

Отчёта по лабораторной работе №7

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

Камбунду Паулине

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Реализация переходов в NASM	6
3.2	Изучение структуры файлы листинга	10
3.3	Задание для самостоятельной работы	13
4	Выводы	17

Список иллюстраций

3.1	Создаем каталог с помощью команды <code>mkdir</code> и файл с помощью команды <code>touch</code>	6
3.2	Заполняем файл	7
3.3	Запускаем файл и смотрим на его работу	7
3.4	Изменяем файл	8
3.5	Запускаем файл и смотрим на его работу	8
3.6	Редактируем файл	9
3.7	Проверяем, сошелся ли наш вывод с данным в условии выводом .	9
3.8	Создаем файл командой <code>touch</code>	9
3.9	Заполняем файл	10
3.10	Смотрим на работу программ	10
3.11	Создаем файл листинга	10
3.12	Изучаем файл	11
3.13	Удаляем операндум из файла	12
3.14	Транслируем файл	12
3.15	Изучаем файл с ошибкой	13
3.16	Создаем файл командой <code>touch</code>	13
3.17	Пишем программу	14
3.18	Смотрим на работу программы(всё верно)	14
3.19	Создаем файл командой <code>touch</code>	15
3.20	Пишем программу	15
3.21	Проверяем работу программы	15
3.22	Проверяем работу программы	16

1 Цель работы

Освоить условного и безусловного перехода. Ознакомиться с назначением и структурой файла листинга.

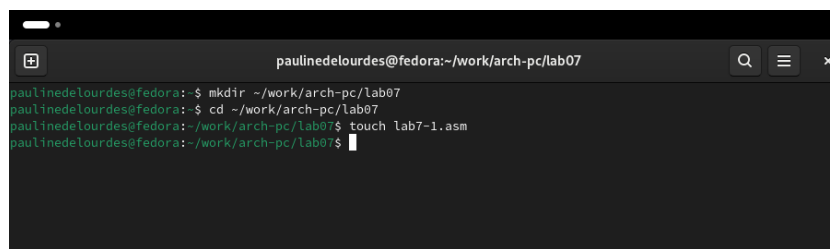
2 Задание

Написать программы для решения системы выражений.

3 Выполнение лабораторной работы

3.1 Реализация переходов в NASM

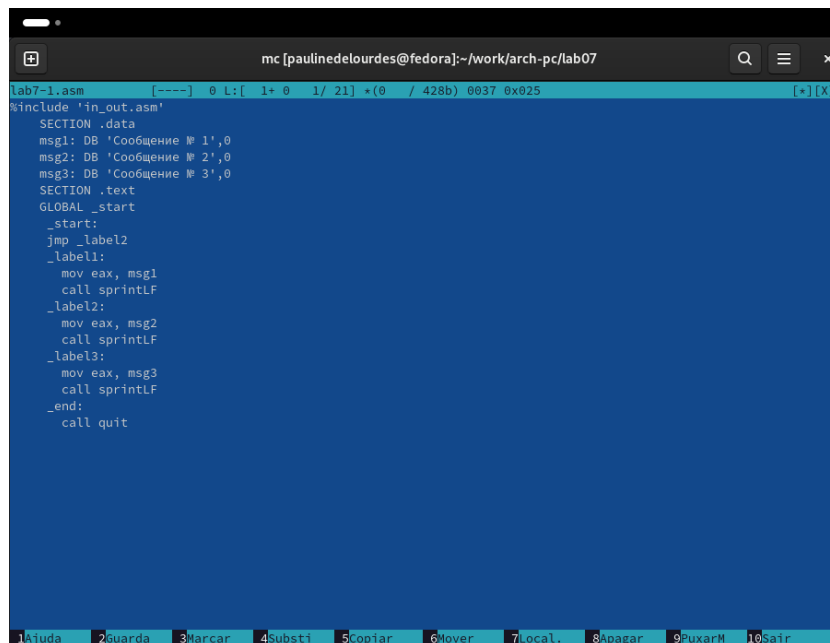
Создаем каталог для программ ЛБ7, и в нем создаем файл (рис. fig. 3.1).



```
paulinedelourdes@fedora:~/work/arch-pc/lab07
paulinedelourdes@fedora:~$ mkdir ~/work/arch-pc/lab07
paulinedelourdes@fedora:~$ cd ~/work/arch-pc/lab07
paulinedelourdes@fedora:~/work/arch-pc/lab07$ touch lab7-1.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.1: Создаем каталог с помощью команды `mkdir` и файл с помощью команды `touch`

Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 7.1 (рис. fig. 3.2).

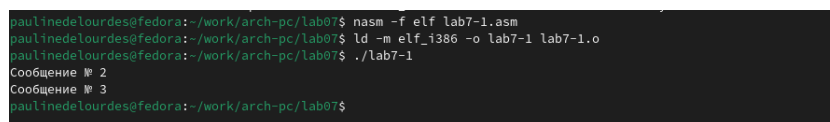


The screenshot shows a text editor window titled 'mc [paulinedelourdes@fedora]:~/work/arch-pc/lab07'. The file 'lab7-1.asm' is open, displaying assembly code. The code includes a data section with three messages and a text section with a loop to print them. The status bar at the bottom shows various editor functions like 'Ajuda', 'Guarda', 'Marcar', etc.

```
lab7-1.asm [----] 0 L: [ 1+ 0 1/ 21] *(0 / 428b) 0037 0x025 [*] [X]
%include 'in_out.asm'
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1
call sprintf
_label2:
mov eax, msg2
call sprintf
_label3:
mov eax, msg3
call sprintf
_end:
call quit
```

Рис. 3.2: Заполняем файл

Создаем исполняемый файл и запускаем его (рис. fig. 3.3).

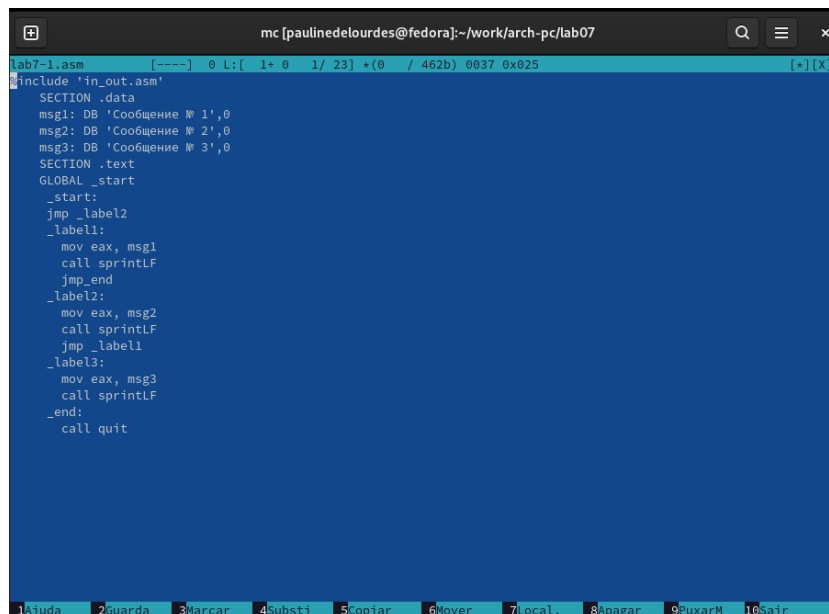


The screenshot shows a terminal window with the following commands and output:

```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.3: Запускаем файл и смотрим на его работу

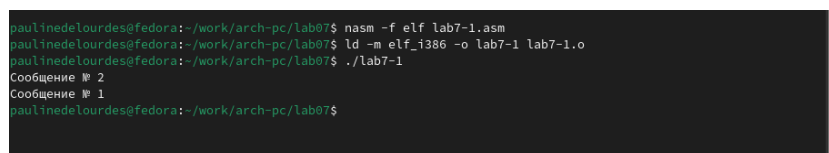
Снова открываем файл для редактирования и изменяем его в соответствии с листингом 7.2 (рис. fig. 3.4).



```
lab7-1.asm [----] 0 L: [ 1+ 0 1/ 23] *(0 / 462b) 0037 0x025 [*] [X]
#include 'in_out.asm'
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 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. 3.4: Изменяем файл

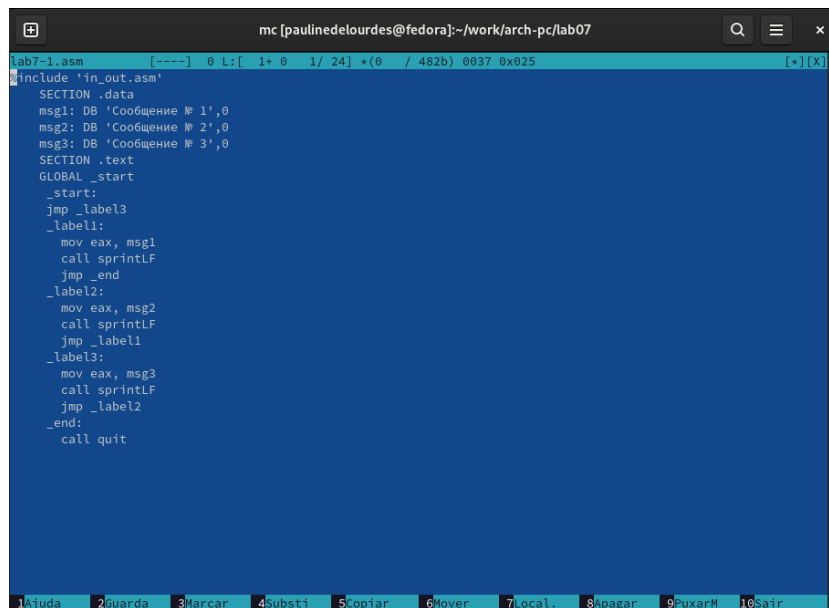
Создаем исполняемый файл и запускаем его (рис. fig. 3.5).



```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.5: Запускаем файл и смотрим на его работу

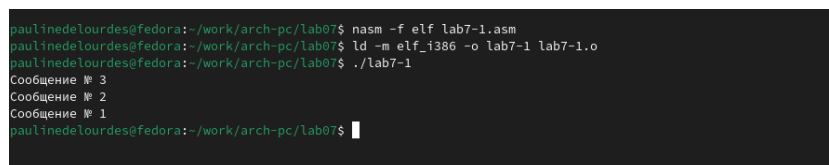
Снова открываем файл для редактирования и изменяем его, чтобы произошел данный вывод (рис. fig. 3.6).

A screenshot of a file editor window titled 'mc [paulinedelourdes@fedora]:~/work/arch-pc/lab07'. The editor shows the contents of 'lab7-1.asm'. The code includes a header section, data section with three messages, and a text section with labels and instructions. The status bar at the bottom shows various icons and a list of actions: 1Ajuda, 2Guarda, 3Marcar, 4Substi, 5Copiar, 6Mover, 7Local, 8Apagar, 9PuxarM, 10Sair.

```
lab7-1.asm [-----] 0 L:[ 1+ 0 1/ 24] *(0 / 482b) 0037 0x025 [*][X]
#include 'in_out.asm'
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1
call sprintf
jmp _end
_label2:
mov eax, msg2
call sprintf
jmp _label1
_label3:
mov eax, msg3
call sprintf
jmp _label2
_end:
call quit
```

Рис. 3.6: Редактируем файл

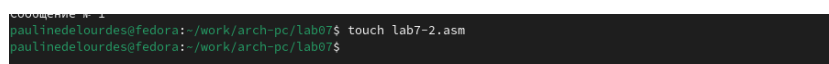
Создаем исполняемый файл и запускаем его (рис. fig. 3.7).

A screenshot of a terminal window showing the compilation and execution of the assembly file. The user runs 'nasm -f elf lab7-1.asm', then 'ld -m elf_i386 -o lab7-1 lab7-1.o', and finally './lab7-1'. The output shows three messages in Russian: 'Сообщение № 3', 'Сообщение № 2', and 'Сообщение № 1'.

```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.7: Проверяем, сошелся ли наш вывод с данным в условии выводом

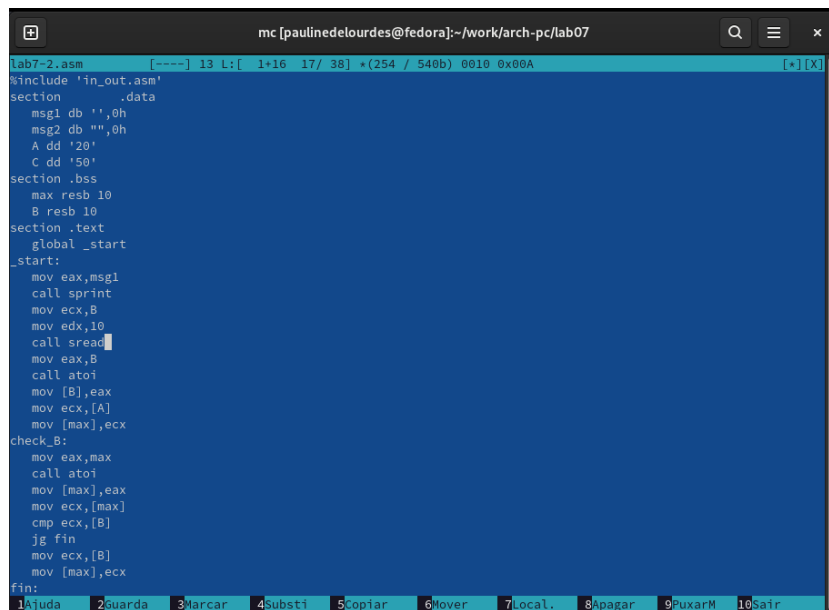
Создаем новый файл (рис. fig. 3.8).

A screenshot of a terminal window showing the creation of a new file 'lab7-2.asm' using the 'touch' command. The user runs 'touch lab7-2.asm' and the prompt returns.

```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ touch lab7-2.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.8: Создаем файл командой touch

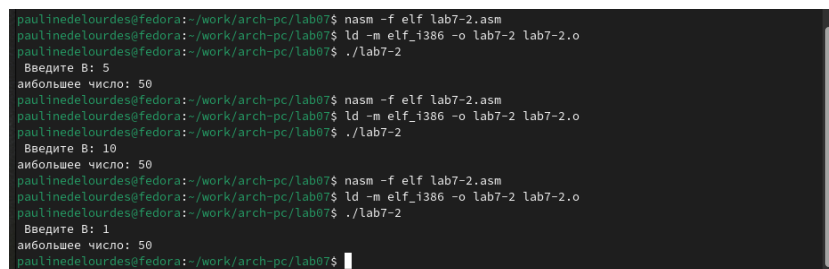
Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 7.3 (рис. fig. 3.9).



```
lab7-2.asm [-----] 13 L: [ 1+16 17/ 38 ] *(254 / 540b) 0010 0x00A [*] [X]
#include 'in_out.asm'
section .data
    msg1 db "",0h
    msg2 db "",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,8
    mov edx,10
    call sread
    mov eax,B
    call atoi
    mov [B],eax
    mov ecx,[A]
    mov [max],ecx
check_B:
    mov eax,max
    call atoi
    mov [max],eax
    mov ecx,[max]
    cmp ecx,[B]
    jg fin
    mov ecx,[B]
    mov [max],ecx
fin:
```

Рис. 3.9: Заполняем файл

Создаем исполняемый файл и проверяем его работу, вводя разные значения B (рис. fig. 3.10).



```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 5
айбольшее число: 50
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 10
айбольшее число: 50
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 1
айбольшее число: 50
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.10: Смотрим на работу программ

3.2 Изучение структуры файлы листинга

Создаем файл листинга для программы lab7-2.asm (рис. fig. 3.11).



```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.11: Создаем файл листинга

Открываем файл листинга с помощью команды mcedit и изучаем его (рис.

fig. 3.12).

```
mc [paulinelourdes@fedora] ~/work/arch-pc/lab07
lab7-2.lst      [----]  0 L: [ 1+ 0  1/217] * (0  /13845b) 0032 0x020      [*] (X)
1               %include 'in_out.asm'
2               <1> ;----- slen -----
3               <1> ; Функция вычисления длины сообщения
4               <1> slen:
5               <1>   push    ebx .....
6               <1>   mov     ebx, eax .....
7               <1>
8               <1>   nextchar: .....
9               <1>   cmp     byte [eax], 0...
10              <1>   jz      finished .....
11              <1>   inc     eax .....
12              <1>   jmp     nextchar .....
13              <1>
14              <1>   finished: .....
15              <1>   sub     eax, ebx .....
16              <1>   pop     ebx .....
17              <1>   ret     .....
18              <1>
19              <1> ;----- sprintf -----
20              <1> ; Функция печати сообщения
21              <1> ; входные данные: mov eax, <message>
22              <1> sprintf:
23              <1>   push    edx .....
24              <1>   push    ecx .....
25              <1>   push    ebx .....
26              <1>   push    eax .....
27              <1>   call    slen .....
28              <1>
29              <1>   mov     edx, eax .....
30              <1>   pop     eax .....
31              <1>
32              <1>   mov     ecx, eax .....
33              <1>   mov     ebx, 1 .....
34              <1>   mov     eax, 4 .....
35              <1>   int     80h .....
36              <1>
37              <1>   pop     ebx .....
38              <1>   pop     ecx .....
39              <1>   pop     edx .....
40              <1>   ret     .....
41              <1>
42              <1>
43              <1> ;----- sprintfLF -----
44              <1> ; Функция печати сообщения с переводом строки
45              <1> ; входные данные: mov eax, <message>
```

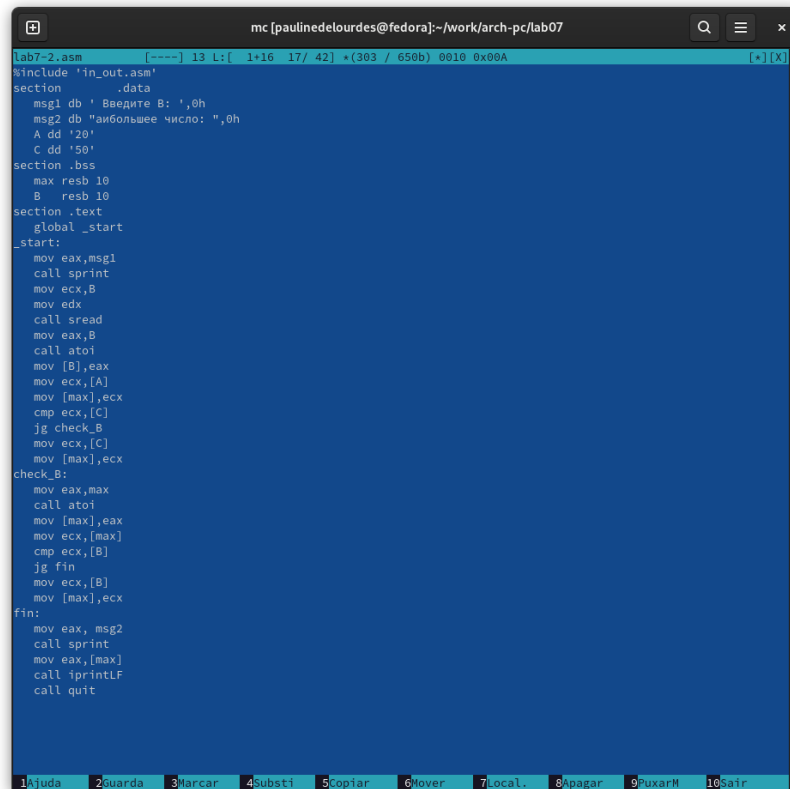
Рис. 3.12: Изучаем файл

Строка 33: 0000001D-адрес в сегменте кода, BB01000000-машинный код, mov
ebx,1-присвоение переменной ebx значения 1.

Строка 34: 00000022-адрес в сегменте кода, B804000000-машинный код, mov
eax,4-присвоение переменной eax значения 4.

Строка 35 00000027-адрес в сегменте кода, CD80-машинный код, int 80h-вызов ядра.

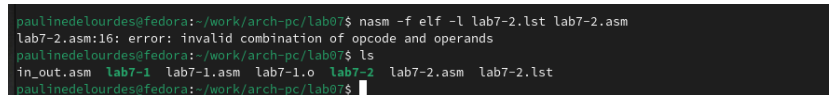
Открываем файл и удаляем один операндум (рис. fig. 3.13).



```
lab7-2.asm [-----] 13 L: 1+16 17/ 42] *(303 / 650b) 0010 0x00A [*)(X]
#include "in_out.asm"
section .data
    msg1 db "Введите B: ",0h
    msg2 db "аибольшее 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
    call sread
    mov eax,B
    call atoi
    mov [B],eax
    mov ecx,[A]
    mov [max],ecx
    cmp ecx,[C]
    jg check_B
    mov ecx,[C]
    mov [max],ecx
check_B:
    mov eax,max
    call atoi
    mov [max],eax
    mov ecx,[max]
    cmp ecx,[B]
    jg fin
    mov ecx,[B]
    mov [max],ecx
fin:
    mov eax, msg2
    call sprint
    mov eax,[max]
    call iprintlnLF
    call quit
```

Рис. 3.13: Удаляем операндум из файла

Транслируем с получением файла листинга (рис. fig. 3.14).



```
paulinedelourdes@fedora: ~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:16: error: invalid combination of opcode and operands
paulinedelourdes@fedora: ~/work/arch-pc/lab07$ ls
in_out.asm lab7-1 lab7-1.asm lab7-1.o lab7-2 lab7-2.asm lab7-2.lst
paulinedelourdes@fedora: ~/work/arch-pc/lab07$
```

Рис. 3.14: Транслируем файл

При трансляции файла, выдается ошибка, но создаются исполнительный файл lab7-2 и lab7-2.lst

Снова открываем файл листинга и изучаем его (рис. fig. 3.15).

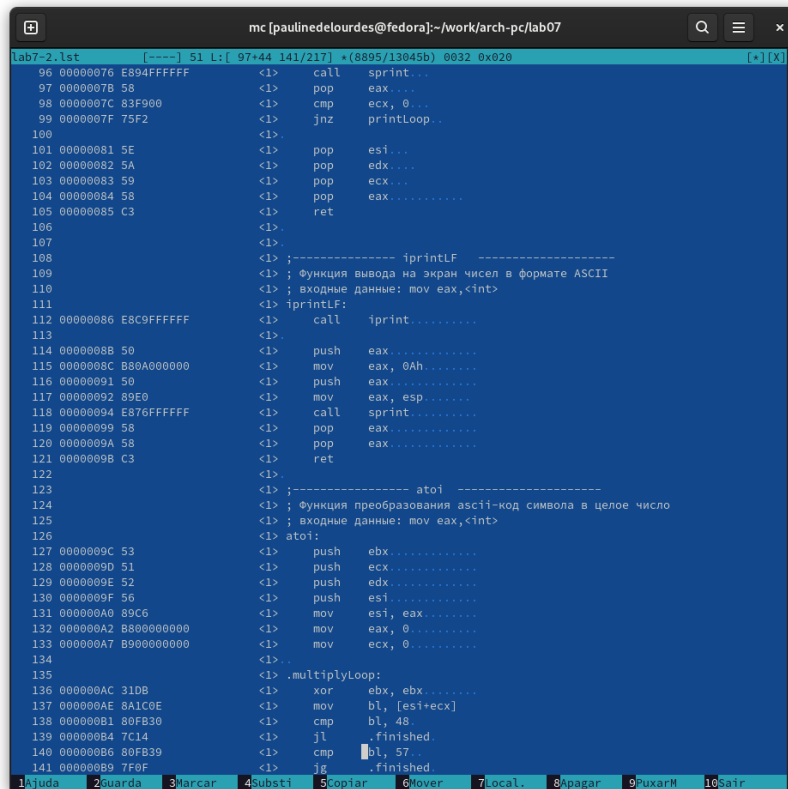


Рис. 3.15: Изучаем файл с ошибкой

3.3 Задание для самостоятельной работы

ВАРИАНТ-20

1. Напишите программу нахождения наименьшей из 3 целочисленных переменных \boxtimes, \boxtimes и с. Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу.

Создаем новый файл (рис. fig. 3.16).

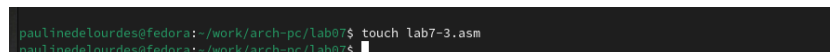


Рис. 3.16: Создаем файл командой touch

Открываем его и пишем программу, которая выберет наименьшее число из трех(2 числа уже в программе, 3е вводится из консоли) (рис. fig. 3.17).

```

lab7-3.asm  [-----]  0 L: 1+ 0 1/ 43] *0 / 656b 0105 0x069 [*)(X]
#include 'in_out.asm'
section .data
    msg1 db 'Введите B: ',0h
    msg2 db "наименьшее число: ",0h
    A dd '95'
    C dd '61'
section .bss
    min 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
    mov ecx,[A]
    mov [min],ecx
    cmp ecx,[C]
    jlt check_B
    mov ecx,[C]
    mov [min],ecx
check_B:
    mov eax,min
    call atoi
    mov [min],eax
    mov ecx,[min]
    cmp ecx,[B]
    jlt fin
    mov ecx,[B]
    mov [min],ecx
fin:
    mov eax, msg2
    call sprint
    mov eax,[min]
    call iprintLF
    call quit
  
```

Рис. 3.17: Пишем программу

Транслируем файл и смотрим на работу программы (рис. fig. 3.18).

```

paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-3
Введите B: 2
наименьшее число: 2
paulinedelourdes@fedora:~/work/arch-pc/lab07$
  
```

Рис. 3.18: Смотрим на работу программы(всё верно)

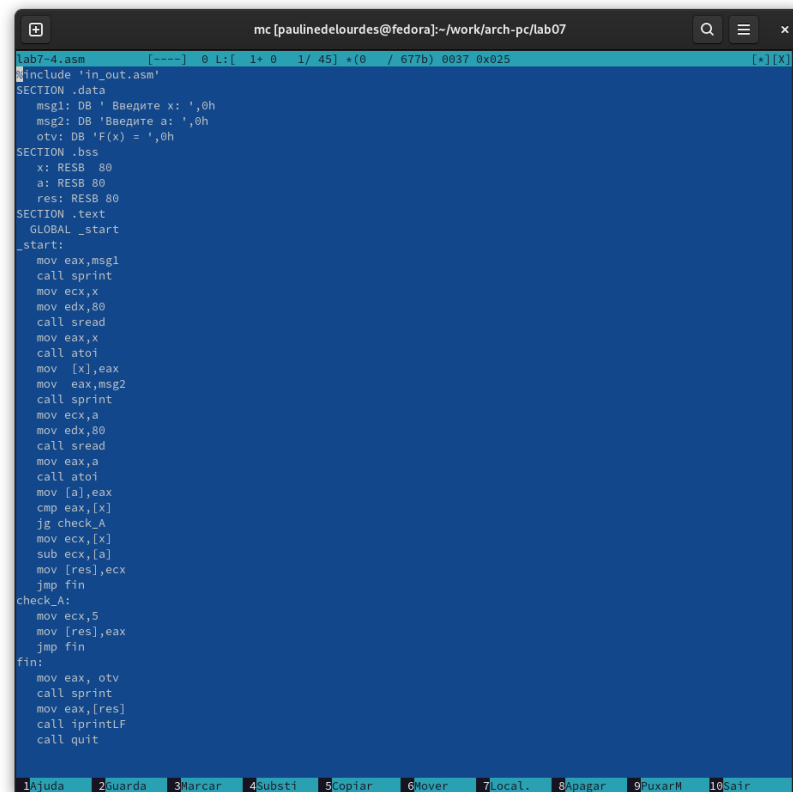
2. Напишите программу, которая для введенных с клавиатуры значений x и y вычисляет значение заданной функции $f(x, y)$ и выводит результат вычислений. Вид функции $f(x, y)$ выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений x и y из 7.6.

Создаем новый файл (рис. fig. 3.19).

```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ touch lab7-4.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.19: Создаем файл командой touch

Открываем его и пишем программу, которая решит систему уравнений, при данных, введенных в консоль (рис. fig. 3.20).



```
lab7-4.asm [-----] 0 L: 1+ 0 1/ 45 *(0 / 677b) 0037 0x025 [*][X]
#include 'in_out.asm'
SECTION .data
msg1: DB 'Введите x: ',0h
msg2: DB 'Введите a: ',0h
otv: DB 'F(x) = ',0h
SECTION .bss
x: RESB 80
a: RESB 80
res: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,msg1
call sprint
mov ecx,x
mov edx,80
call sread
mov eax,x
call atoi
mov [x],eax
mov eax,msg2
call sprint
mov ecx,a
mov edx,80
call sread
mov eax,a
call atoi
mov [a],eax
cmp eax,[x]
jg check_A
mov ecx,[x]
sub ecx,[a]
mov [res],ecx
jmp fin
check_A:
mov ecx,5
mov [res],eax
jmp fin
fin:
mov eax,otv
call sprint
mov eax,[res]
call iprintlnLF
call quit
```

Рис. 3.20: Пишем программу

Транслируем файл и проверяем его работу при $x=1$ и $a=2$ (рис. fig. 3.21).

```
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 1
Введите a: 2
F(x) = 2
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.21: Проверяем работу программы

Транслируем файл и проверяем его работу при $x=2$ и $a=1$ (рис. fig. 3.22).

```
F(x) = 2
paulinedelourdes@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
paulinedelourdes@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 2
Введите a: 1
F(x) = 1
paulinedelourdes@fedora:~/work/arch-pc/lab07$
```

Рис. 3.22: Проверяем работу программы

4 Выводы

Мы познакомились с структурой файла листинга, изучили команды условного и безусловного перехода.