

# **Отчет по лабораторной работе №8**

*дисциплина: Архитектура компьютера*

Галацан Николай, НПИбд-01-22

# Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Выполнение заданий для самостоятельной работы	13
4	Выводы	19

## Список иллюстраций

2.1	Редактирование файла lab8-1.asm . . . . .	6
2.2	Трансляция, компоновка, запуск исполняемого файла lab 8-1 . . .	6
2.3	Редактирование файла lab8-1.asm по листингу 8.2 . . . . .	7
2.4	Трансляция, компоновка, запуск измененного исполняемого файла lab 8-1 . . . . .	7
2.5	Редактирование файла lab8-1.asm по инструкции . . . . .	8
2.6	Трансляция, компоновка, запуск исполняемого файла lab 8-1 после 3-го изменения . . . . .	8
2.7	Редактирование файла lab 8-2.asm . . . . .	9
2.8	Трансляция, компоновка, запуск исполняемого файла lab 8-2 . . .	9
2.9	Создание файла листинга lab8-2.lst . . . . .	10
2.10	Открытый файл листинга lab8-2.lst . . . . .	10
2.11	Удаление одного операнда в lab8-2.asm . . . . .	11
2.12	Получение файла листинга измененной программы lab8-2.asm . .	11
2.13	Открытый файл листинга lab8-2.lst измененной программы . . .	12
3.1	Трансляция, компоновка, запуск исполняемого файла lab8-sam-1	15
3.2	Трансляция, компоновка, запуск исполняемого файла lab8-sam-2	17

# 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

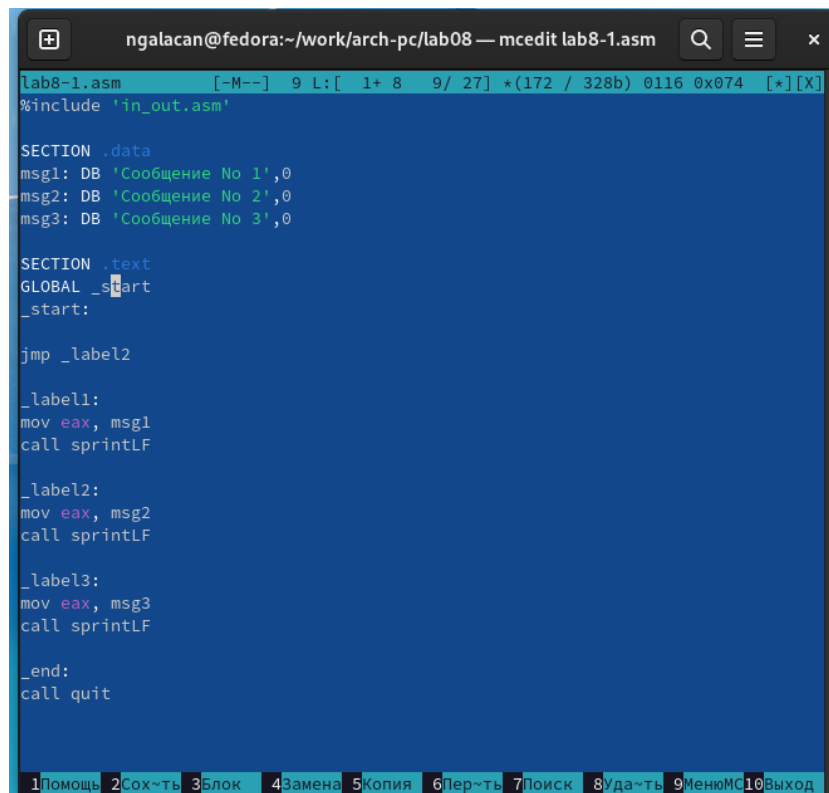
1. Ввожу команды для создания каталога лабораторной работы, перехожу в него, создаю файл `lab8-1.asm`

```
mkdir ~/work/arch-pc/lab08
```

```
cd ~/work/arch-pc/lab08
```

```
touch lab8-1.asm
```

2. Ввожу в файл `lab8-1.asm` текст программы из листинга 8.1, сохраняю файл. (рис. 2.1).



```
lab8-1.asm  [-M--]  9  L:[  1+ 8  9/ 27]  *(172 / 328b)  0116 0x074  [*][X]
#include 'in_out.asm'

SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0

SECTION .text
GLOBAL _start
_start:

jmp _label2

_label1:
mov eax, msg1
call sprintLF

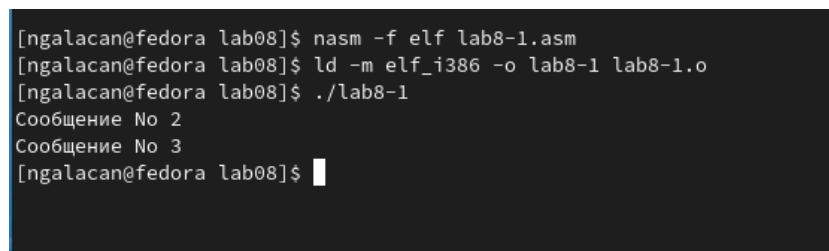
_label2:
mov eax, msg2
call sprintLF

_label3:
mov eax, msg3
call sprintLF

_end:
call quit
```

Рис. 2.1: Редактирование файла lab8-1.asm

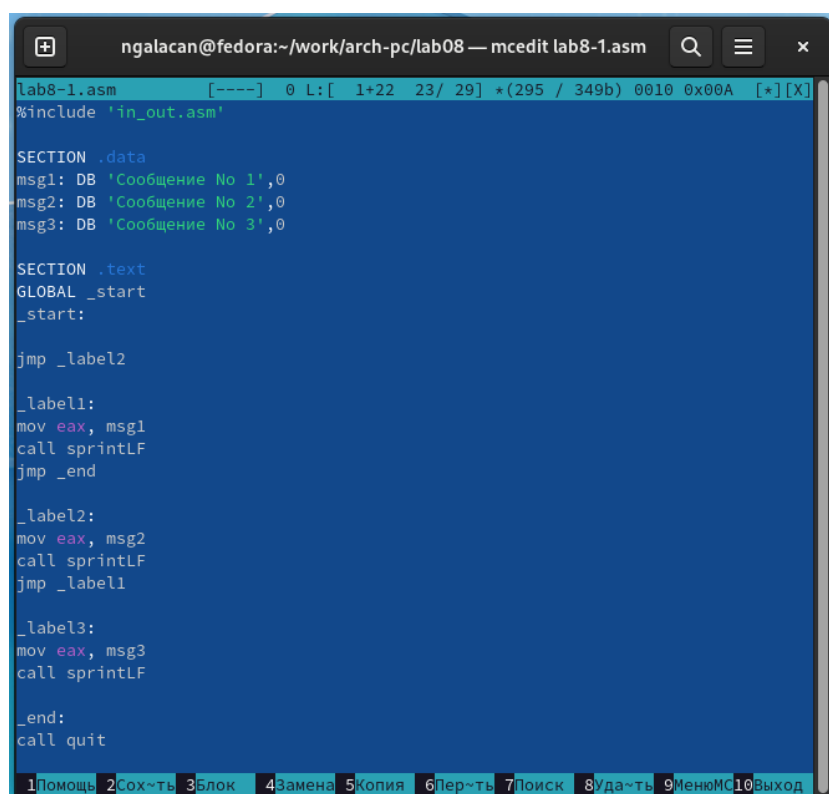
Создаю исполняемый файл и запускаю его, предварительно скопировав файл `in_out.asm` в соответствующий каталог (рис. 2.2).



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

Рис. 2.2: Трансляция, компоновка, запуск исполняемого файла lab 8-1

Результат исполнения программы соответствует результату из инструкции. Далее изменяю текст программы в соответствии с листингом 8.2 (рис. 2.3).



```
lab8-1.asm  [----]  0 L: [ 1+22 23/ 29] *(295 / 349b) 0010 0x00A [*][X]
#include 'in_out.asm'

SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0

SECTION .text
GLOBAL _start
_start:

jmp _label2

_label1:
mov eax, msg1
call sprintf
jmp _end

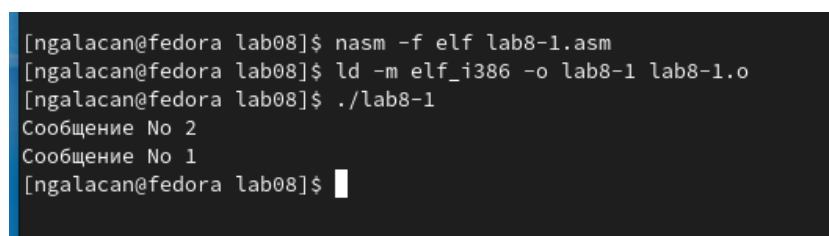
_label2:
mov eax, msg2
call sprintf
jmp _label1

_label3:
mov eax, msg3
call sprintf

_end:
call quit
```

Рис. 2.3: Редактирование файла lab8-1.asm по листингу 8.2

Создаю исполняемый файл и запускаю его (рис. 2.4).

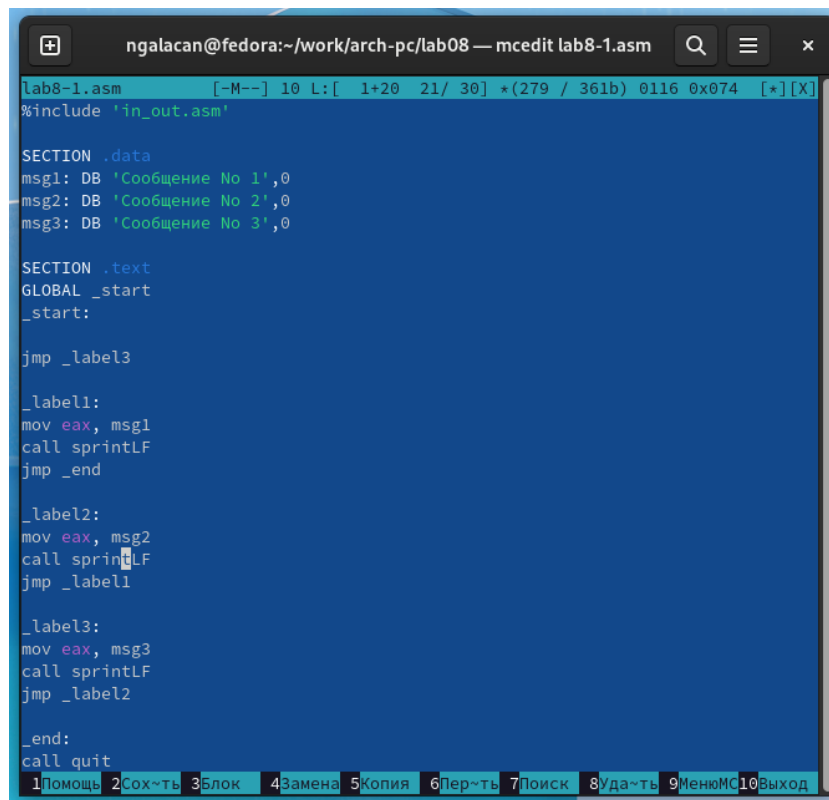


```
[ngalacan@fedora lab08]$ nasm -f elf lab8-1.asm
[ngalacan@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[ngalacan@fedora lab08]$ ./lab8-1
Сообщение No 2
Сообщение No 1
[ngalacan@fedora lab08]$
```

Рис. 2.4: Трансляция, компоновка, запуск измененного исполняемого файла lab 8-1

В результате исполнения программы выводится сначала “Сообщение №2”, а после “Сообщение №1”, так как был применен безусловный переход.

Изменяю текст программы, чтобы сообщения выводились в обратном порядке (от третьего к первому) рис. 2.5).



```
lab8-1.asm      [-M--] 10 L: [ 1+20 21/ 30] *(279 / 361b) 0116 0x074 [*][X]
#include 'in_out.asm'

SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0

SECTION .text
GLOBAL _start
_start:

jmp _label3

_label1:
mov eax, msg1
call sprintLF
jmp _end

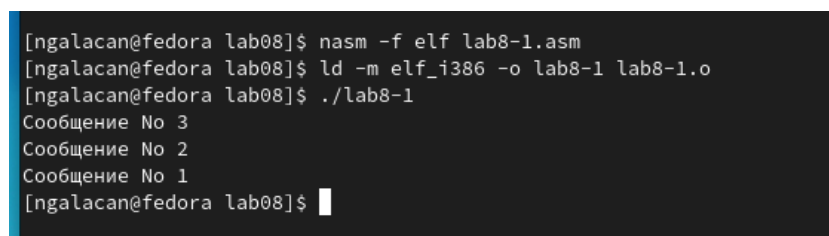
_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

Рис. 2.5: Редактирование файла lab8-1.asm по инструкции

Создаю исполняемый файл и запускаю его (рис. 2.6).



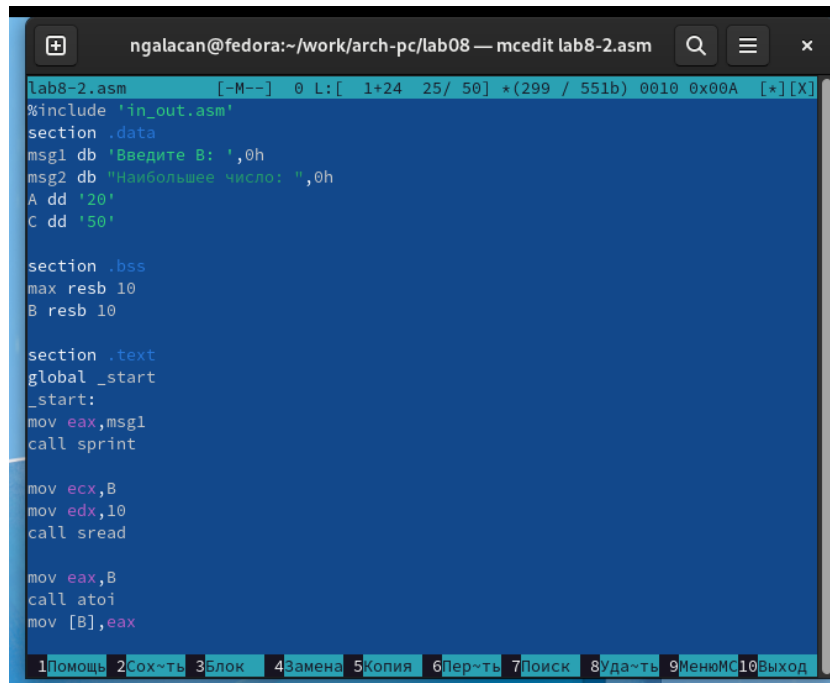
```
[ngalacan@fedora lab08]$ nasm -f elf lab8-1.asm
[ngalacan@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[ngalacan@fedora lab08]$ ./lab8-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
[ngalacan@fedora lab08]$
```

Рис. 2.6: Трансляция, компоновка, запуск исполняемого файла lab 8-1 после 3-го изменения

В результате работы программа выводит сообщения в нужном порядке.

3. Создаю новый файл: `touch lab8-2.asm`. Ввожу в него текст программы из листинга 8.3, внимательно изучив (рис. 2.7).





```
lab8-2.asm [-M--] 0 L: [ 1+24 25/ 50] *(299 / 551b) 0010 0x00A [*][X]
#include 'in_out.asm'
section .data
msg1 db 'Введите B: ',0h
msg2 db "Наибольшее число: ",0h
A dd '20'
C dd '50'

section .bss
max resb 10
B resb 10

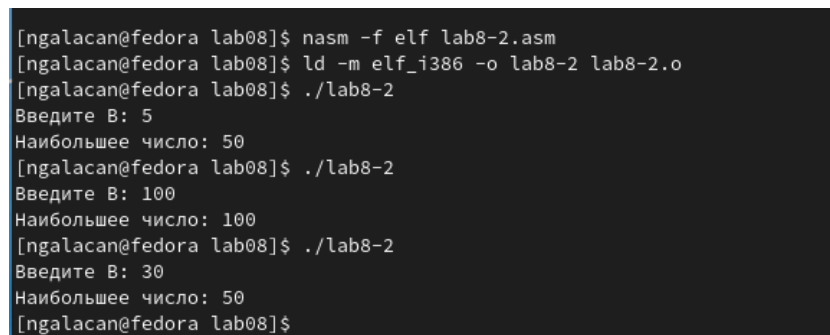
section .text
global _start
_start:
mov eax,msg1
call sprint

mov ecx,B
mov edx,10
call sread

mov eax,B
call atoi
mov [B],eax
```

Рис. 2.7: Редактирование файла lab 8-2.asm

Создаю исполняемый файл и запускаю его. Проверяю работу программы для разных значений *B*. (рис. 2.8).



```
[ngalacan@fedora lab08]$ nasm -f elf lab8-2.asm
[ngalacan@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[ngalacan@fedora lab08]$ ./lab8-2
Введите B: 5
Наибольшее число: 50
[ngalacan@fedora lab08]$ ./lab8-2
Введите B: 100
Наибольшее число: 100
[ngalacan@fedora lab08]$ ./lab8-2
Введите B: 30
Наибольшее число: 50
[ngalacan@fedora lab08]$
```

Рис. 2.8: Трансляция, компоновка, запуск исполняемого файла lab 8-2

4. Ввожу команду для получения файла листинга и открываю его в mcedit(рис. 2.9, рис. 2.10).

```
[ngalacan@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
[ngalacan@fedora lab08]$ ls
in_out.asm  lab8-1.asm  lab8-2     lab8-2.lst
lab8-1      lab8-1.o    lab8-2.asm lab8-2.o
[ngalacan@fedora lab08]$ mcedit lab8-2.lst
```

Рис. 2.9: Создание файла листинга lab8-2.lst

```
lab8-2.lst  [----]  0 L:[188+24 212/226] *(12602/13306b) 0032 0x020  [*][X]
13                                     global _start
14                                     _start:
15 000000E8 B8[00000000]               mov eax,msg1
16 000000ED E81DFFFFFF               call sprint
17 .....
18 000000F2 B9[0A000000]               mov ecx,B
19 000000F7 BA0A000000               mov edx,10
20 000000FC E842FFFFFF               call sread
21 .....
22 00000101 B8[0A000000]               mov eax,B
23 00000106 E891FFFFFF               call atoi
24 0000010B A3[0A000000]               mov [B],eax
25 .....
26 00000110 8B0D[35000000]               mov ecx,[A]
27 00000116 890D[00000000]               mov [max],ecx
28 .....
29 0000011C 3B0D[39000000]               cmp ecx,[C]
30 00000122 7F0C                       jg check_B
31 00000124 8B0D[39000000]               mov ecx,[C]
32 0000012A 890D[00000000]               mov [max],ecx
33 .....
34                                     check_B:
35 00000130 B8[00000000]               mov eax,max
36 00000135 E862FFFFFF               call atoi
37 0000013A A3[00000000]               mov [max],eax
1Помощь 2Сох-ть 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 2.10: Открытый файл листинга lab8-2.lst

## Строки

```
mov ecx,B
mov edx,10
call sread
```

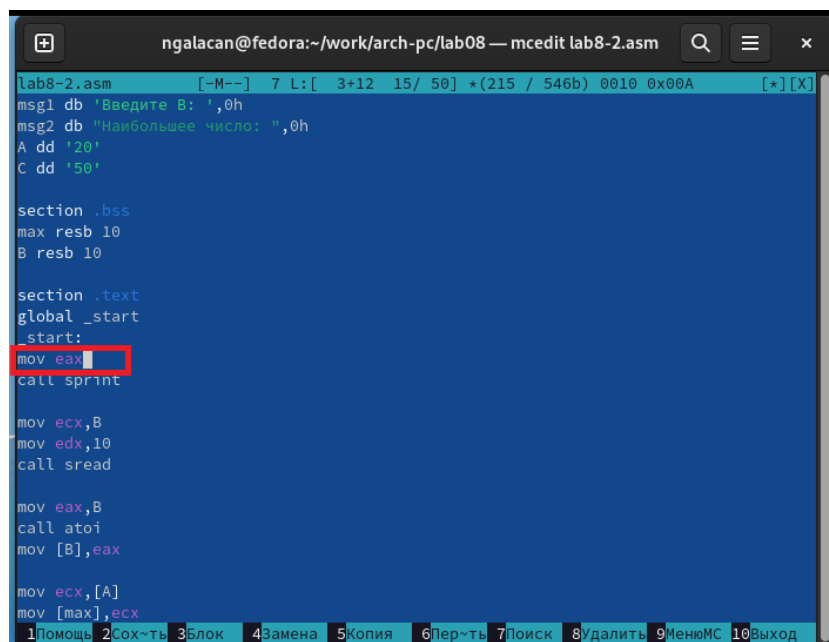
отвечают за считывание переменной *B*. Файл листинга, кроме исходного текста программы, содержит дополнительную информацию:

- в первом столбце указаны номера строк
- во втором столбце указан адрес - смещение машинного кода от начала текущего сегмента

- в третьем столбце - машинный код в шестнадцатеричном представлении, который ассемблируется из исходных строк

Например, инструкция `mov ecx,B` ассемблируется в машинный код `B9[0A000000]`, а адресом является `000000F2`, и все это находится на строке 18. Инструкция `mov edx,10` ассемблируется в машинный код `BA0A000000`, а адресом является `000000F7`, находится на строке 19. Инструкция `call sread` ассемблируется в машинный код `E842FFFFFF`, а адресом является `000000FC`, находится на строке 20.

Открываю файл с программой `lab8-2.asm` и в инструкции `mov eax, msg1` удаляю один операнд (рис. 2.11).



```

lab8-2.asm  [-M--]  7 L:[ 3+12 15/ 50] *(215 / 546b) 0010 0x00A  [+] [X]
msg1 db 'Введите B: ',0h
msg2 db "Наибольшее число: ",0h
A dd '20'
C dd '50'

section .bss
max resb 10
B resb 10

section .text
global _start
_start:
mov eax,
call sprint

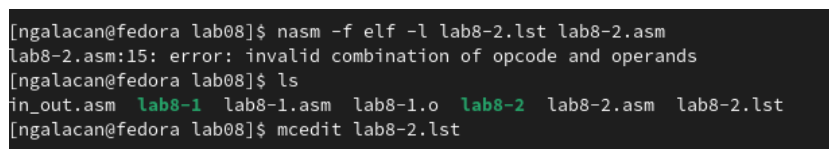
mov ecx,B
mov edx,10
call sread

mov eax,B
call atoi
mov [B],eax

mov ecx,[A]
mov [max],ecx
  
```

Рис. 2.11: Удаление одного операнда в `lab8-2.asm`

Выполняю трансляцию с получением файла листинга (рис. 2.12).

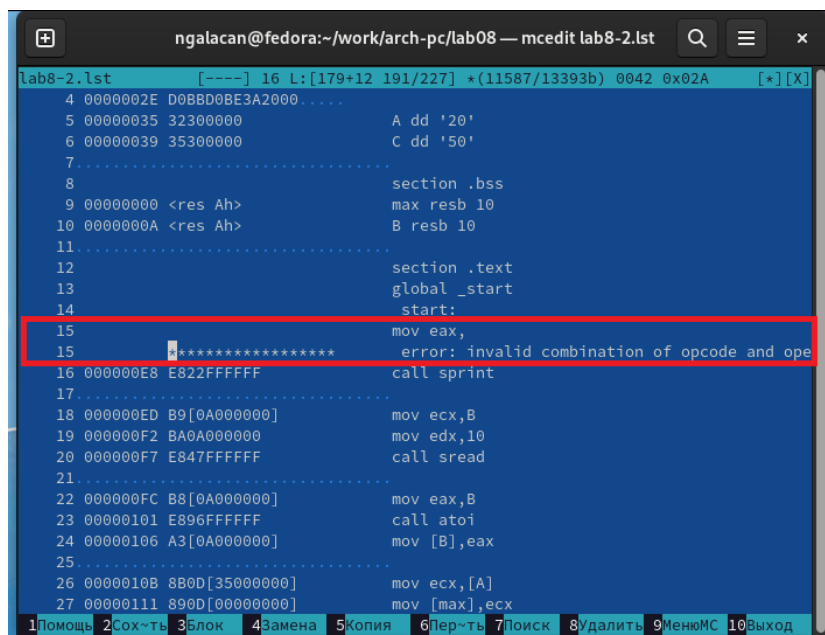


```

[ngalacan@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:15: error: invalid combination of opcode and operands
[ngalacan@fedora lab08]$ ls
in_out.asm  lab8-1  lab8-1.asm  lab8-1.o  lab8-2  lab8-2.asm  lab8-2.lst
[ngalacan@fedora lab08]$ mcedit lab8-2.lst
  
```

Рис. 2.12: Получение файла листинга измененной программы `lab8-2.asm`

Транслятор указывает на ошибку. Не создается объектный файл. Проверяю наличие файла листинга и открываю его (рис. 2.13).



```
lab8-2.lst [----] 16 L:[179+12 191/227] *(11587/13393b) 0042 0x02A [*][X]
4 0000002E D0BBD0BE3A2000.....
5 00000035 32300000          A dd '20'
6 00000039 35300000          C dd '50'
7 .....
8                          section .bss
9 00000000 <res Ah>        max resb 10
10 0000000A <res Ah>       B resb 10
11 .....
12                          section .text
13                          global _start
14                          start:
15                          mov eax,
15                          ***** error: invalid combination of opcode and operand
16 000000E8 E822FFFFFF      call sprint
17 .....
18 000000ED B9[0A000000]        mov ecx,B
19 000000F2 BA0A000000     mov edx,10
20 000000F7 E847FFFFFF      call sread
21 .....
22 000000FC B8[0A000000]        mov eax,B
23 00000101 E896FFFFFF      call atoi
24 00000106 A3[0A000000]        mov [B],eax
25 .....
26 0000010B 8B0D[35000000]     mov ecx,[A]
27 00000111 890D[00000000]     mov [max],ecx
1Помощь 2Сох-ть 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 2.13: Открытый файл листинга lab8-2.lst измененной программы

В листинге добавляется строка с указанием ошибки под строкой, в которой был удален операнд. Раздел с машинным кодом заполнен “\*”, адрес пуст.

## 3 Выполнение заданий для самостоятельной работы

В ЛР №7 был получен 4-ый вариант заданий.

1. Необходимо написать программу для нахождения наименьшей из 3 целочисленных переменных. Значения переменных выбраны из табл. 8.5 в соответствии с вариантом, и равны 8, 88, 68 соответственно.

Создаю файл lab8-sam-1.asm и набираю текст программы. Написанная программа имеет следующий вид:

```
%include 'in_out.asm'

section .data
res db 'Наибольшее число: ',0h
A dd '8'
B dd '88'
C dd '68'

section .bss
max resb 10

section .text
global _start
_start:
```

```
mov ecx,[A] ;запись A в max  
mov [max],ecx
```

```
cmp ecx,[B] ;сравнение A и B  
jg check_C ;если A>B  
mov ecx,[B] ;иначе  
mov [max],ecx
```

```
check_C:  
cmp ecx,[C] ;сравнение большего из A и B с C  
jg fin ;если max>C  
mov ecx,[C]  
mov [max],ecx
```

```
fin:  
mov eax, res  
call sprint
```

```
mov eax,max  
call atoi ;преобразование символа в число  
mov [max],eax
```

```
mov eax,[max]  
call iprintLF  
call quit
```

Создаю исполняемый файл и запускаю, программа выводит верное наибольшее число (рис. 3.1).

```

[ngalacan@fedora lab08]$ nasm -f elf lab8-sam-1.asm
[ngalacan@fedora lab08]$ ld -m elf_i386 -o lab8-sam-1 lab8-sam-1.o
[ngalacan@fedora lab08]$ ./lab8-sam-1
Наибольшее число: 88
[ngalacan@fedora lab08]$

```

Рис. 3.1: Трансляция, компоновка, запуск исполняемого файла lab8-sam-1

2. Необходимо написать программу, которая вычисляет значение функции в соответствии с введенными  $x$  и  $a$  и выводит результат. Вид функции выбран из табл. 8.6 согласно варианту.

Создаю файл lab8-sam-2.asm и набираю текст программы. Для вычисления значения функции в зависимости от  $a$  использую условный переход `je`, переход совершается при выполнении равенства операндов инструкции `cmp`. Также использую команду безусловного перехода `jmp`. Написанная программа имеет следующий вид:

```

%include 'in_out.asm'

section .data
var db 'Вариант 4.',0h
msgx db 'Введите x: ',0h
msga db 'Введите a: ',0h
res db 'Результат: ',0h

section .bss
x resb 10
a resb 10
r resb 10

section .text
global _start
_start:

```

```
mov eax,var  
call sprintf
```

```
mov eax,msgx  
call sprintf
```

```
mov ecx,x  
mov edx,10  
call sread
```

```
mov eax, x  
call atoi  
mov [x],eax
```

```
mov eax,msga  
call sprintf
```

```
mov ecx,a  
mov edx,10  
call sread
```

```
mov eax, a  
call atoi  
mov [a],eax
```

```
mov edx,0  
cmp edx,[a]  
je label1  
mov eax,[x]
```



```

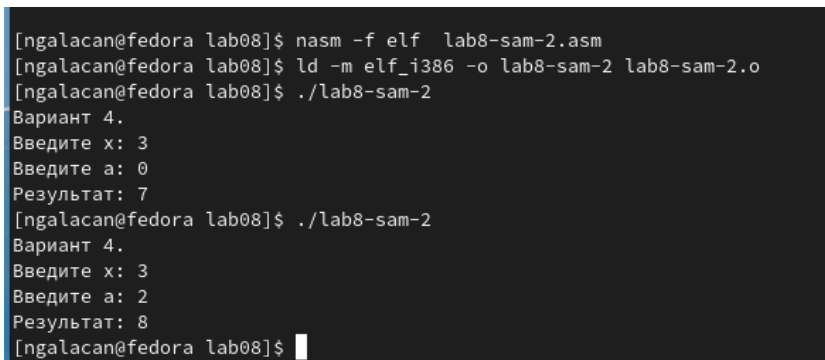
mov ebx,2
mul ebx
add eax, [a]
mov [r],eax
jmp fin

label1:
mov eax,[x]
mov ebx,2
mul ebx ; 2*x
inc eax ; 2*x+1
mov [r],eax

fin:
mov eax, res
call sprint
mov eax,[r]
call iprintLF
call quit

```

Создаю исполняемый файл и запускаю. Проверяю результат для разных  $x$  и  $a$ , взятых из табл. 8.6 (рис. 3.2).



```

[ngalacan@fedora lab08]$ nasm -f elf lab8-sam-2.asm
[ngalacan@fedora lab08]$ ld -m elf_i386 -o lab8-sam-2 lab8-sam-2.o
[ngalacan@fedora lab08]$ ./lab8-sam-2
Вариант 4.
Введите x: 3
Введите a: 0
Результат: 7
[ngalacan@fedora lab08]$ ./lab8-sam-2
Вариант 4.
Введите x: 3
Введите a: 2
Результат: 8
[ngalacan@fedora lab08]$

```

Рис. 3.2: Трансляция, компоновка, запуск исполняемого файла lab8-sam-2

Программа вычисляет верные значения функции.

## 4 Выводы

В ходе лабораторной работы изучены команды условного и безусловного переходов. Приобретены навыки написания программ с использованием переходов. Изучено назначение и структура файла листинга.