

# **Отчет по лабораторной работе №7**

**Дисциплина: Архитектура компьютера**

Рамазанов Абуталим Абдулмеджидович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>

## Список иллюстраций

2.1	Создание каталога и файла . . . . .	6
2.2	Заполняем файл . . . . .	6
2.3	Смотрим результат . . . . .	7
2.4	Изменяем файл . . . . .	7
2.5	Запускаем файл . . . . .	8
2.6	Создаем и заполняем файл . . . . .	9
2.7	Запускаем файл . . . . .	9
2.8	Создание листинга для программы и его запуск . . . . .	10
2.9	Убираем операндум . . . . .	11
2.10	Трансляция . . . . .	11
2.11	Запускаем листинг . . . . .	12
2.12	Новый файл . . . . .	12
2.13	Пишем программу . . . . .	13
2.14	Запуск программы . . . . .	13
2.15	Новый файл . . . . .	13
2.16	Пишем программу . . . . .	14
2.17	Запуск программы . . . . .	14

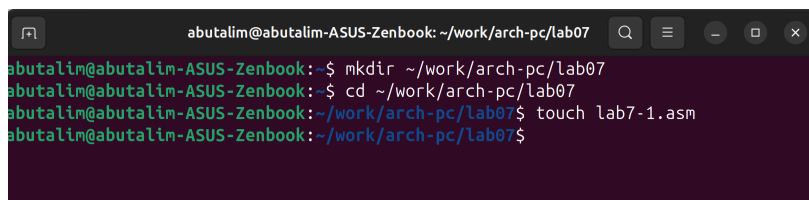
## **Список таблиц**

# 1 Цель работы

Освоение команд условного и безусловного переходов. Нарботка умений кодирования программ при содействии переходов. Ознакомление с ролью и форматом файла листинга.

## 2 Выполнение лабораторной работы

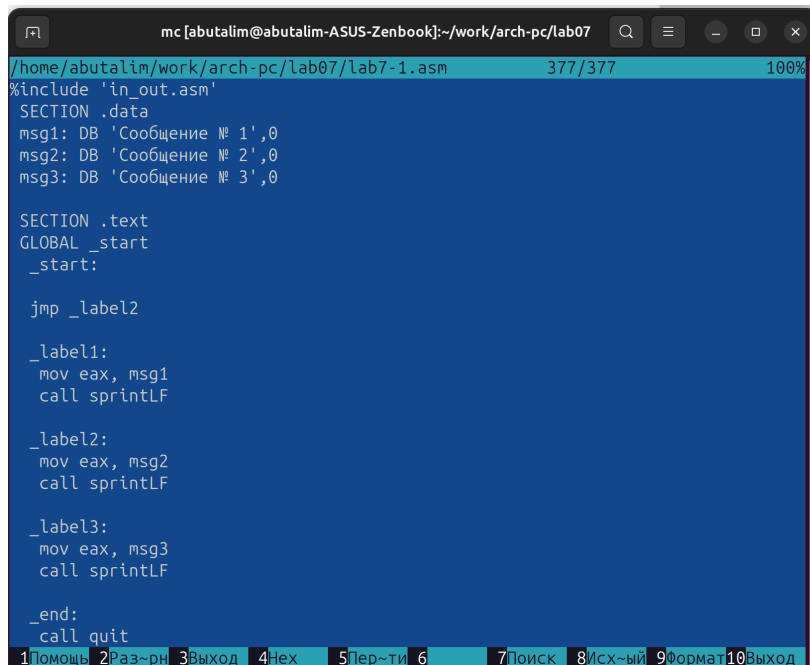
Создаем каталог для программ лабораторной работы № 7, перейдем в него и создадим файл lab7-1.asm:



```
abutalim@abutalim-ASUS-Zenbook: ~/work/arch-pc/lab07
abutalim@abutalim-ASUS-Zenbook:~$ mkdir ~/work/arch-pc/lab07
abutalim@abutalim-ASUS-Zenbook:~$ cd ~/work/arch-pc/lab07
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ touch lab7-1.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.1: Создание каталога и файла

Заполняем файл в соответствии с листингом 7.1



```
mc [abutalim@abutalim-ASUS-Zenbook]:~/work/arch-pc/lab07
/home/abutalim/work/arch-pc/lab07/lab7-1.asm 377/377 100%
%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 sprintfLF

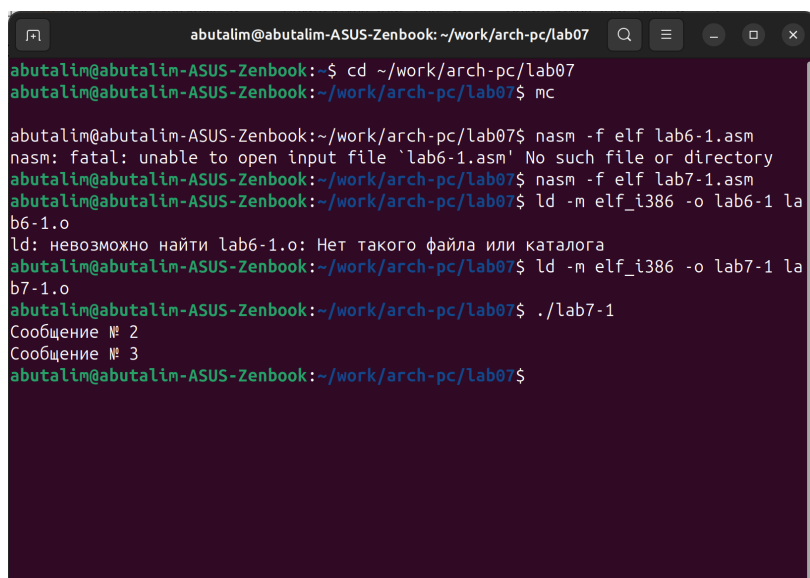
_label2:
mov eax, msg2
call sprintfLF

_label3:
mov eax, msg3
call sprintfLF

_end:
call quit
```

Рисунок 2.2: Заполняем файл

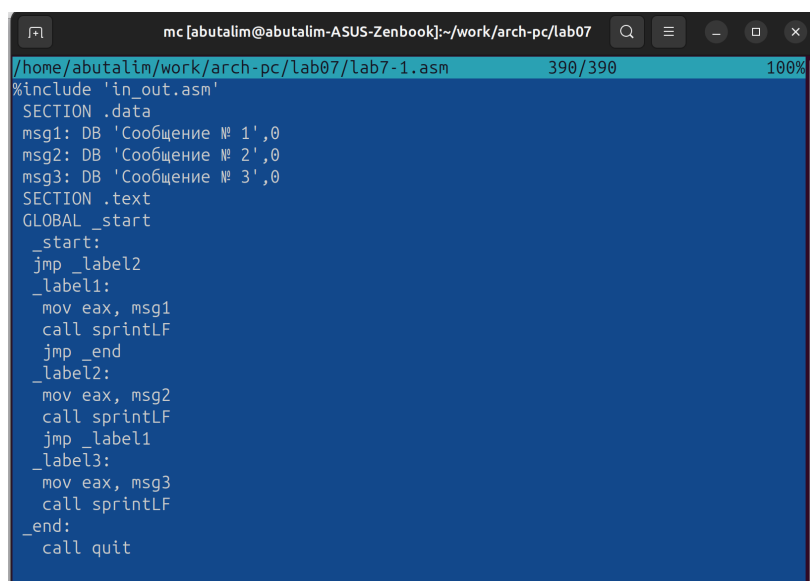
Проверяем результат при запуске исполняемого файла



```
abutalim@abutalim-ASUS-Zenbook: ~/work/arch-pc/lab07
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ cd ~/work/arch-pc/lab07
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ mc
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf lab6-1.asm
nasm: fatal: unable to open input file 'lab6-1.asm' No such file or directory
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab6-1 la
b6-1.o
ld: невозможно найти lab6-1.o: Нет такого файла или каталога
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 la
b7-1.o
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.3: Смотрим результат

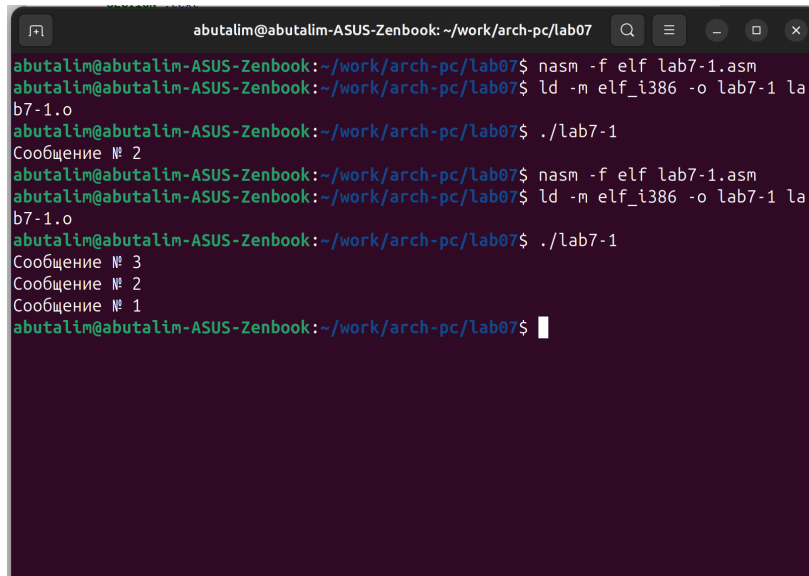
Изменяем файл в соответствии с листингом 7.2



```
mc [abutalim@abutalim-ASUS-Zenbook]:~/work/arch-pc/lab07
/home/abutalim/work/arch-pc/lab07/lab7-1.asm 390/390 100%
%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
```

Рисунок 2.4: Изменяем файл

Запускаем и проверяем его работу

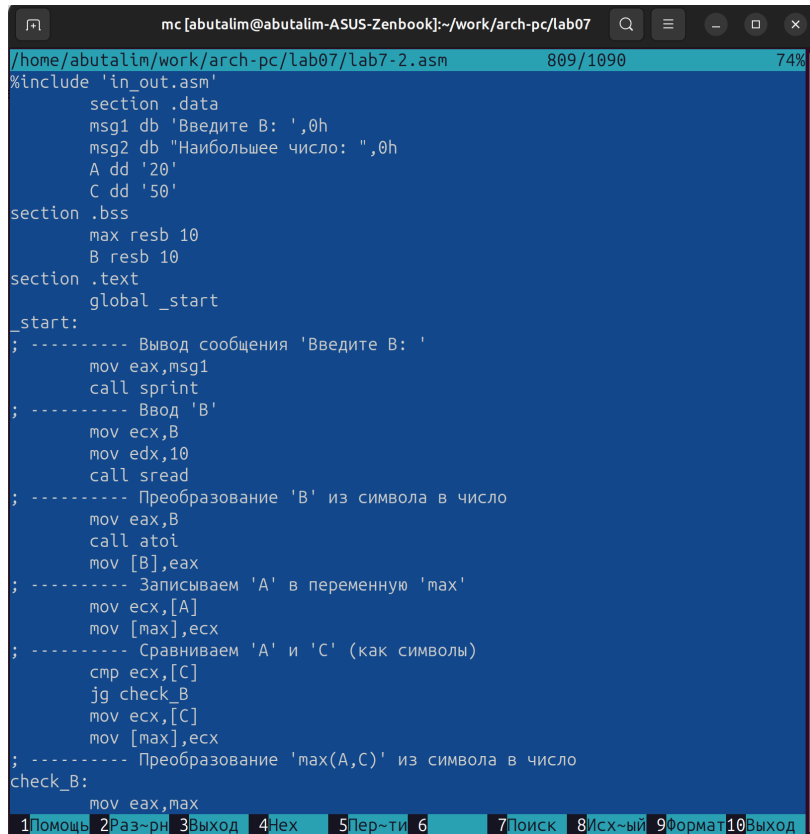


```
abutalim@abutalim-ASUS-Zenbook: ~/work/arch-pc/lab07
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 la
b7-1.o
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 la
b7-1.o
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.5: Запускаем файл

Создаем файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. Заполнем в соответствии с листингом 7.2

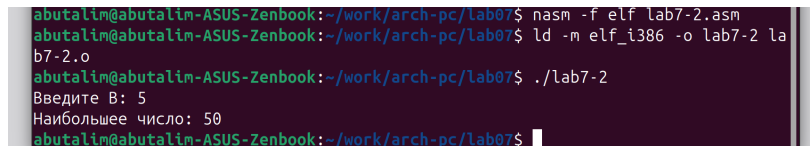




```
mc [abutalim@abutalim-ASUS-Zenbook]:~/work/arch-pc/lab07
/home/abutalim/work/arch-pc/lab07/lab7-2.asm 809/1090 74%
%include 'in_out.asm'
    section .data
        msg1 db 'Введите B: ',0h
        msg2 db "Наибольшее число: ",0h
        A dd '20'
        C dd '50'
    section .bss
        max resb 10
        B resb 10
    section .text
        global _start
_start:
; ----- Вывод сообщения 'Введите B: '
        mov eax,msg1
        call sprint
; ----- Ввод 'B'
        mov ecx,B
        mov edx,10
        call sread
; ----- Преобразование 'B' из символа в число
        mov eax,B
        call atoi
        mov [B],eax
; ----- Записываем 'A' в переменную 'max'
        mov ecx,[A]
        mov [max],ecx
; ----- Сравниваем 'A' и 'C' (как символы)
        cmp ecx,[C]
        jg check_B
        mov ecx,[C]
        mov [max],ecx
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
        mov eax,max
1Помощь 2Раз-рн 3Выход 4Нех 5Пер-ти 6 7Поиск 8Исх-ый 9Формат10Выход
```

Рисунок 2.6: Создаем и заполняем файл

Запускаем файл lab7-2.asm



```
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 5
Наибольшее число: 50
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.7: Запускаем файл

Создаем файл листинга для программы из файла lab7-2.asm указав ключ -l и запускаем его с помощью команды mcedit lab7-2.lst

```

/home/abutalim-07/lab7-2.lst  [----]  0 1: 1+ 0 1/225] *(0 /13804b) 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> ;----- sprint -----
20                              <1> ; Функция печати сообщения
21                              <1> ; входные данные: mov eax, <message>
22                              <1> sprint:
23                              <1> push     edx

```

Рисунок 2.8: Создание листинга для программы и его запуск

Строка 33: 0000001D — адрес в сегменте кода, B01000000 — 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.

Откроем файл с программой `lab7-2.asm` и в любой инструкции с двумя операндами удаляем один операнд.

```
GNU nano 7.2 /home/abutalim/work/arch-pc/lab07/lab7-2.asm *
msg2 db "Наибольшее число: ",0h
A dd '20'
C dd '50'
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
; ----- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
; ----- Ввод 'B'
mov ecx,B
mov edx,
call sread
; ----- Преобразование 'B' из символа в число
mov eax,B
call atoi
mov [B],eax

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Позиция
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^J Выровнять ^/ К строке
```

Рисунок 2.9: Убираем операндум

Выполняем трансляцию с получением файла листинга:

```
abutalim@abutalim-ASUS-Zenbook:~$ cd ~/work/arch-pc/lab07
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:18: error: invalid combination of opcode and operands
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ls
in_out.asm lab7-1 lab7-1.o lab7-2 lab7-2.asm lab7-2.lst
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.10: Трансляция

Запускаем листинг и изучаем его

```
abutalim@abutalim-ASUS-Zenbook: ~/work/arch-pc/lab07
/home/ab-7-2.lst [----] 0 L: [ 1+ 0 1/226] *(0 /13893b) 0032 0x020 [*][X]
1                               %include 'in_out.asm'
1                               <1> ;----- slen -----
2                               <1> ; Функция вычисления длины сообщения
3                               <1> slen:.....
4 00000000 53                   <1> push    ebx.....
5 00000001 89C3                 <1> mov     ebx, eax.....
6                               <1>.....
7                               <1> nextchar:.....
8 00000003 803800              <1> cmp     byte [eax], 0...
9 00000006 7403                <1> jz     finished.....
10 00000008 40                 <1> inc     eax.....
11 00000009 EBF8               <1> jmp     nextchar.....
12                               <1>.....
13                               <1> finished:
14 0000000B 29D8               <1> sub     eax, ebx
15 0000000D 5B                 <1> pop     ebx.....
16 0000000E C3                 <1> ret.....
17                               <1>
18                               <1>
19                               <1> ;----- sprint -----
20                               <1> ; Функция печати сообщения
21                               <1> ; входные данные: mov eax,<message>
```

Рисунок 2.11: Запускаем листинг

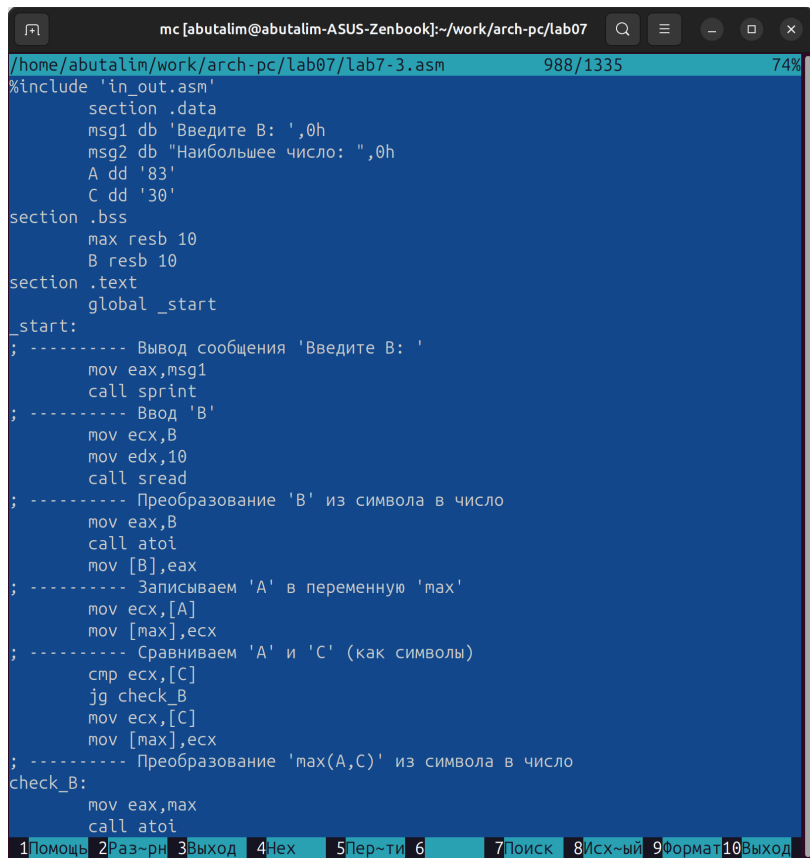
#Самостоятельная работа Вариант - 18

Создаем новый файл

```
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ touch lab7-3.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.12: Новый файл

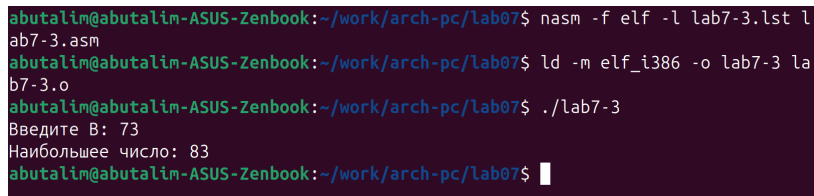
Напишем программу нахождения наименьшей из 3 целочисленных переменных `a`, `b` и `c`.



```
mc [abutalim@abutalim-ASUS-Zenbook]:~/work/arch-pc/lab07
/home/abutalim/work/arch-pc/lab07/lab7-3.asm 988/1335 74%
#include 'in_out.asm'
    section .data
        msg1 db 'Введите B: ',0h
        msg2 db "Наибольшее число: ",0h
        A dd '83'
        C dd '30'
    section .bss
        max resb 10
        B resb 10
    section .text
        global _start
    _start:
; ----- Вывод сообщения 'Введите B: '
        mov eax,msg1
        call sprint
; ----- Ввод 'B'
        mov ecx,B
        mov edx,10
        call sread
; ----- Преобразование 'B' из символа в число
        mov eax,B
        call atoi
        mov [B],eax
; ----- Записываем 'A' в переменную 'max'
        mov ecx,[A]
        mov [max],ecx
; ----- Сравниваем 'A' и 'C' (как символы)
        cmp ecx,[C]
        jg check_B
        mov ecx,[C]
        mov [max],ecx
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
        mov eax,max
        call atoi
```

Рисунок 2.13: Пишем программу

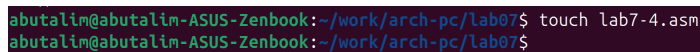
Запускаем программу и смотрим как она работает



```
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf -l lab7-3.lst lab7-3.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ./lab7-3
Введите B: 73
Наибольшее число: 83
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.14: Запуск программы

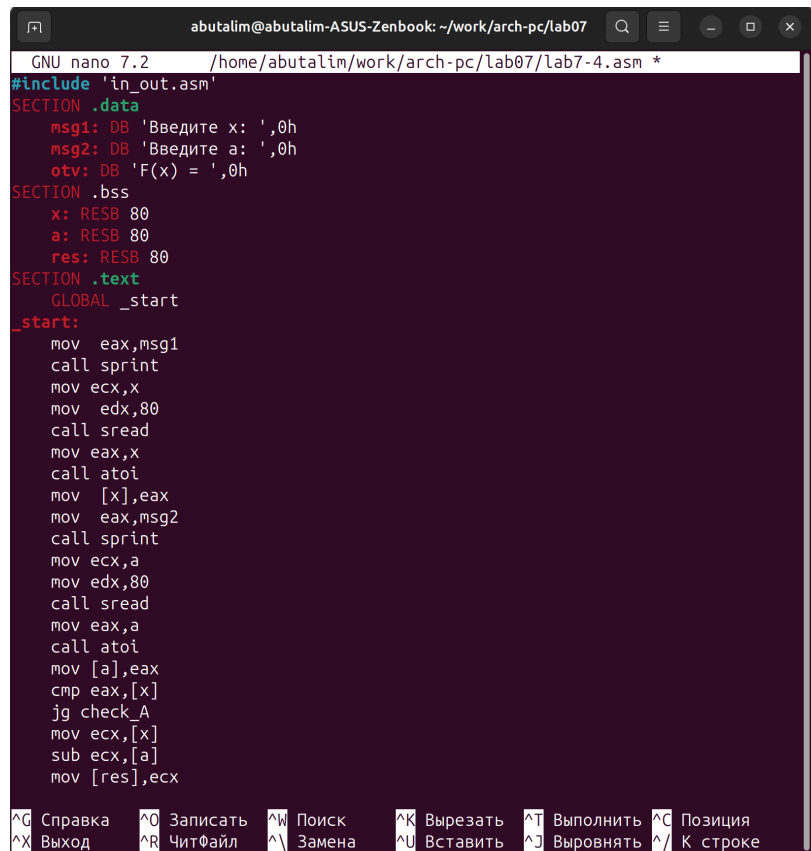
Создаем новый файл



```
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ touch lab7-4.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.15: Новый файл

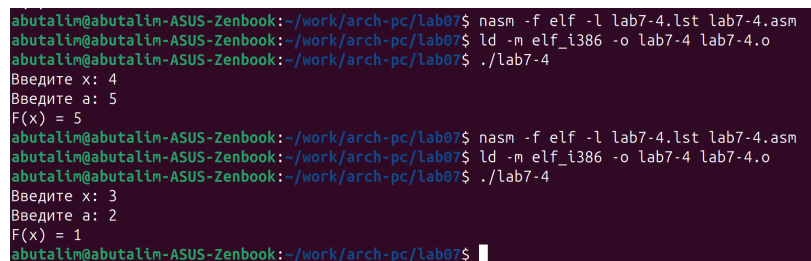
Напишем программу, которая для введенных с клавиатуры значений  $x$  и  $a$  вычисляет значение заданной функции  $F(x)$  и выводит результат вычислений.



```
GNU nano 7.2 /home/abutalim/work/arch-pc/lab07/lab7-4.asm *
#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
```

Рисунок 2.16: Пишем программу

Запускаем программу и смотрим как она работает



```
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf -l lab7-4.lst lab7-4.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 4
Введите a: 5
F(x) = 5
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ nasm -f elf -l lab7-4.lst lab7-4.asm
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 2
F(x) = 1
abutalim@abutalim-ASUS-Zenbook:~/work/arch-pc/lab07$
```

Рисунок 2.17: Запуск программы

#Вывод

Мы познакомились с структурой файла листинге и изучили условный и без-  
условный переход