

# **Лабораторная работа №8.**

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

Хрусталеv Влад Николаевич

# Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Выводы	15

## Список иллюстраций

2.1	Создание файла lab8-1.asm в соответствующем каталоге . . . . .	5
2.2	Текст программы из листинга 8.1. . . . .	5
2.3	Запуск исполняемого файла lab8-1.asm . . . . .	6
2.4	Текст измененной программы . . . . .	6
2.5	Запуск исправленного исполняемого файла lab8-1.asm . . . . .	6
2.6	Листинг программы с требуемым выводом . . . . .	7
2.7	Запуск измененного исполняемого файла lab8-1.asm . . . . .	7
2.8	Листинга 8.3 . . . . .	8
2.9	Запуск исполняемого файла lab8-2.asm . . . . .	8
2.10	Создание файла листинга . . . . .	9
2.11	Листинг программы . . . . .	10
2.12	Ошибка трансляции в термине . . . . .	10
2.13	Вывод ошибки в листинге . . . . .	10
2.14	Текст программы . . . . .	11
2.15	Текст программы . . . . .	12
2.16	Проверка работы исполняемого файла . . . . .	12
2.17	Листинг программы lab8-4.asm . . . . .	13
2.18	Проверка работы программы . . . . .	14

# 1 Цель работы

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

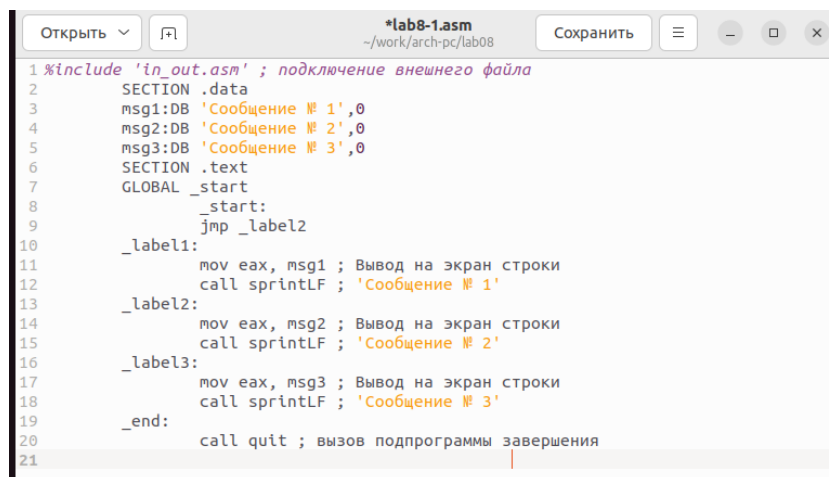
## 2 Выполнение лабораторной работы

Создадим каталог для программ лабораторной работы № 8, перейдем в него и создадим файл lab8-1.asm(рис. 2.1)

```
vlkhrustalev@user:~$ mkdir ~/work/arch-pc/lab08
vlkhrustalev@user:~$ cd ~/work/arch-pc/lab08
vlkhrustalev@user:~/work/arch-pc/lab08$ touch lab8-1.asm
vlkhrustalev@user:~/work/arch-pc/lab08$
```

Рис. 2.1: Создание файла lab8-1.asm в соответствующем каталоге

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



```
Открыть  [F]  *lab8-1.asm  Сохранить  [Menu]  [Zoom]  [Close]
~/work/arch-pc/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     _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
21
```

Рис. 2.2: Текст программы из листинга 8.1.

Создадим исполняемый файл и запустим его.(рис. 2.3)

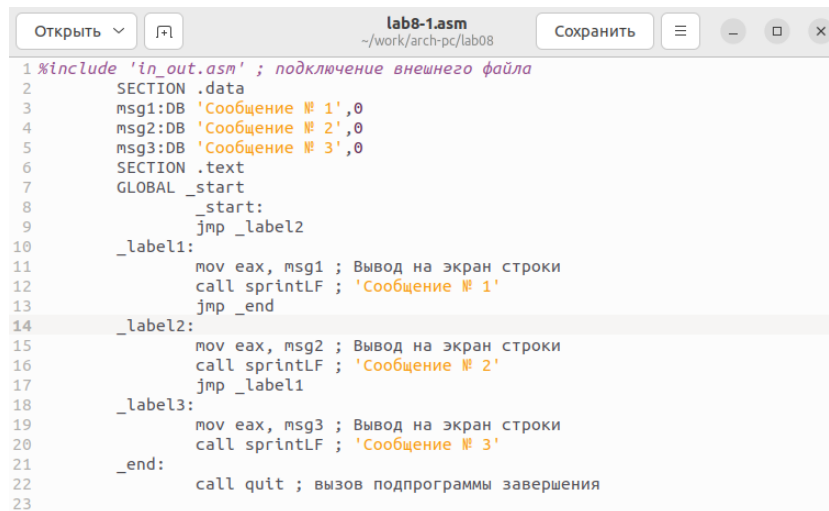
```

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

```

Рис. 2.3: Запуск исполняемого файла lab8-1.asm

Далее изменим текст программы в соответствии с листингом 8.2(рис. 2.4)



```

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 завершения
23

```

Рис. 2.4: Текст измененной программы

Создадим исполняемый файл исправленного текста программы lab8-1.asm и запустим его.(рис. 2.5)

```

vlkhrustalev@user:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
vlkhrustalev@user:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vlkhrustalev@user:~/work/arch-pc/lab08$ ./lab8-1
Сообщение № 2
Сообщение № 1
vlkhrustalev@user:~/work/arch-pc/lab08$

```

Рис. 2.5: Запуск исправленного исполняемого файла lab8-1.asm

Требуемый вывод:

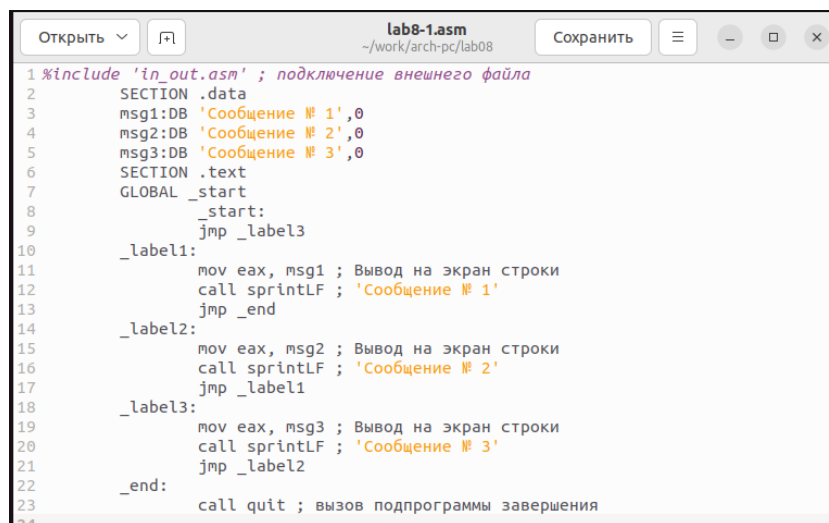
user@dk4n31:~\$ ./lab8-1

Сообщение № 3

Сообщение № 2

Сообщение № 1

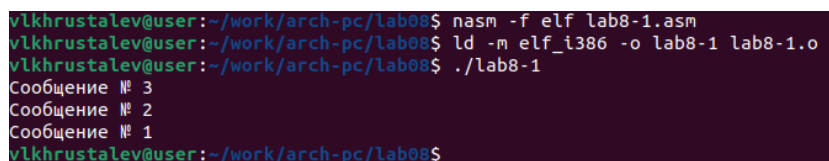
user@dk4n31:~\$(рис. 2.6)



```
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 завершения
24
```

Рис. 2.6: Листинг программы с требуемым выводом

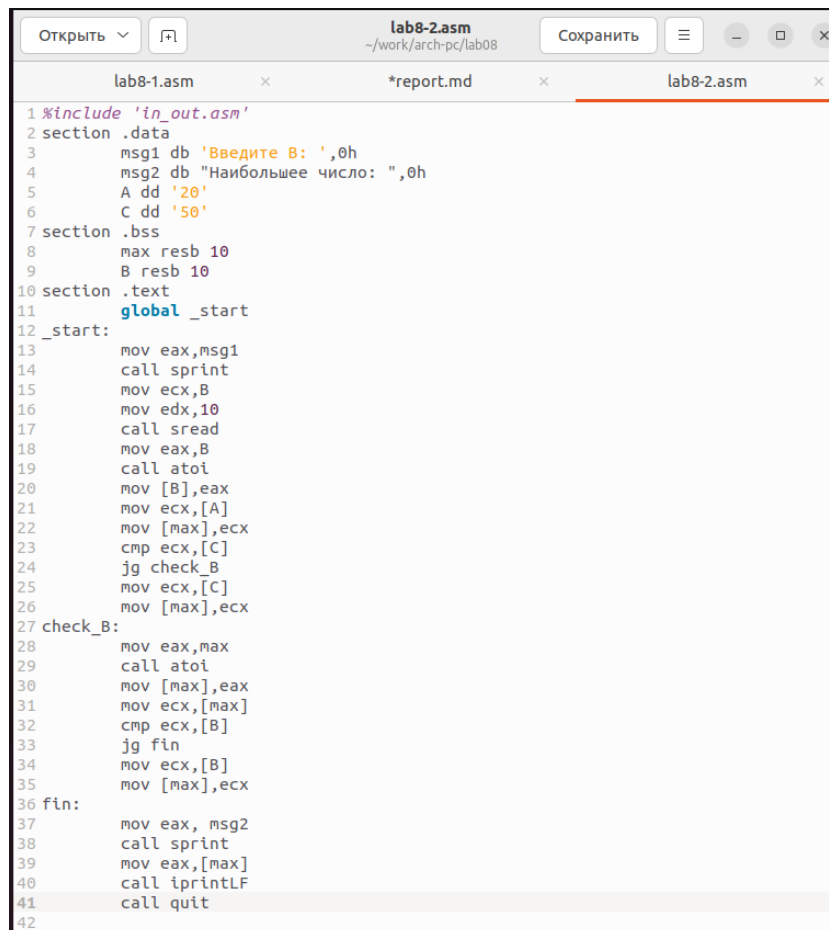
Создадим исполняемый файл и запустим его(рис. 2.7)



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

Рис. 2.7: Запуск измененного исполняемого файла lab8-1.asm

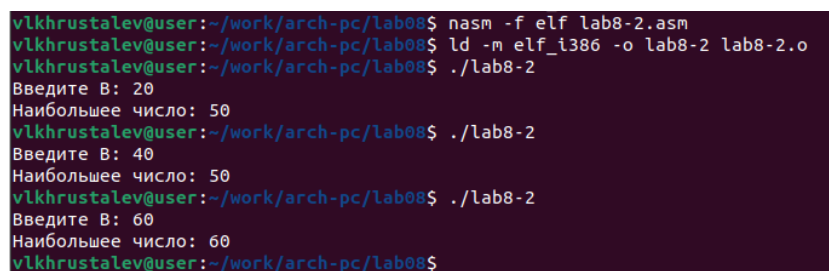
Создадим файл lab8-2.asm в каталоге ~/work/arch-pc/lab08. Внимательно изучим текст программы из листинга 8.3 и введем в lab8-2.asm.(рис. 2.8)



```
1 %include 'in_out.asm'
2 section .data
3     msg1 db 'Введите B: ',0h
4     msg2 db "Наибольшее число: ",0h
5     A dd '20'
6     C dd '50'
7 section .bss
8     max resb 10
9     B resb 10
10 section .text
11     global _start
12 _start:
13     mov eax,msg1
14     call sprint
15     mov ecx,B
16     mov edx,10
17     call sread
18     mov eax,B
19     call atoi
20     mov [B],eax
21     mov ecx,[A]
22     mov [max],ecx
23     cmp ecx,[C]
24     jg check_B
25     mov ecx,[C]
26     mov [max],ecx
27 check_B:
28     mov eax,max
29     call atoi
30     mov [max],eax
31     mov ecx,[max]
32     cmp ecx,[B]
33     jg fin
34     mov ecx,[B]
35     mov [max],ecx
36 fin:
37     mov eax, msg2
38     call sprint
39     mov eax,[max]
40     call iprintfLF
41     call quit
42
```

Рис. 2.8: Листинга 8.3

Создадим измененный исполняемый файл и запустим его(рис. 2.9)



```
vlkhrustalev@user:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
vlkhrustalev@user:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
vlkhrustalev@user:~/work/arch-pc/lab08$ ./lab8-2
Введите B: 20
Наибольшее число: 50
vlkhrustalev@user:~/work/arch-pc/lab08$ ./lab8-2
Введите B: 40
Наибольшее число: 50
vlkhrustalev@user:~/work/arch-pc/lab08$ ./lab8-2
Введите B: 60
Наибольшее число: 60
vlkhrustalev@user:~/work/arch-pc/lab08$
```

Рис. 2.9: Запуск исполняемого файла lab8-2.asm

Создадим файл листинга для программы из файла lab8-2.asm и откроем файл при помощи текстового редактора mcedit(рис. 2.10)



```

vlkhrustalev@user:~/work/arch-pc/lab08$ nasm -f elf -l lab8-2.lst lab8-2.asm
vlkhrustalev@user:~/work/arch-pc/lab08$ mcedit lab8-2.lst
vlkhrustalev@user:~/work/arch-pc/lab08$ █

```

Рис. 2.10: Создание файла листинга

Разберем три строки листинга программы

15 000000F2 B9[0A000000] mov ecx,B

Значение строки:

15-номер строки в коде листинга от начала сегмента

000000F2 - адрес

B9[0A000000] - машинный код(B9[0A000000] - инструкция mov ecx,B; B9 - обозначает что действие производится с регистром ecx, а конкретно mov ecx(в данной программе) ; [0A000000] - ссылка на переменную B)

mov ecx,B - исходный текст программы

16 000000F7 BA0A000000 mov edx,10

Значение строки:

16-номер строки в коде листинга от начала сегмента

000000F7 - адрес

BA0A000000 - машинный код(BA0A000000 - инструкция mov edx,10 ; BA - обозначает что работает с регистром edx)

mov edx,10 - исходный текст программы

17 000000FC E842FFFFFF call sread

Значение строки:

17-номер строки в коде листинга от начала сегмента

000000FC - адрес

E842FFFFFF - машинный код(E842FFFFFF - инструкция call sread, E8 - значит что работает через переменную eax)

call sread - исходный текст программы(рис. 2.11)

```

4 9 0000000A <res An>          b resd 10
5 10                          section .text
5 11                          global _start
7 12                          _start:
3 13 000000E8 B8[00000000]      mov eax,msg1
3 14 000000ED E81DFFFFFF      call sprint
3 15 000000F2 B9[0A000000]      mov ecx,B
1 16 000000F7 BA0A000000      mov edx,10
2 17 000000FC E842FFFFFF      call sread
3 18 00000101 B8[0A000000]      mov eax,B
4 19 00000106 E891FFFFFF      call atoi
5 20 0000010B A3[0A000000]      mov [B],eax
5 21 00000110 0000000000000000

```

Рис. 2.11: Листинг программы

Откроем файл с программой lab8-2.asm и в любой инструкции с двумя операндами удалим один операнд. Выполните трансляцию с получением файла листинга:(рис. 2.12)

```

vlkhrustalev@user:~/work/arch-pc/lab08$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:37: error: invalid combination of opcode and operands
vlkhrustalev@user:~/work/arch-pc/lab08$

```

Рис. 2.12: Ошибка трансляции в терминале

```

08 33 0000014B 7F0C          jg fin
09 34 0000014D 8B0D[0A000000] mov ecx,[B]
10 35 00000153 890D[00000000] mov [max],ecx
11 36                          fin:
12 37                          mov eax
13 37 ***** error: invalid combination of opcode
and operands
14 38 00000159 E8B1FFFFFF      call sprint
15 39 0000015E A1[00000000]      mov eax,[max]
16 40 00000163 E81EFFFFFF      call iprintLF
17 41 00000168 E86EFFFFFF      call quit

```

Рис. 2.13: Вывод ошибки в листинге

На выходе получаем листинг lab8-2.lst с ошибкой 46 \*\*\*\*\* error: invalid combination of opcode and operands

#Самостоятельная работа

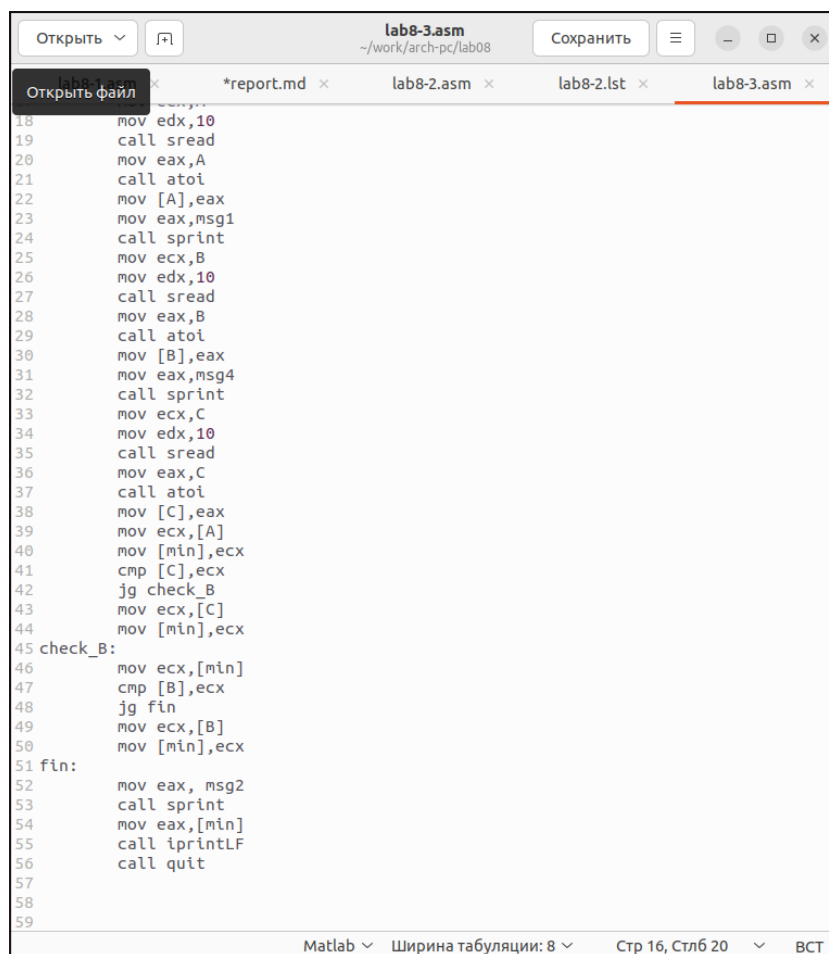
Выполняем вариант номер 12

Напишем программу нахождения наименьшей из 3 целочисленных переменных А, В, С. Значения переменных выберем из табл. 8.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создадим исполняемый файл и проверьте его работу.(рис. 2.14)

```
1 %include 'in_out.asm'
2 section .data
3     msg1 db 'Введите B: ',0h
4     msg2 db "Наименьшее число: ",0h
5     msg3 db 'Введите A: ',0h
6     msg4 db 'Введите C: ',0h
7 section .bss
8     min resb 10
9     B resb 10
10    A resb 10
11    C resb 10
12 section .text
13     global _start
14 _start:
15     mov eax,msg3
16     call sprint
17     mov ecx,A
18     mov edx,10
19     call sread
20     mov eax,A
21     call atoi
22     mov [A],eax
23     mov eax,msg1
24     call sprint
25     mov ecx,B
26     mov edx,10
27     call sread
28     mov eax,B
29     call atoi
30     mov [B],eax
31     mov eax,msg4
32     call sprint
33     mov ecx,C
34     mov edx,10
35     call sread
36     mov eax,C
37     call atoi
38     mov [C],eax
39     mov ecx,[A]
40     mov [min],ecx
41     cmp [C],ecx
42     jg check_B
```

Matlab    Ширина табуляции: 8    Стр 16, Стлб 20    ВСТ

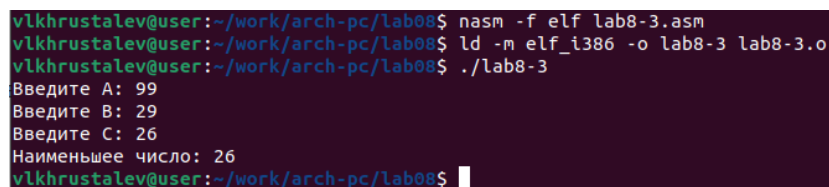
Рис. 2.14: Текст программы



```
18     mov edx,10
19     call sread
20     mov eax,A
21     call atoi
22     mov [A],eax
23     mov eax,msg1
24     call sprint
25     mov ecx,B
26     mov edx,10
27     call sread
28     mov eax,B
29     call atoi
30     mov [B],eax
31     mov eax,msg4
32     call sprint
33     mov ecx,C
34     mov edx,10
35     call sread
36     mov eax,C
37     call atoi
38     mov [C],eax
39     mov ecx,[A]
40     mov [min],ecx
41     cmp [C],ecx
42     jg check_B
43     mov ecx,[C]
44     mov [min],ecx
45 check_B:
46     mov ecx,[min]
47     cmp [B],ecx
48     jg fin
49     mov ecx,[B]
50     mov [min],ecx
51 fin:
52     mov eax, msg2
53     call sprint
54     mov eax,[min]
55     call iprintLF
56     call quit
57
58
59
```

Рис. 2.15: Текст программы

Проверим работу исполняемого файла(рис. 2.16)



```
vlkhrustalev@user:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
vlkhrustalev@user:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
vlkhrustalev@user:~/work/arch-pc/lab08$ ./lab8-3
Введите A: 99
Введите B: 29
Введите C: 26
Наименьшее число: 26
vlkhrustalev@user:~/work/arch-pc/lab08$
```

Рис. 2.16: Проверка работы исполняемого файла

Напишем 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. 2.17)

```
1 %include 'in_out.asm'
2 section .data
3     msg1 db 'Введите a: ',0h
4     msg2 db "Ответ: ",0h
5     msg3 db 'Введите x: ',0h
6 section .bss
7     min resb 10
8     x resb 10
9     a resb 10
10 section .text
11     global _start
12 _start:
13     mov eax,msg3
14     call sprint
15     mov ecx,x
16     mov edx,10
17     call sread
18     mov eax,x
19     call atoi
20     mov [x],eax
21     mov eax,msg1
22     call sprint
23     mov ecx,a
24     mov edx,10
25     call sread
26     mov eax,a
27     call atoi
28     mov [a],eax
29     mov ecx,5
30     cmp [x],ecx
31     jb check_B
32     mov eax,[x]
33     mov ecx,-5
34     add eax,ecx
35     mov [min],eax
36     jmp fin
37 check_B:
38     mov eax,[a]
39     mov ecx,[x]
40     mul ecx
41     mov [min],eax
42 fin:
43     mov eax, msg2
44     call sprint
45     mov eax,[min]
46     call iprintLF
47     call quit
```

Рис. 2.17: Листинг программы lab8-4.asm

```
vlkhrustalev@user:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
vlkhrustalev@user:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
vlkhrustalev@user:~/work/arch-pc/lab08$ ./lab8-4
Введите x: 6
Введите a: 4
Ответ: 1
vlkhrustalev@user:~/work/arch-pc/lab08$ ./lab8-4
Введите x: 3
Введите a: 7
Ответ: 21
vlkhrustalev@user:~/work/arch-pc/lab08$
```

Рис. 2.18: Проверка работы программы

## 3 Выводы

В ходе лабораторной работы мы изучили команды условного и безусловного переходов, приобрели навыки написания программ с использованием переходов, познакомились с назначением и структурой файла листинга.

Ссылка на github: [https://github.com/bezura/study\\_2022-2023\\_arch-pc](https://github.com/bezura/study_2022-2023_arch-pc)