

# **Отчёт по лабораторной работе №7**

**Дисциплина: архитектура компьютеров и операционные системы**

Бизев Никита Владимирович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
4.1	Реализация переходов в NASM . . . . .	8
4.2	Изучение структуры файлы листинга . . . . .	13
<b>5</b>	<b>Выводы</b>	<b>15</b>
<b>6</b>	<b>Список литературы</b>	<b>16</b>

## Список иллюстраций

4.1	Создание файлов для лабораторной работы . . . . .	8
4.2	Ввод текста программы из листинга 7.1 . . . . .	9
4.3	Запуск программного кода . . . . .	9
4.4	Изменение текста программы . . . . .	10
4.5	Создание исполняемого файла . . . . .	10
4.6	Изменение текста программы . . . . .	11
4.7	Вывод программы . . . . .	11
4.8	Создание файла . . . . .	12
4.9	Ввод текста программы из листинга 7.3 . . . . .	12
4.10	Проверка работы файла . . . . .	12
4.11	Создание файла листинга . . . . .	13
4.12	Изучение файла листинга . . . . .	13
4.13	Выбранные строки файла . . . . .	14
4.14	Удаление выделенного операнда из кода . . . . .	14
4.15	Получение файла листинга . . . . .	14

## **Список таблиц**

# 1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

## 2 Задание

1. Реализация переходов в NASM.
2. Изучение структуры файлы листинга.
3. Задания для самостоятельной работы.

### 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.

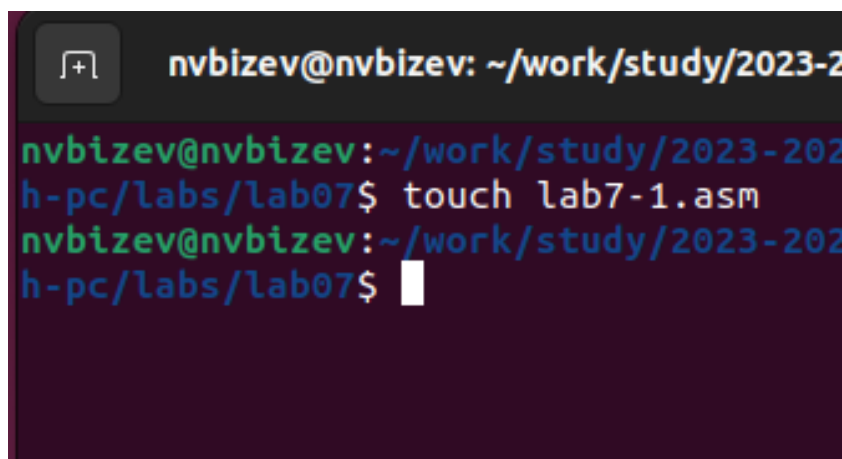
Безусловный переход выполняется инструкцией `jmp`. Инструкция `cmp` является одной из инструкций, которая позволяет сравнить операнды и выставляет флаги в зависимости от результата сравнения. Инструкция `cmp` является командой сравнения двух операндов и имеет такой же формат, как и команда вычитания.

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

### 4.1 Реализация переходов в NASM

Создаю каталог для программ лабораторной работы № 7, перехожу в него и создаю файл lab7-1.asm. (рис. 4.1).

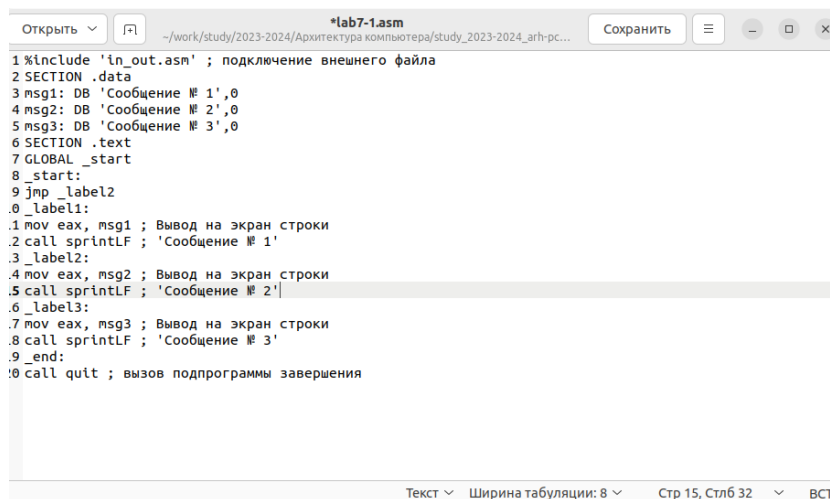


```
nvbizev@nvbizev: ~/work/study/2023-2024
nvbizev@nvbizev:~/work/study/2023-2024
h-pc/labs/lab07$ touch lab7-1.asm
nvbizev@nvbizev:~/work/study/2023-2024
h-pc/labs/lab07$
```

Рис. 4.1: Создание файлов для лабораторной работы

Ввожу в файл lab7-1.asm текст программы из листинга 7.1. (рис. 4.2).

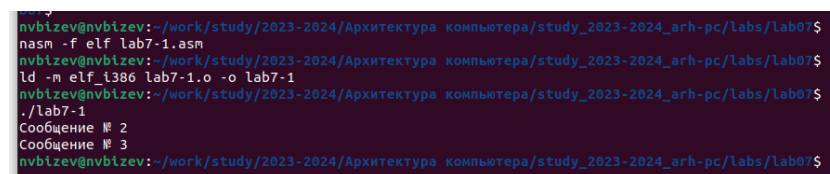




```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label2
10 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'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

Рис. 4.2: Ввод текста программы из листинга 7.1

Создаю исполняемый файл и запускаю его. (рис. 4.3).

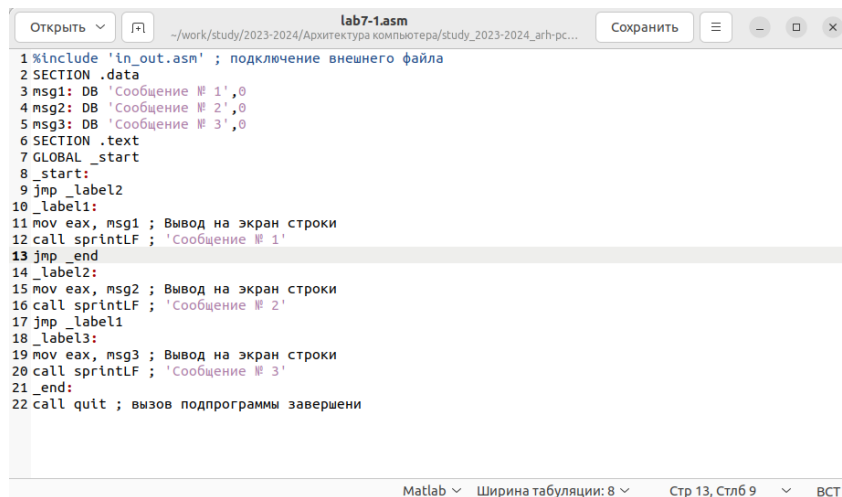


```
nvblizev@nvblizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab07$
nasm -f elf lab7-1.asm
nvblizev@nvblizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab07$
ld -m elf_i386 lab7-1.o -o lab7-1
nvblizev@nvblizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab07$
./lab7-1
Сообщение № 2
Сообщение № 3
nvblizev@nvblizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/labs/lab07$
```

Рис. 4.3: Запуск программного кода

Таким образом, использование инструкции `jmp _label2` меняет порядок исполнения инструкций и позволяет выполнить инструкции начиная с метки `_label2`, пропустив вывод первого сообщения.

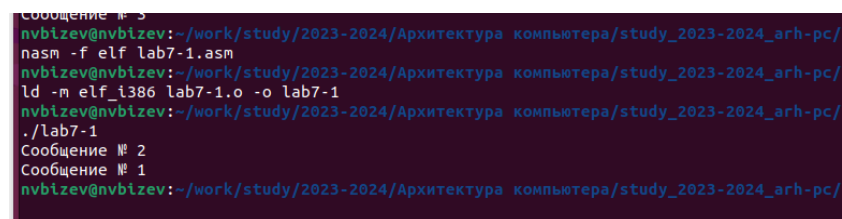
Изменяю программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого изменяю текст программы в соответствии с листингом 7.2. (рис. 4.4).



```
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 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 jmp _end
14 _label2:
15 mov eax, msg2 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 2'
17 jmp _label1
18 _label3:
19 mov eax, msg3 ; Вывод на экран строки
20 call sprintf ; 'Сообщение № 3'
21 _end:
22 call quit ; вызов подпрограммы завершени
```

Рис. 4.4: Изменение текста программы

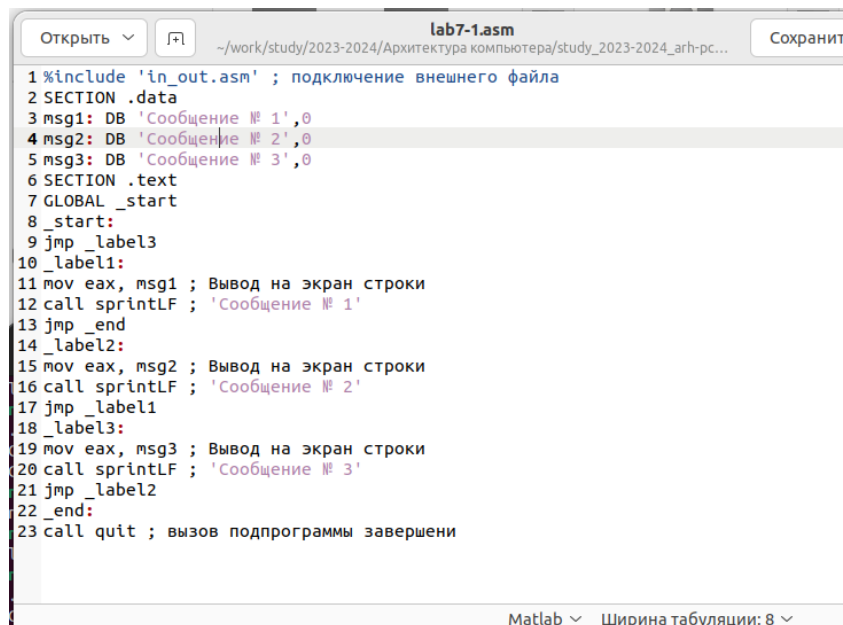
Создаю исполняемый файл и проверяю его работу. (рис. 4.5).



```
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/
nasm -f elf lab7-1.asm
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/
ld -m elf_i386 lab7-1.o -o lab7-1
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/
./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arh-pc/
```

Рис. 4.5: Создание исполняемого файла

Затем изменяю текст программы, добавив в начале программы `jmp _label3`, `jmp _label2` в конце метки `jmp _label3`, `jmp _label1` добавляю в конце метки `jmp _label2`, и добавляю `jmp _end` в конце метки `jmp _label1`, (рис. 4.6).

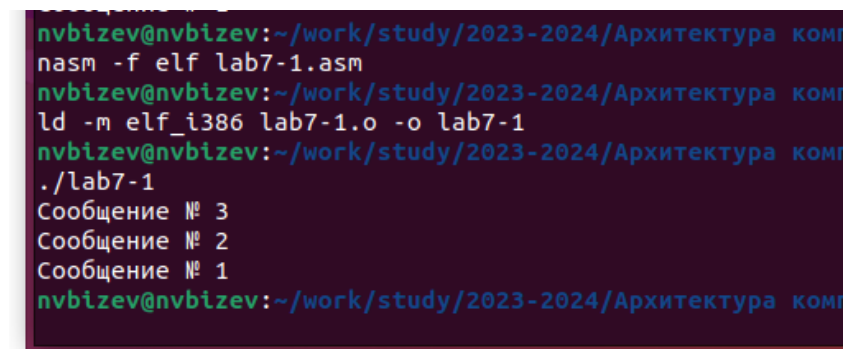


```
lab7-1.asm
~/work/study/2023-2024/Архитектура компьютера/study_2023-2024_arch-пс...
Открыть Сохранить

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 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 jmp _end
14 _label2:
15 mov eax, msg2 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 2'
17 jmp _label1
18 _label3:
19 mov eax, msg3 ; Вывод на экран строки
20 call sprintf ; 'Сообщение № 3'
21 jmp _label2
22 _end:
23 call quit ; вызов подпрограммы завершени
```

Рис. 4.6: Изменение текста программы

чтобы вывод программы был следующим: (рис. 4.7).



```
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура комп
nasm -f elf lab7-1.asm
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура комп
ld -m elf_i386 lab7-1.o -o lab7-1
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура комп
./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура комп
```

Рис. 4.7: Вывод программы

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.

Создаю файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. (рис. 4.8).

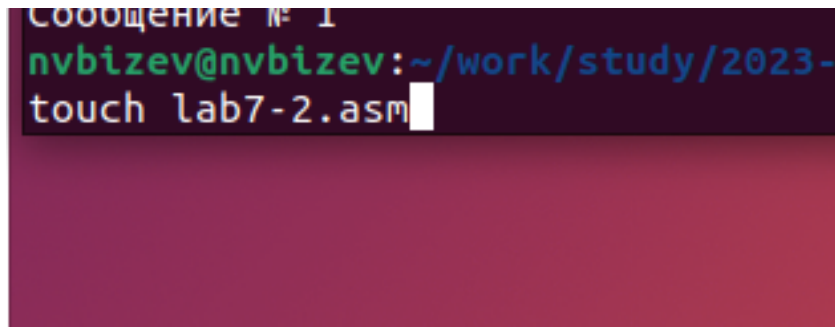


Рис. 4.8: Создание файла

Текст программы из листинга 7.3 ввожу в lab7-2.asm. (рис. 4.9).

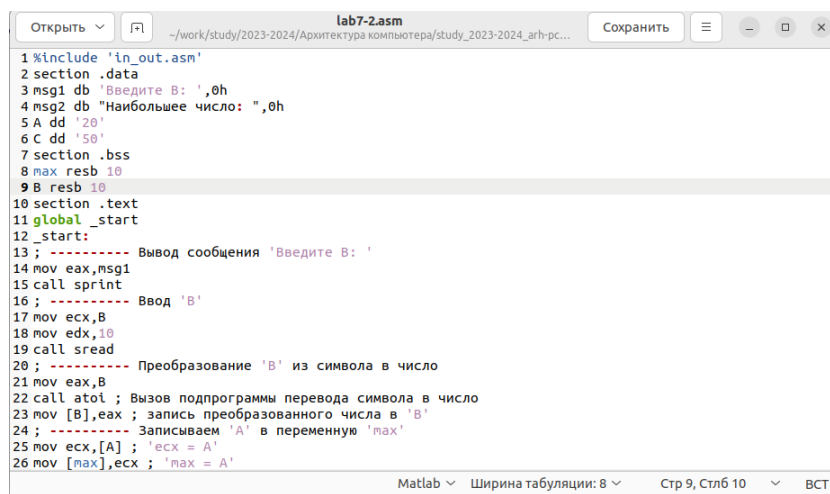


Рис. 4.9: Ввод текста программы из листинга 7.3

Создаю исполняемый файл и проверьте его работу. (рис. 4.10).

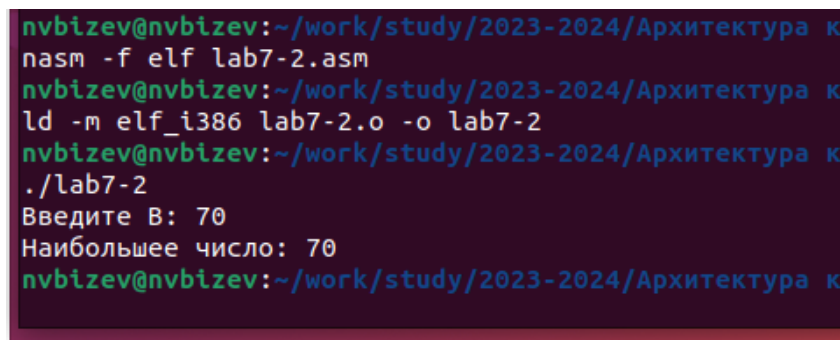


Рис. 4.10: Проверка работы файла

Файл работает корректно.

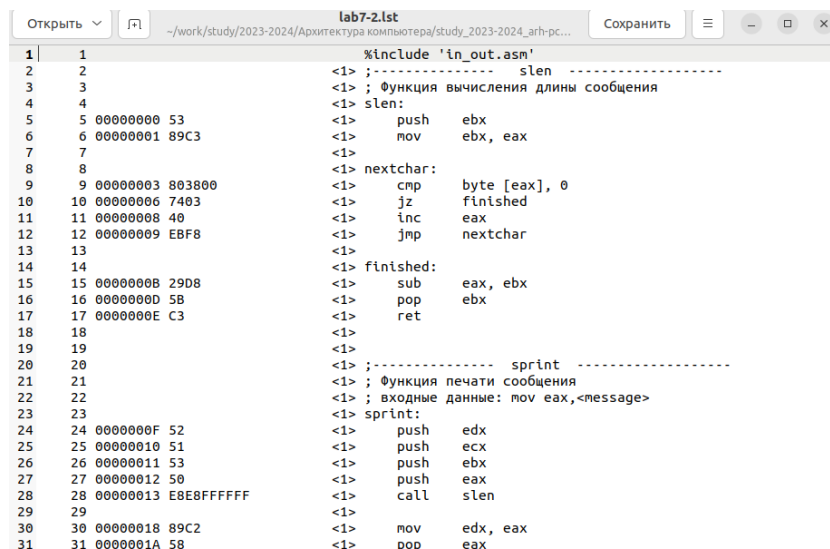
## 4.2 Изучение структуры файлы листинга

Создаю файл листинга для программы из файла lab7-2.asm. (рис. 4.11).

```
nvbizev@nvbizev:~/work/study/2023-2024/Архитек  
nasm -f elf -l lab7-2.lst lab7-2.asm  
nvbizev@nvbizev:~/work/study/2023-2024/Архитек
```

Рис. 4.11: Создание файла листинга

Открываю файл листинга lab7-2.lst с помощью текстового редактора и внимательно изучаю его формат и содержимое. (рис. 4.12).



```
1 1 %include 'in_out.asm'
2 2 <1> ;----- slen -----
3 3 <1> ; Функция вычисления длины сообщения
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 EBF8 <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> ;----- sprint -----
21 21 <1> ; Функция печати сообщения
22 22 <1> ; входные данные: mov eax, <message>
23 23 <1> sprint:
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 5B <1> pop eax
```

Рис. 4.12: Изучение файла листинга

В представленных трех строчках содержатся следующие данные: (рис. 4.13).

3	3	<1> ; Функция вычисления длины сообщения
4	4	<1> slen:
5	5 00000000 53	<1> push ebx

Рис. 4.13: Выбранные строки файла

“3” - номер строки кода, “; Функция вычисления длины сообщения” - комментарий к коду, не имеет адреса и машинного кода.

“4” - номер строки кода, “slen” - название функции, не имеет адреса и машинного кода.

“5” - номер строки кода, “00000000” - адрес строки, “53” - машинный код, “push ebx” - исходный текст программы, инструкция “push” помещает операнд “ebx” в стек.

Открываю файл с программой lab7-2.asm и в выбранной мной инструкции с двумя операндами удаляю выделенный операнд. (рис. 4.14).

```
mov [max],ecx ; max = A
----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
je check_B ; если 'A' < 'C' то переход из метки 'check_B'
```

Рис. 4.14: Удаление выделенного операнда из кода

Выполняю трансляцию с получением файла листинга. (рис. 4.15).

```
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура компьютера/stu
nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:28: error: invalid combination of opcode and operands
nvbizev@nvbizev:~/work/study/2023-2024/Архитектура компьютера/stu
```

Рис. 4.15: Получение файла листинга

На выходе я не получаю ни одного файла из-за ошибки:инструкция mov (единственная в коде содержит два операнда) не может работать, имея только один операнд, из-за чего нарушается работа кода.

## 5 Выводы

По итогам данной лабораторной работы я изучил команды условного и безусловного переходов, приобрел навыки написания программ с использованием переходов и ознакомился с назначением и структурой файла листинга, что поможет мне при выполнении последующих лабораторных работ.

## 6 Список литературы

[illegible]