

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**  
**Факультет физико-математических и естественных наук**  
**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №7**

*Дисциплина:* **Архитектура компьютеров**

Студент: Болдырева Дельгир

Группа: НКАбд-01-25

Москва

2025 г.

## **Оглавление**

1 Цель работы.....	3
2 Задание.....	4
3 Теоретическое введение.....	5
4 Выполнение лабораторной работы.....	6
4.1 Реализация переходов в NASM.....	6
4.2 Изучение структуры файла листинга.....	10
5 Задания для самостоятельной работы.....	12
6 Выводы.....	15
Список литературы.....	16

## ***1 Цель работы***

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

## ***2 Задания***

1. Реализация переходов в NASM
2. Изучение структуры файлов листинга
3. Самостоятельное написание программ по материалам лабораторной Работы.

### ***3 Теоретическое введение***

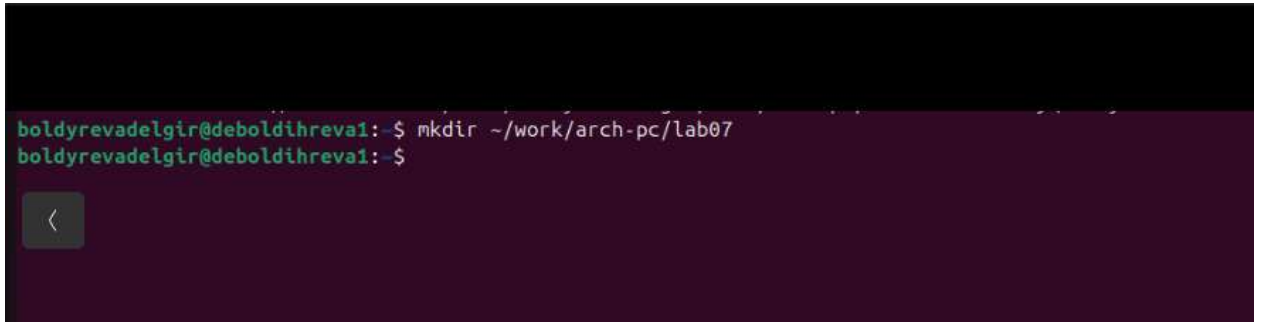
Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов:

- условный переход—выполнение или не выполнение перехода в определенную точку программы в зависимости от проверки условия.
- безусловный переход—выполнение передачи управления в определенную точку программы без каких-либо условий.

## 4 Выполнение лабораторной работы

### 4.1 Реализация переходов в NASM

Создадим каталог для программ лабораторной работы № 7, перейдем в него и создадим файл lab7-1.asm (рис.4.1.1-4.1.3)



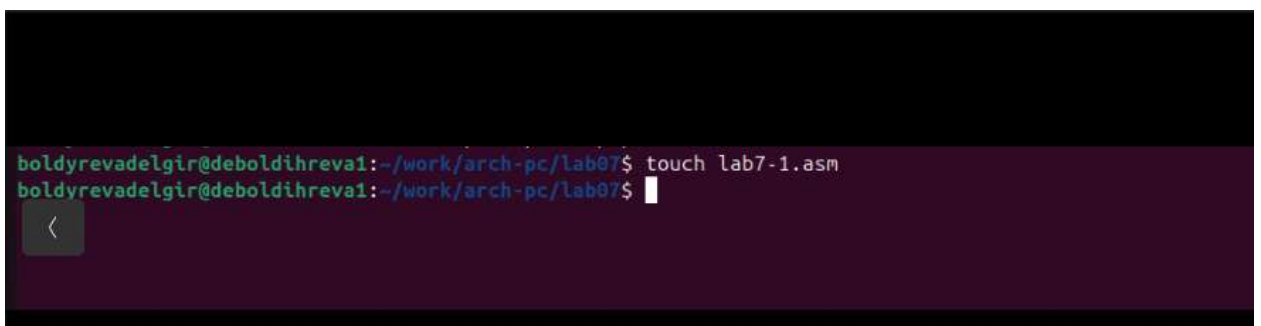
```
boldyrevadelgir@deboldihreva1:~$ mkdir ~/work/arch-pc/lab07
boldyrevadelgir@deboldihreva1:~$
```

Рис. 4.1.1



```
boldyrevadelgir@deboldihreva1:~$ cd ~/work/arch-pc/lab07
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

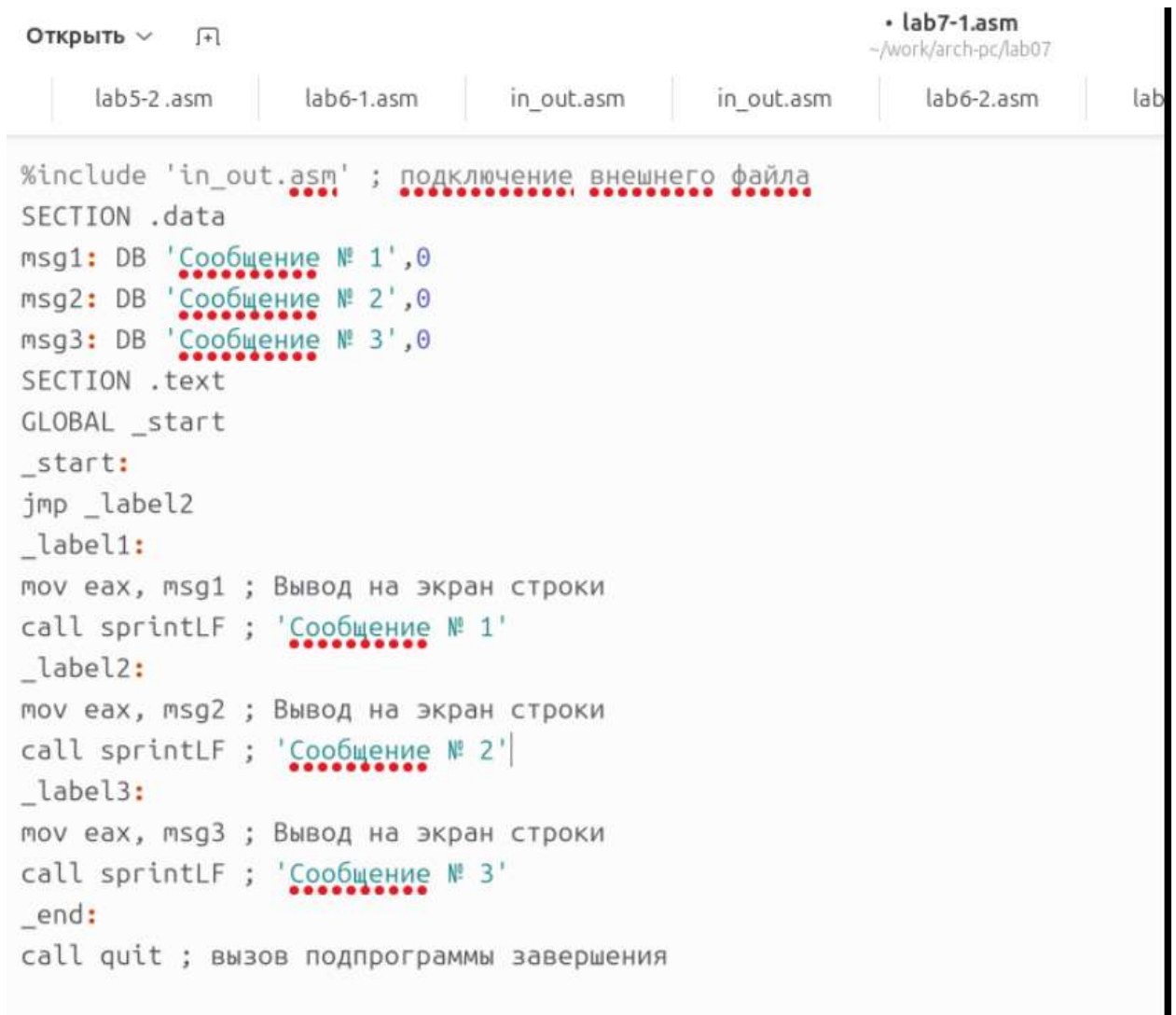
Рис. 4.1.2




```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ touch lab7-1.asm
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис. 4.1.3

Введем в файл lab7-1.asm текст программы из листинга (рис.4.1.4)



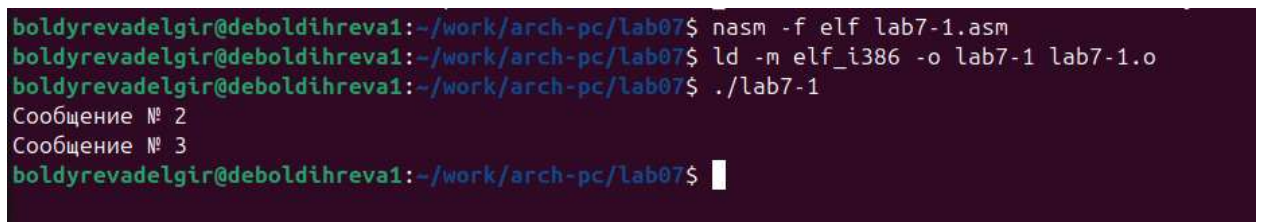
```
Открыть ▾ 
• lab7-1.asm
~/work/arch-pc/lab07

lab5-2.asm | lab6-1.asm | in_out.asm | in_out.asm | lab6-2.asm | lab7-1.asm

%include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.1.4

Создадим исполняемый файл и запустим его (рис. 4.1.5)



```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.4.1.5

Изменим текст программы в соответствии с листингом (рис.4.1.6)

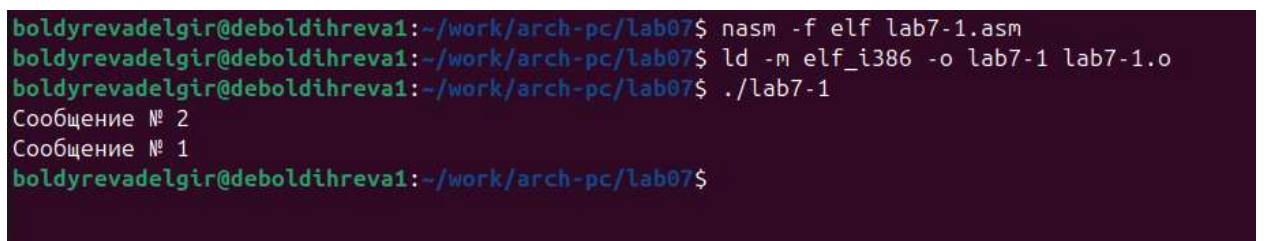


```
Открыть  [F4] lab7-1.asm ~/work/arch-pc/lab07
lab5-2.asm lab6-1.asm in_out.asm in_out.asm lab6-2.asm lab6-3.asm variant.

%include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис.4.1.6

Запустим файл и проверим его работу(рис.4.1.7)



```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.4.1.7

Создадим файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. Внимательно изучим текст программы из листинга 7.3 и введем в lab7-2.asm (рис 4.1.8-4.1.9)



```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ touch lab7-2.asm
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.4.1.8

```

Открыть  [F]
lab7-2.asm
~/work/arch-pc/lab07
lab6-1.asm  in_out.asm  in_out.asm  lab6-2.asm  lab6-3.asm  variant.asm  lab7-2.asm

%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 ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
; ----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; ecx = A

```

Рис.4.1.9

Создадим исполняемый и проверим его работу (рис. 4.1.10-4.1.12)

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 35
Наибольшее число: 50
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.4.1.10

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 90
Наибольшее число: 90
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.4.1.11

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 10
Наибольшее число: 50
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.4.1.12

## 4.2 Изучение структуры файла листинга

Создадим файл листинга для программы из файла lab7-2.asm (рис.4.2.1-4.2.2)

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ mcedit lab7-2.lst
```

Рис.4.2.1

```
1 %include 'in_out.asm'
2
3 ;----- slen -----
4 ; Функция вычисления длины сообщения
5 slen:
6     push    ebx
7     mov     ebx, eax
8     nextchar:
9     cmp     byte [eax], 0
10    jz       finished
11    inc     eax
12    jmp     nextchar
13
14 finished:
15     sub     eax, ebx
16     pop     ebx
17     ret
18
19 ;----- sprint -----
20 ; Функция печати сообщения
21 ; входные данные: mov eax, <message>
22 sprint:
23     push    edx
24     push    ecx
25     push    ebx
26     push    eax
27     call    slen
28
29     mov     edx, eax
```

Рис.4.2.2

Откроем файл и удалим один операнд (рис.4.2.3)

```

; ----- Сравниваем 'A' и 'C' (как символы)
mov ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в 'max'
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
mov ebx,[B] ; Сравниваем 'max(A,C)' и 'B'
jg fin ; если 'max(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [max],ecx
; ----- Вывод результата
fin:
mov eax, |
call sprintf ; Вывод сообщения 'Наибольшее число: '
mov eax,[max]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход

```

Рис.4.2.3

Новый файл при запуске выдает ошибку (рис.4.2.4)

```

boldyrevadlgir@deboldihreva1:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
lab7-2.asm:45: error: invalid combination of opcode and operands
boldyrevadlgir@deboldihreva1:~/work/arch-pc/lab07$

```

Рис.4.2.4

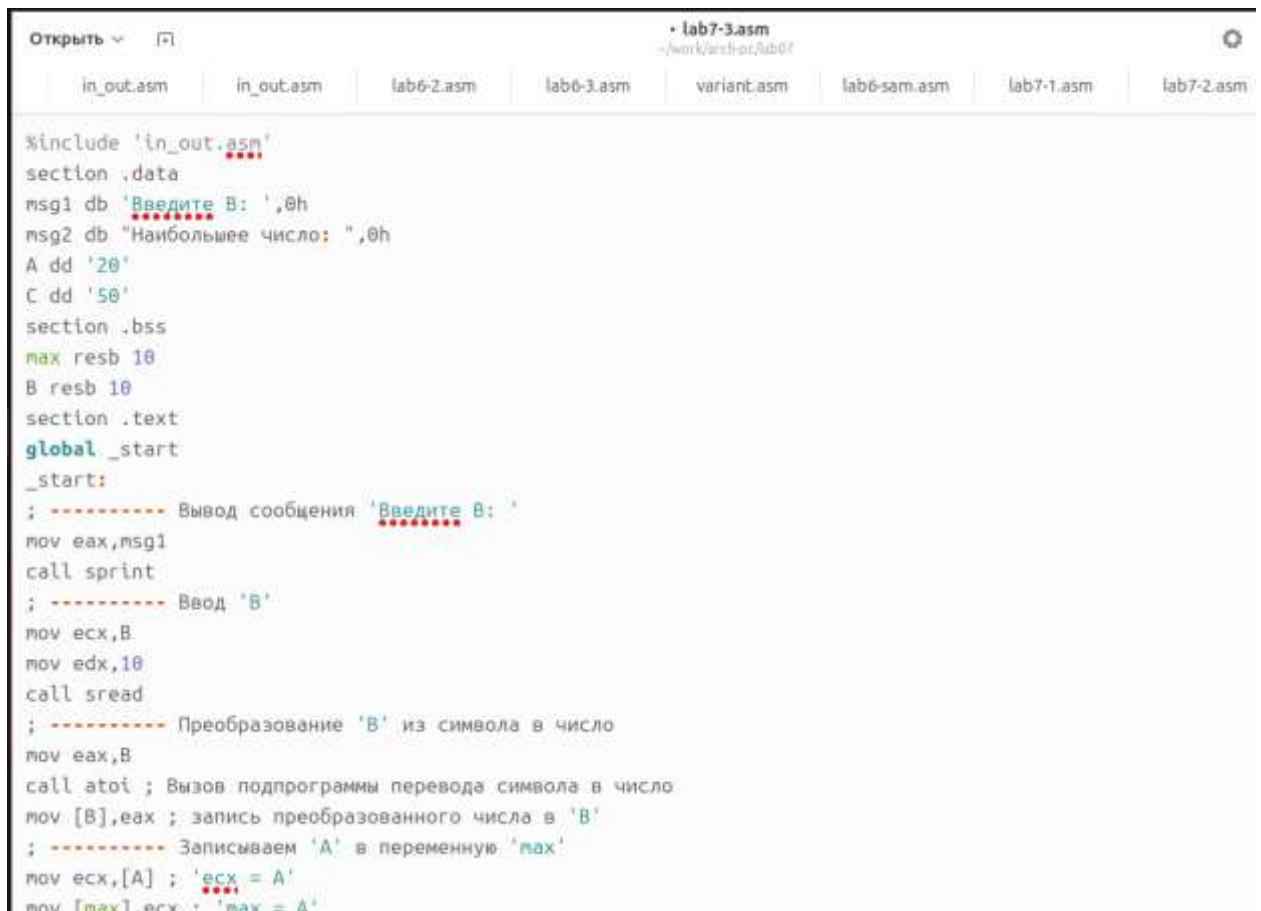
## 5 Задания для самостоятельной работы

Напишем программу нахождения наименьшей из 3 целочисленных

переменных  $a$ ,  $b$  и  $c$ . Значения переменных 94,5,58. Создадим файл lab7-3.asm из листинга (рис.5.1-5.5)

```
boldyrevadelgir@deboldihreval:~/work/arch-pc/lab07$ touch lab7-3.asm
boldyrevadelgir@deboldihreval:~/work/arch-pc/lab07$
```

Рис.5.1



```
Открыть ▾ [icon] • lab7-3.asm
~/work/arch-pc/lab07
in_out.asm in_out.asm lab6-2.asm lab6-3.asm variant.asm lab6-sam.asm lab7-1.asm lab7-2.asm

#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 ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
; ----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
```

Рис.5.2

```
boldyrevadelgir@deboldihreval:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
boldyrevadelgir@deboldihreval:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
boldyrevadelgir@deboldihreval:~/work/arch-pc/lab07$ ./lab7-3
Введите B: 94
Наибольшее число: 94
boldyrevadelgir@deboldihreval:~/work/arch-pc/lab07$
```

Рис.5.3

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-3
Введите В: 5
Наибольшее число: 50
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.5.4

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-3
Введите В: 58
Наибольшее число: 58
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$
```

Рис.5.5

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

Функция под номером 3. Создадим исполняемый файл и проверим его работу для значений  $x$  и  $a$  (Рис.5.6-5.9)

```

Открыть ▾ [F1]
• lab7-4.asm
~/work/arch-pc/lab07
in_out.asm lab6-2.asm lab6-3.asm variant.asm lab6-sam.asm lab7-1.asm lab7-2.asm

SECTION .data
msg_x: DB "Введите значение x: ", 0
msg_a: DB "Введите значение a: ", 0
res: DB "Результат: ", 0

SECTION .bss
x: RESB 80
a: RESB 80

SECTION .text
GLOBAL _start

_start:
    ; Ввод x
    mov esi, msg_x
    call sprint

    mov ecx, x
    mov edx, 80
    call sread

    mov esi, x
    call atoi
    mov edi, eax    ; сохраняем x в edi

    ; Ввод a

```

Рис.5.6

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-4
Введите значение x: █
```

Рис.5.7

```
Введите значение x: 3
Введите значение a: 4
Результат: 134520904
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ █
```

Рис.5.8

```
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ ./lab7-4
Введите значение x: 1
Введите значение a: 4
Результат: 134520904
boldyrevadelgir@deboldihreva1:~/work/arch-pc/lab07$ █
```

Рис.5.9

## ***6 Выводы***

Мы изучили команды условного и безусловного перехода, приобрели навыки написания программ с использованием переходов, ознакомились с назначением и структурой файла листинга

## Список литературы

1. GDB:TheGNUProjectDebugger.—URL:<https://www.gnu.org/software/gdb/>.
2. GNUBashManual.—2016.—URL:<https://www.gnu.org/software/bash/manual/>.
3. Midnight CommanderDevelopment Center.—2021.—URL: <https://midnightcommander.org/>.
4. NASMASsemblyLanguageTutorials.—2021.—URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. —(In a Nutshell). —ISBN 0596009658.—URL: [http://www.amazon.com/Learning bash-Shell-Programming-Nutshell/dp/0596009658](http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658).
6. RobbinsA. Bash Pocket Reference.—O'Reilly Media,2016.—156 с.—ISBN 978-1491941591.
7. TheNASMdocumentation.—2021.—URL:<https://www.nasm.us/docs.php>.