

Лабораторная работа No8.

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

Коршунова Полина Юрьевна

Содержание

| | | |
|----------|---|-----------|
| 1 | Цель работы | 5 |
| 2 | Выполнение лабораторной работы | 6 |
| 3 | Задания для самостоятельной работы | 11 |
| 4 | Выводы | 13 |

Список иллюстраций

| | | |
|------|--|----|
| 2.1 | Создание файла | 6 |
| 2.2 | Ввожу в файл lab8-1.asm текст программы из листинга 8.1 | 6 |
| 2.3 | Запускаю программу | 7 |
| 2.4 | Изменяем программу | 7 |
| 2.5 | Запускаю программу | 7 |
| 2.6 | Запускаю программу | 8 |
| 2.7 | Запускаю программу | 8 |
| 2.8 | Создаю файл lab8-2.asm | 9 |
| 2.9 | Запускаю программу | 9 |
| 2.10 | Открываю файл листинга lab8-2.lst с помощью текстового редактора mscedit | 10 |
| 2.11 | Удаляю один операнд | 10 |
| 3.1 | Вывод наименьшего числа | 11 |
| 3.2 | Проверка для значений (3;0) | 11 |
| 3.3 | Проверка для значений (3;2) | 12 |

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.

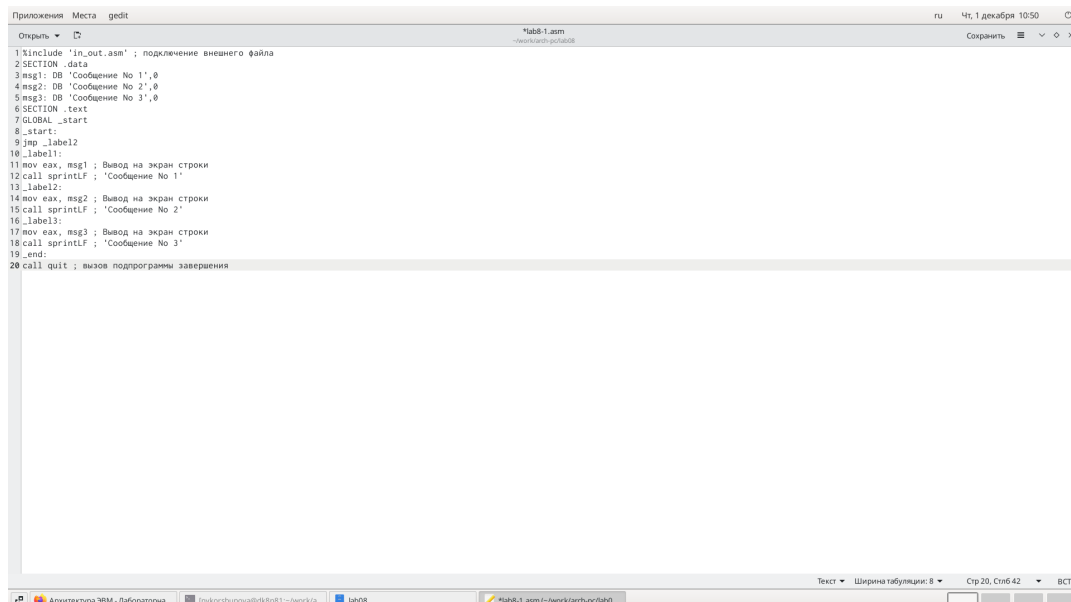
2 Выполнение лабораторной работы

1. Создаю каталог для программ лабораторной работы No 8, перехожу в него и создаю файл lab8-1.asm (рис. 2.1)

```
pykorshunova@dk8n81 ~ $ mkdir ~/work/arch-pc/lab08
pykorshunova@dk8n81 ~ $ cd ~/work/arch-pc/lab08
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ touch lab8-1.asm
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $
```

Рис. 2.1: Создание файла

2. Ввожу в файл lab8-1.asm текст программы из листинга 8.1 (рис. 2.2)



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

Рис. 2.2: Ввожу в файл lab8-1.asm текст программы из листинга 8.1

3. Запускаю программу (рис. 2.3)

```
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 3
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $
```

Рис. 2.3: Запускаю программу

4. Изменяем программу таким образом, чтобы она выводила сначала ‘Сообщение No 2’, потом ‘Сообщение No 1’ и завершала работу (рис. 2.4)

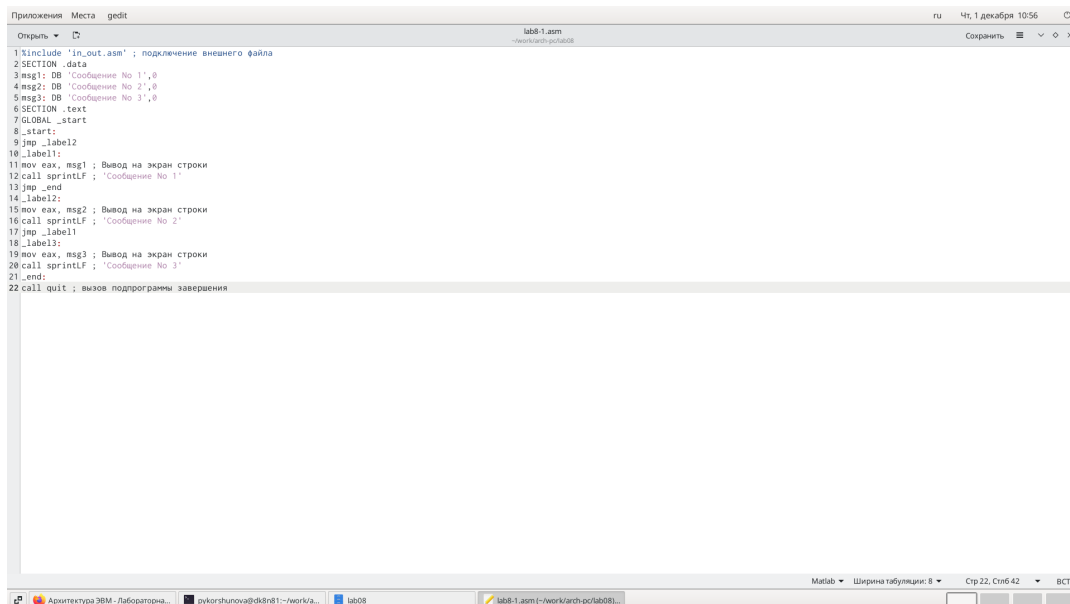


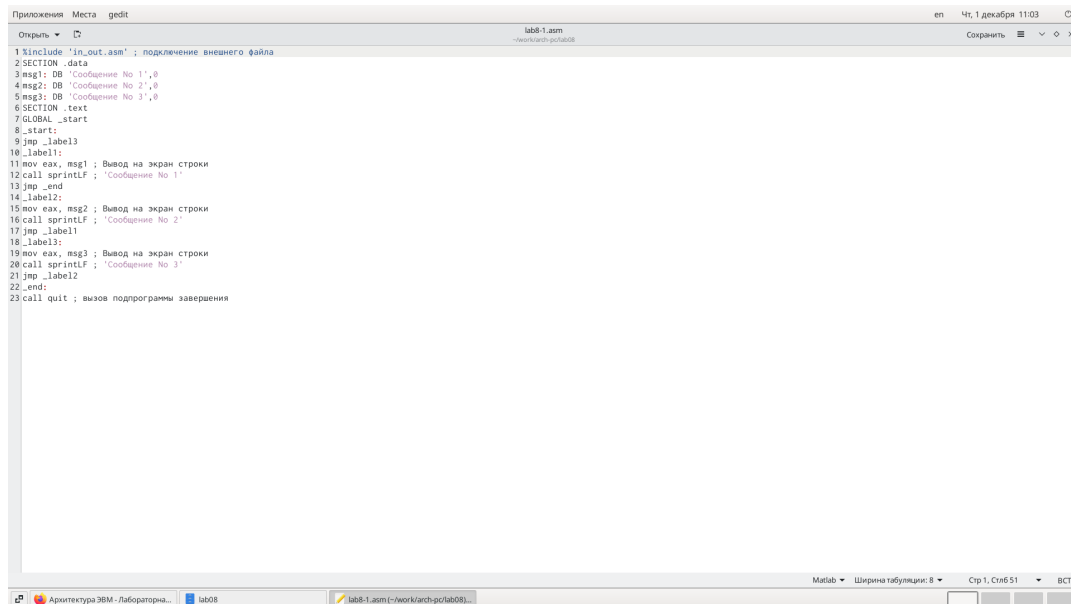
Рис. 2.4: Изменяем программу

5. Запускаю программу (рис. 2.5)

```
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 1
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $
```

Рис. 2.5: Запускаю программу

6. Изменяю текст программы, чтобы выводилось '3, 2, 1' (рис. 2.6)



```
1 %include "in_out.asm" ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение No 1',0
4 msg2: DB 'Сообщение No 2',0
5 msg3: DB 'Сообщение No 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 No 1'
13 jmp _end
14 _label2:
15 mov eax, msg2 ; Вывод на экран строки
16 call sprintf ; 'Сообщение No 2'
17 jmp _label1
18 _label3:
19 mov eax, msg3 ; Вывод на экран строки
20 call sprintf ; 'Сообщение No 3'
21 jmp _label2
22 _end:
23 call quit ; вызов подпрограммы завершения
```

Рис. 2.6: Запускаю программу

7. Запускаю программу (рис. 2.7)

```
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ./lab8-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $
```

Рис. 2.7: Запускаю программу

8. Создаю файл lab8-2.asm в каталоге ~/work/arch-pc/lab08. Внимательно изучитю текст программы из листинга 8.3 и ввожу в lab8-2.asm (рис. 2.8)

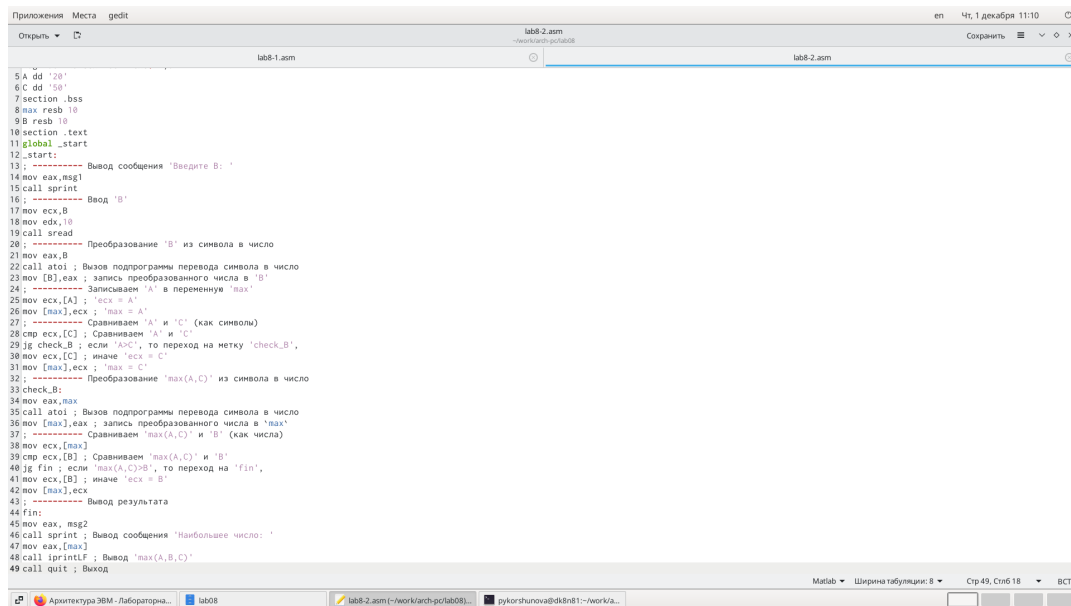


Рис. 2.8: Создаю файл lab8-2.asm

9. Запускаю программу (рис. 2.9)

```

зукоршунова@dk8n81 ~ $ cd ~/work/arch-pc/lab08
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ touch lab8-2.asm
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ ./lab8-2
Введите B: 68
Наибольшее число: 68
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ ./lab8-2
Введите B: 33
Наибольшее число: 50
зукоршунова@dk8n81 ~/work/arch-pc/lab08 $ 

```

Рис. 2.9: Запускаю программу

10. Создаю файл листинга для программы из файла lab8-2.asm. Открываю файл листинга lab8-2.lst с помощью текстового редактора mcedit. Объясняю содержимое трёх строк файла листинга:

- 1) 20, 21, 22 - номер строки.
- 2) 000000F2, 000000F7, 000000FC - это адрес строки.

- 3) B90A000000, BA0A000000, E842FFFFFF - это машинный код.
- 4) 'mov ecx,B', 'call atoi', mov [B], eax - это исходный текст программы (рис. 2.10)

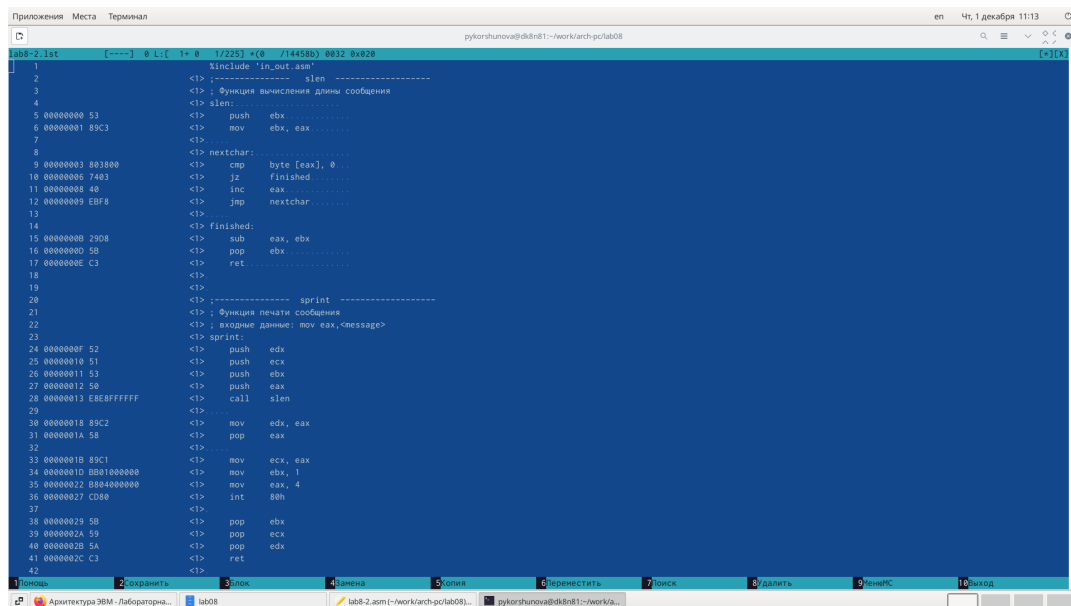


Рис. 2.10: Открываю файл листинга lab8-2.lst с помощью текстового редактора mcedit

11. Открываю файл с программой lab8-2.asm и в любой инструкции с двумя операндами удаляю один операнд. Выполняю трансляцию с получением файла листинга. Получаем на выводе ошибку, при этом файл создается. Если открыть его, мы увидим, что в файле листинга также обозначена ошибка отсутствия одного операнда (рис. 2.11)

```

ykorshunova@dk8n81 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
lab8-2.asm:25: error: invalid combination of opcode and operands
ykorshunova@dk8n81 ~/work/arch-pc/lab08 $

```

Рис. 2.11: Удаляю один операнд

3 Задания для самостоятельной работы

1. Мой вариант - 4, поэтому, согласно файлу, мои значения для первого задания: 8,88,68. Значит, программа в качестве результата должна выводить число 8. Создаю файл lab8-3.asm и пишу в нем программу. Провожу проверку (рис. 3.1)

```
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
pykorshunova@dk8n81 ~/work/arch-pc/lab08 $ ./lab8-3
Введите В: 88
Наименьшее число: 8
```

Рис. 3.1: Вывод наименьшего числа

2. Создаю файл lab8-4.asm и пишу программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции f(x) и выводит результат вычислений (рис. 3.2) (рис. 3.3)

```
pykorshunova@dk1n22 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
pykorshunova@dk1n22 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
pykorshunova@dk1n22 ~/work/arch-pc/lab08 $ ./lab8-4
Введите x:3
Введите a:0
Функция равна:7
```

Рис. 3.2: Проверка для значений (3;0)

```
pykorshunova@dk1n22 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
pykorshunova@dk1n22 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
pykorshunova@dk1n22 ~/work/arch-pc/lab08 $ ./lab8-4
Введите x:3
Введите a:2
Функция равна:8
```

□

Рис. 3.3: Проверка для значений (3;2)

4 Выводы

В ходе данной лабораторной работы я изучила команды условного и безусловного переходов, приобрела навыки написания программ с использованием переходов и понакомилась с назначением и структурой файла листинга.