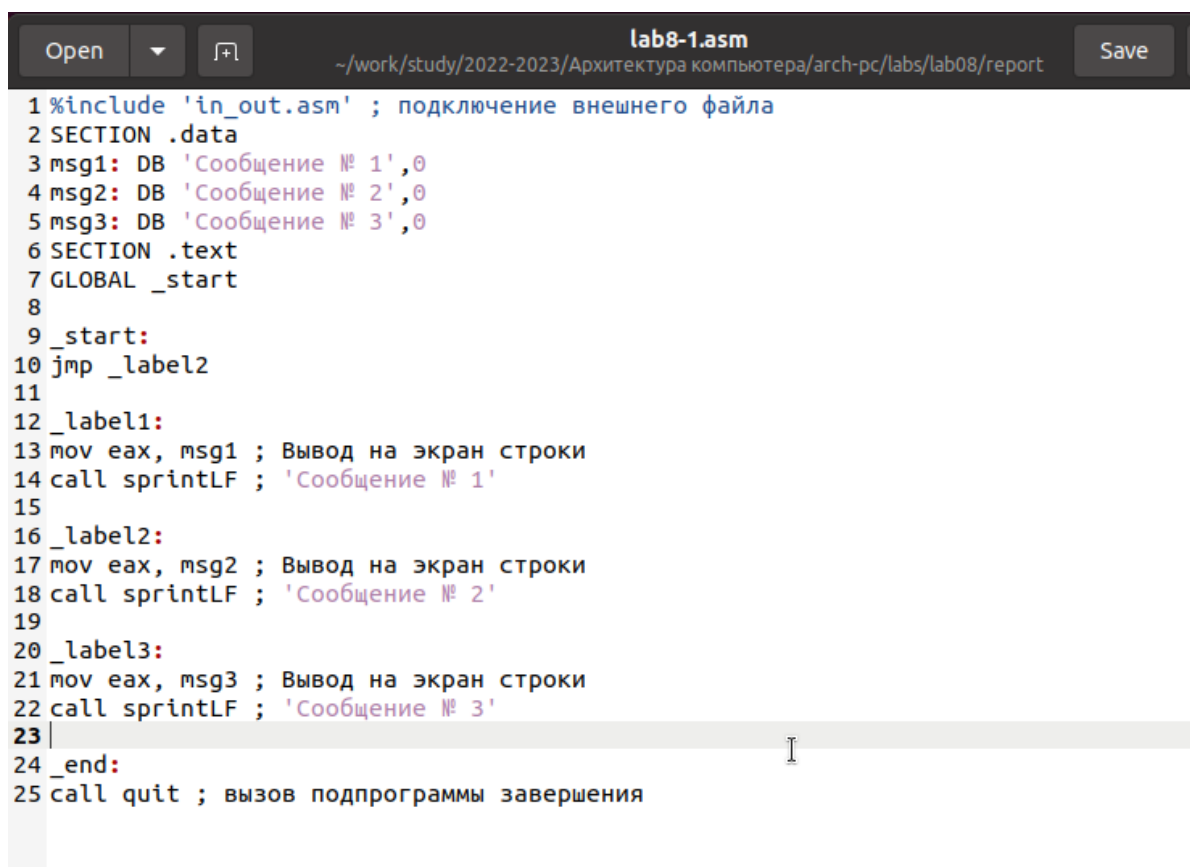
Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов:

- 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

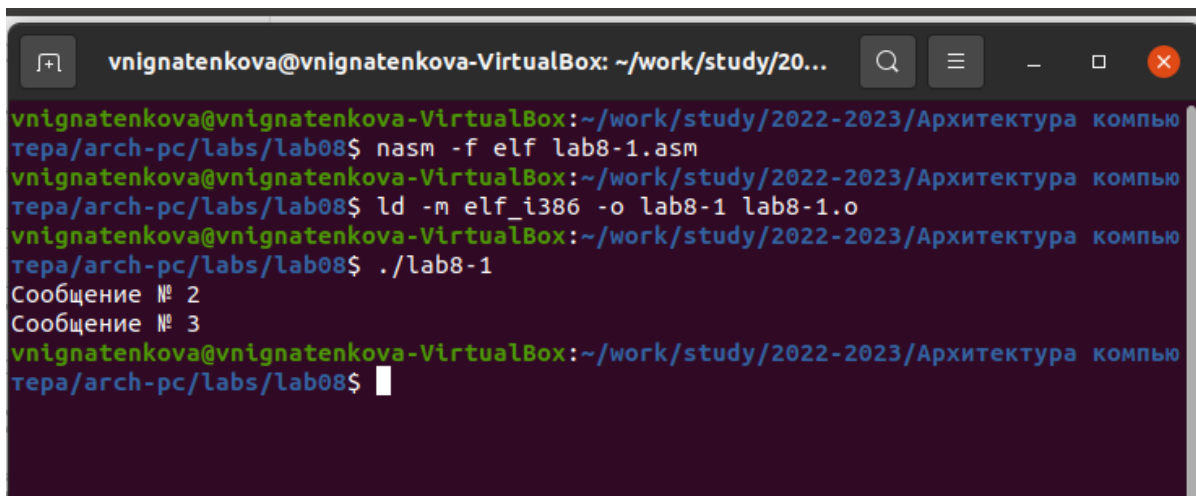
1. Создайте каталог для программ лабораторной работы № 8, перейдите в него и создайте файл lab8-1.asm
2. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции jmp. Введите в файл lab8-1.asm текст программы из листинга 8.1. (рис. 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 sprintfLF ; 'Сообщение № 1'
15
16 _label2:
17 mov eax, msg2 ; Вывод на экран строки
18 call sprintfLF ; 'Сообщение № 2'
19
20 _label3:
21 mov eax, msg3 ; Вывод на экран строки
22 call sprintfLF ; 'Сообщение № 3'
23
24 _end:
25 call quit ; вызов подпрограммы завершения
```

Рис. 4.1: Файл lab8-1.asm:

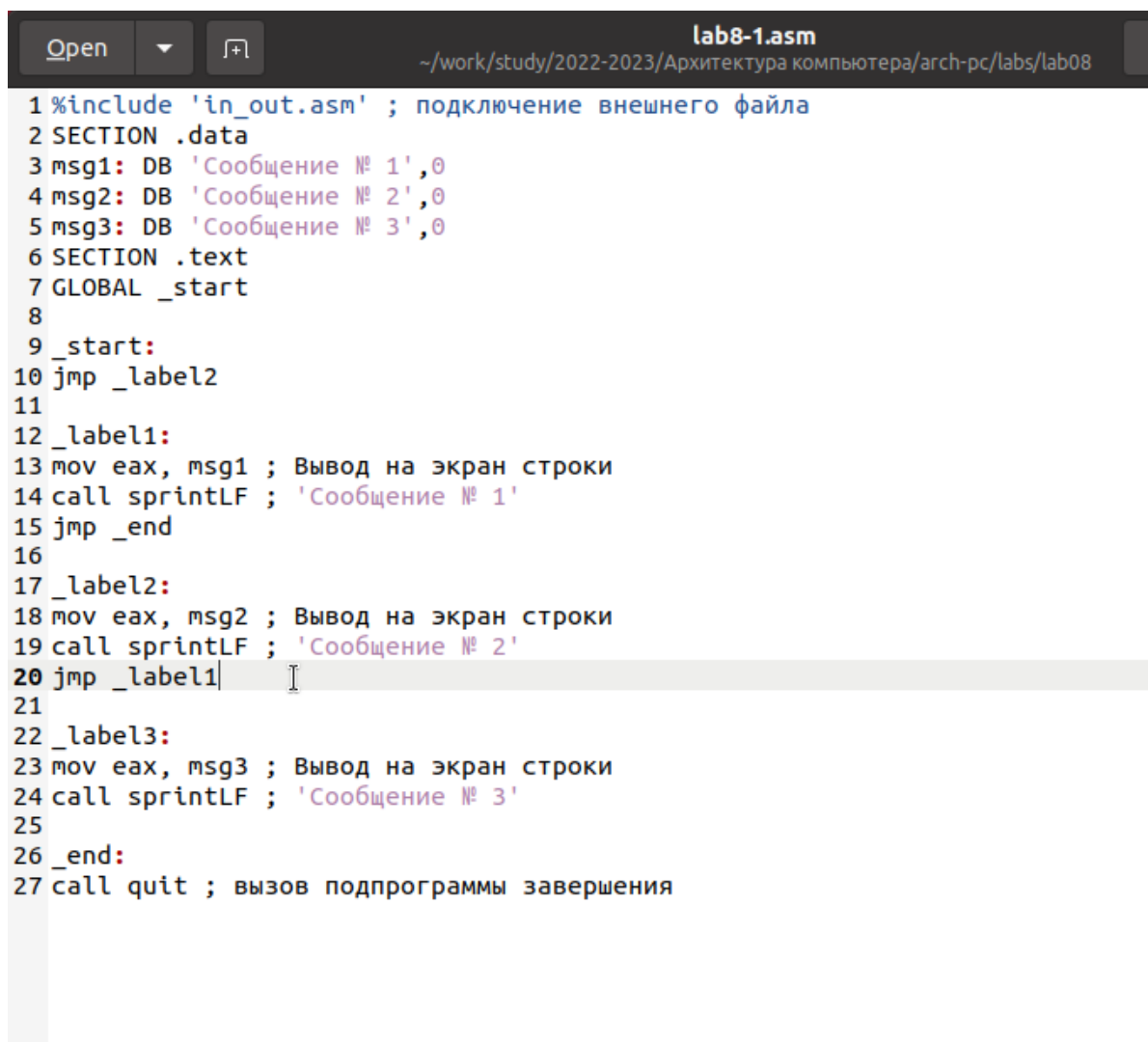
Создайте исполняемый файл и запустите его. (рис. 4.2)



```
vnignatenkova@vnignatenkova-VirtualBox: ~/work/study/20...
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ./lab8-1
Сообщение № 2
Сообщение № 3
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/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)



```
lab8-1.asm
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08

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 printf ; 'Сообщение № 1'
15 jmp _end
16
17 _label2:
18 mov eax, msg2 ; Вывод на экран строки
19 call printf ; 'Сообщение № 2'
20 jmp _label1
21
22 _label3:
23 mov eax, msg3 ; Вывод на экран строки
24 call printf ; 'Сообщение № 3'
25
26 _end:
27 call quit ; вызов подпрограммы завершения
```

Рис. 4.3: Файл lab8-1.asm:

```
vnignatenkova@vnignatenkova-VirtualBox: ~/work/study/20...
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ./lab8-1
Сообщение № 2
Сообщение № 3
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
lab8-1.asm:20: error: invalid combination of opcode and operands
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ./lab8-1
Сообщение № 2
Сообщение № 1
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$
```

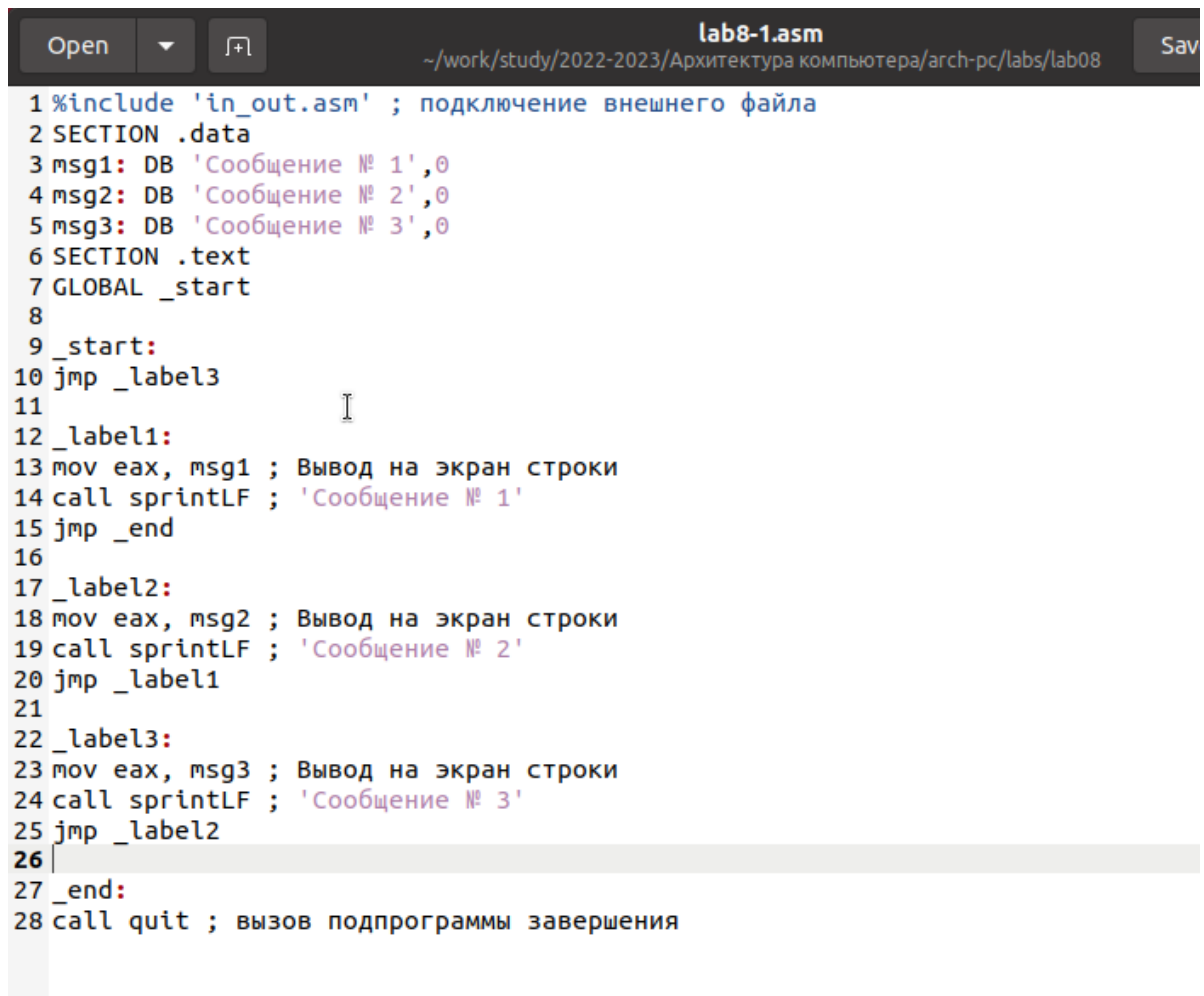
Рис. 4.4: Программа lab8-1.asm:

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

Сообщение № 3

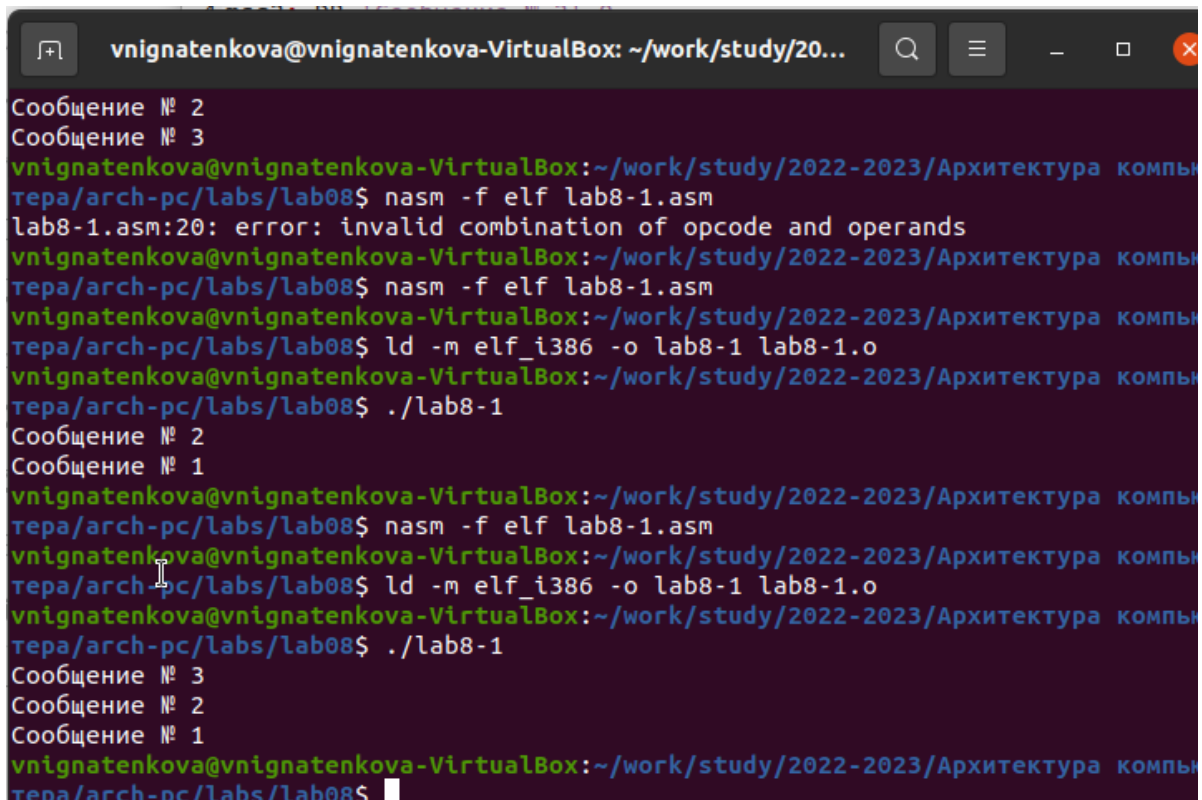
Сообщение № 2

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
9 _start:
10 jmp _label3
11
12 _label1:
13 mov eax, msg1 ; Вывод на экран строки
14 call printf ; 'Сообщение № 1'
15 jmp _end
16
17 _label2:
18 mov eax, msg2 ; Вывод на экран строки
19 call printf ; 'Сообщение № 2'
20 jmp _label1
21
22 _label3:
23 mov eax, msg3 ; Вывод на экран строки
24 call printf ; 'Сообщение № 3'
25 jmp _label2
26
27 _end:
28 call quit ; вызов подпрограммы завершения
```


Рис. 4.5: Файл lab8-1.asm



```
vnignatenkova@vnignatenkova-VirtualBox: ~/work/study/20...
Сообщение № 2
Сообщение № 3
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
lab8-1.asm:20: error: invalid combination of opcode and operands
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/lab08$ ./lab8-1
Сообщение № 2
Сообщение № 1
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/lab08$ ./lab8-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компьютерных систем/тепа/arch-pc/labs/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: A, B и C. Значения для A и C задаются в программе, значение B вводится с клавиатуры. Создайте исполняемый файл и проверьте его работу для разных значений B. (рис. 4.7, 4.8)



```
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 jg check_B ; если 'A>C', то переход на метку 'check_B',
30 mov ecx,[C] ; иначе 'ecx = C'
31 mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 mov eax,max
35 call atoi ; Вызов подпрограммы перевода символа в число
36 mov [max],eax ; запись преобразованного числа в 'max'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 mov ecx,[max]
39 cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
40 jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 mov ecx,[B] ; иначе 'ecx = B'
42 mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 mov eax,msg2
46 call sprint ; Вывод сообщения 'Наибольшее число: '
47 mov eax,[max]
48 call iprintLF ; Вывод 'max(A,B,C)'
49 call quit ; Выход
50
```

Saving file "/home/vnignatenkova/work/study/2022-2023/Архите... Matlab ▾ Tab Width: 8 ▾ Ln 34,

Рис. 4.7: Файл lab8-2.asm

```

тера/arch-pc/labs/lab08$
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ nasm -f elf lab8-2.asm
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ./lab8-2
Введите В: 120
Наибольшее число: 120
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ./lab8-2
Введите В: 40
Наибольшее число: 50
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/lab08$ ./lab8-2
Введите В: 50
Наибольшее число: 50
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тера/arch-pc/labs/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)

```

1 1 %include 'in_out.asm'
2 1 <1> ;----- slen -----
3 2 <1> ; Функция вычисления длины сообщения
4 3 <1> slen:
5 4 00000000 53 <1> push ebx
6 5 00000001 89C3 <1> mov ebx, eax
7 6 <1>
8 7 <1> nextchar:
9 8 00000003 803800 <1> cmp byte [eax], 0
10 9 00000006 7403 <1> jz finished
11 10 00000008 40 <1> inc eax
12 11 00000009 EBF8 <1> jmp nextchar
13 12 <1>
14 13 <1> finished:
15 14 0000000B 29D8 <1> sub eax, ebx
16 15 0000000D 5B <1> pop ebx
17 16 0000000E C3 <1> ret
18 17 <1>
19 18 <1>
20 19 <1> ;----- sprint -----
21 20 <1> ; Функция печати сообщения
22 21 <1> ; входные данные: mov eax,<message>
23 22 <1> sprint:
24 23 0000000F 52 <1> push edx
25 24 00000010 51 <1> push ecx
26 25 00000011 53 <1> push ebx
27 26 00000012 50 <1> push eax
28 27 00000013 E8E8FFFFFF <1> call slen
29 28 <1>
30 29 00000018 89C2 <1> mov edx, eax
31 30 0000001A 58 <1> pop eax
32 31 <1>
33 32 0000001B 89C1 <1> mov ecx, eax
34 33 0000001D B801000000 <1> mov ebx, 1
35 34 00000022 B804000000 <1> mov eax, 4

```

Рис. 4.9: 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
- 00000008 - адрес
- 40 - машинный код
- inc eax - код программы

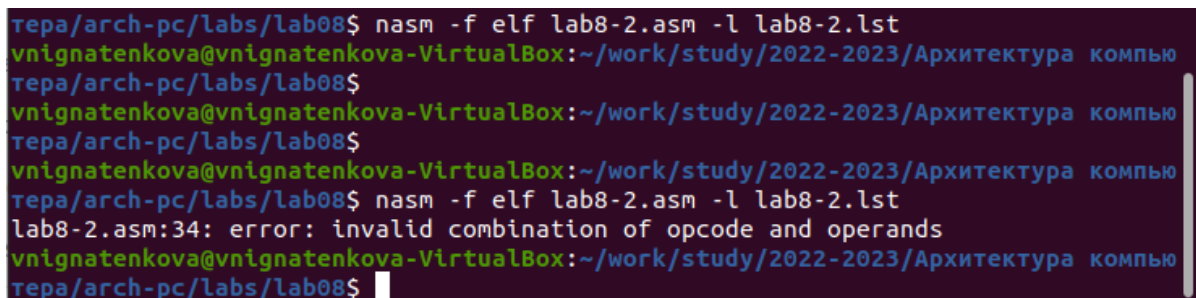
строка 11

- 11 - номер строки
- 00000009 - адрес
- EBF8 - машинный код
- jmp nextchar- код программы

строка 14

- 14 - номер строки
- 0000000B - адрес
- 29D8 - машинный код
- sub eax, ebx - код программы

Откройте файл с программой lab8-2.asm и в любой инструкции с двумя операндами удалите один операнд. Выполните трансляцию с получением файла листинга (рис. 4.10,4.11)



```

тепа/arch-pc/labs/lab08$ nasm -f elf lab8-2.asm -l lab8-2.lst
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$ nasm -f elf lab8-2.asm -l lab8-2.lst
lab8-2.asm:34: error: invalid combination of opcode and operands
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$

```

Рис. 4.10: ошибка трансляции lab8-2

```

lab8-2.asm
196 21 00000101 B8[0A000000]
197 22 00000106 E891FFFFFF
число
198 23 0000010B A3[0A000000]
199 24
200 25 00000110 8B0D[35000000]
201 26 00000116 890D[00000000]
202 27
203 28 0000011C 3B0D[39000000]
204 29 00000122 7F0C
'check_B',
205 30 00000124 8B0D[39000000]
206 31 0000012A 890D[00000000]
207 32
число
208 33
209 34
210 34 *****
211 35 00000130 E867FFFFFF
число
212 36 00000135 A3[00000000]
213 37
214 38 0000013A 8B0D[00000000]
215 39 00000140 3B0D[0A000000]
216 40 00000146 7F0C
217 41 00000148 8B0D[0A000000]
218 42 0000014E 890D[00000000]
219 43
220 44
221 45 00000154 B8[13000000]
222 46 00000159 E8B1FEFFFF
223 47 0000015E A1[00000000]
224 48 00000163 E81EFFFFFF
225 49 00000168 E86EFFFFFF
226 50

lab8-2.lst
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в
; ----- 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
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в
check_B:
mov eax
error: invalid combination of opcode and operands
call atoi ; Вызов подпрограммы перевода символа в
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. 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 18 - 83, 73, 30

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

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
43  call atoi
44  mov [C],eax
45 ;_____algorithm_____
46
47  mov ecx,[A] ;ecx = A
48  mov [min],ecx ;min = A
49
50  cmp ecx, [B] ; A&B
51  jl check_C ; if a<b: goto check_C
52  mov ecx, [B]
53  mov [min], ecx ;else min = B
54
55 check_C:
56  cmp ecx, [C]
57  jl finish
58  mov ecx,[C]
59  mov [min],ecx
60
61 finish:
62  mov eax,answer
63  call sprint
64
65  mov eax, [min]
66  call iprintLF
67
68  call quit
69
70
```

Рис. 4.12: Файл lab8-3.asm

```

тепа/arch-pc/labs/lab08$
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$ nasm -f elf lab8-3.asm
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$ ./lab8-3
Input A: 83
Input B: 73
Input C: 30
Smallest: 30
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью
тепа/arch-pc/labs/lab08$
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью

```

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 18

$$\begin{cases} a^2, a \neq 1 \\ 10 + x, a = 1 \end{cases}$$

```
lab8-4.asm
~/work/study/2022-2023/Архитектура компью

15     mov eax,msgA
16     call sprint
17     mov ecx,A
18     mov edx,80
19     call sread
20     mov eax,A
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, 1
35     mov edx, [A]
36     cmp ebx, edx
37     jne first
38     jmp second
39
40 first:
41     mov eax,[A]
42     mov ebx,[A]
43     mul ebx
44     call iprintLF
45     call quit
46 second:
47     mov eax,[X]
48     add eax,10
49     call iprintLF
50     call quit
51
52
```

Рис. 4.14: Файл lab8-4.asm

```
repa/arch-pc/labs/lab08$  
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью  
repa/arch-pc/labs/lab08$ nasm -f elf lab8-4.asm  
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью  
repa/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o  
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью  
repa/arch-pc/labs/lab08$ ./lab8-4  
Input A: 2  
Input X: 1  
4  
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью  
repa/arch-pc/labs/lab08$ ./lab8-4  
Input A: 1  
Input X: 2  
12  
vnignatenkova@vnignatenkova-VirtualBox:~/work/study/2022-2023/Архитектура компью  
repa/arch-pc/labs/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