

Лабораторная работа №7

Дисциплина: архитектура компьютера

Выслоух Алиса Александровна

Содержание

| | | |
|----------|---|-----------|
| 1 | Цель работы | 5 |
| 2 | Задание | 6 |
| 3 | Выполнение лабораторной работы | 7 |
| 3.1 | Реализация переходов в NASM. | 7 |
| 3.2 | Изучение структуры файлы листинга. | 9 |
| 3.3 | Задание для самостоятельной работы. | 10 |
| 4 | Выводы | 15 |

Список иллюстраций

| | | |
|------|-----------------------------------|----|
| 3.1 | Создание папки. | 7 |
| 3.2 | Ввод программы. | 7 |
| 3.3 | Создание и запуск файла. | 8 |
| 3.4 | Ввод текста из листинга. | 8 |
| 3.5 | Создание и запуск файла. | 8 |
| 3.6 | Ввод текста из листинга. | 9 |
| 3.7 | Создание файла. | 9 |
| 3.8 | Получение файла листинга. | 9 |
| 3.9 | Первая выбранная строка. | 10 |
| 3.10 | Вторая выбранная строка. | 10 |
| 3.11 | Третья выбранная строка. | 10 |
| 3.12 | Ввод программы. | 11 |
| 3.13 | Создание и запуск файл. | 11 |
| 3.14 | написание программы. | 13 |
| 3.15 | Создание и запуск файла. | 14 |

Список таблиц

1 Цель работы

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

2 Задание

1. Реализация переходов в NASM.
2. Изучение структуры файлы листинга.
3. Задание для самостоятельной работы.

3 Выполнение лабораторной работы

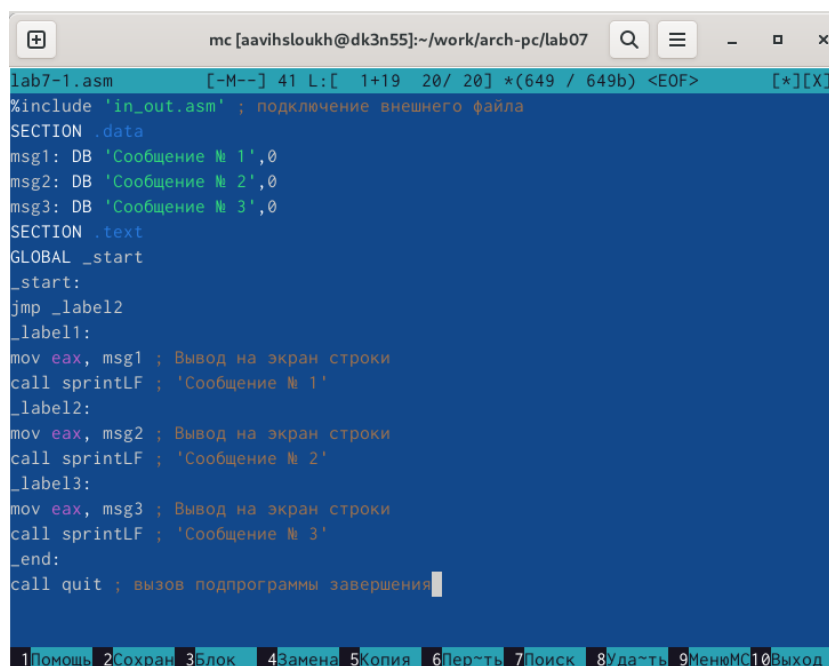
3.1 Реализация переходов в NASM.

Создаю папку lab7, перехожу в нее и создаю файл для работы (рис. 3.1).

```
aavihsloukh@dk3n55 ~ $ mkdir ~/work/arch-pc/lab07
aavihsloukh@dk3n55 ~ $ cd ~/work/arch-pc/lab07
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ touch lab7-1.asm
```

Рис. 3.1: Создание папки.

Ввожу программу из листинга 7.1 в созданный файл (рис. 3.2).



```
lab7-1.asm [-M--] 41 L:[ 1+19 20/ 20] *(649 / 649b) <EOF> [*][X]
%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
```

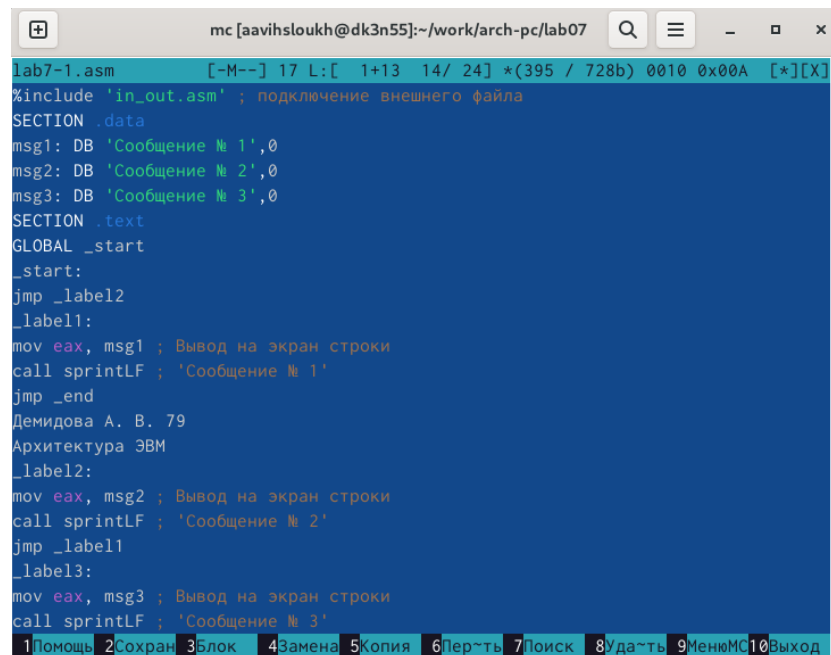
Рис. 3.2: Ввод программы.

Создаю и исполняю файл и запускаю его(рис. 3.3).

```
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-1
Сообщение № 2
Сообщение № 3
```

Рис. 3.3: Создание и запуск файла.

Ввожу измененный текст программы из листинга 7.2 (рис. 3.4).



```
lab7-1.asm [-M--] 17 L:[ 1+13 14/ 24] *(395 / 728b) 0010 0x00A [*][X]
#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'
jmp _end
Демидова А. В. 79
Архитектура ЭВМ
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
```

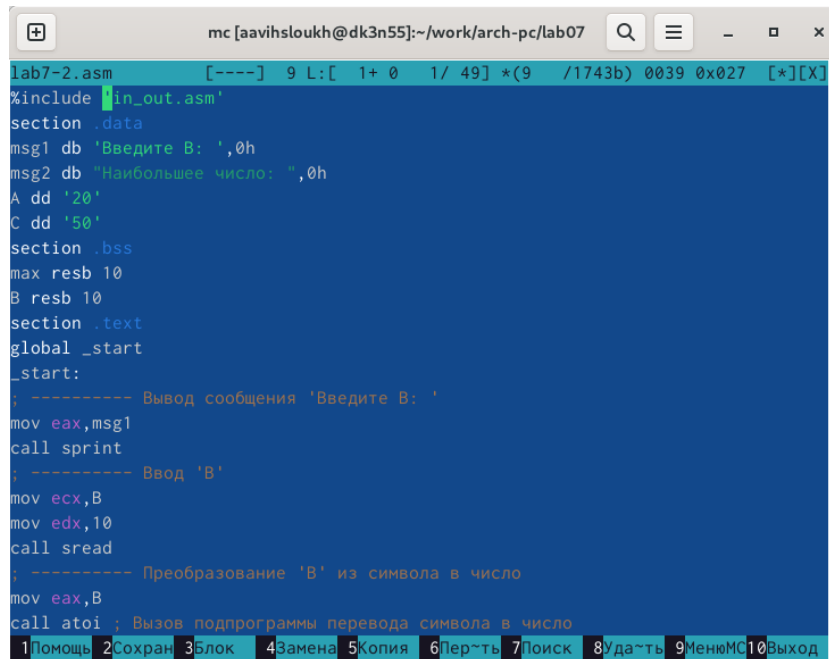
Рис. 3.4: Ввод текста из листинга.

Создаю и исполняю файл и запускаю его(рис. 3.5).

```
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-1
Сообщение № 2
Сообщение № 1
```

Рис. 3.5: Создание и запуск файла.

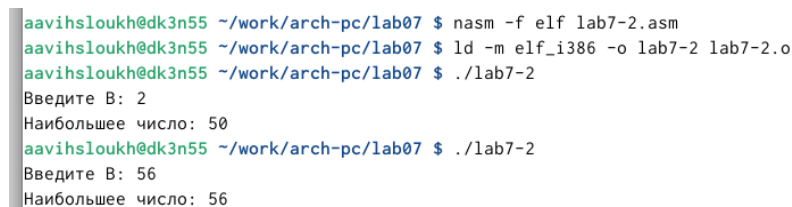
Создаю новый файл для работы 7-2 и вставляю в него текст из листинга 3 (рис. 3.6).



```
lab7-2.asm [----] 9 L: [ 1+ 0 1/ 49] *(9 /1743b) 0039 0x027 [*][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:
; ----- 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
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход
```

Рис. 3.6: Ввод текста из листинга.

Создаю, исполняю и запускаю файл, вводя разные значения B. Я вводила значения 2 и 56 (рис. 3.7).

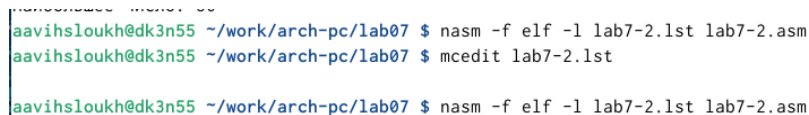


```
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-2.asm
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-2 lab7-2.o
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-2
Введите B: 2
Наибольшее число: 50
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-2
Введите B: 56
Наибольшее число: 56
```

Рис. 3.7: Создание файла.

3.2 Изучение структуры файлы листинга.

Получаю файл листинга, указав ключ -l и задав имя файла листинга в командной строке. (рис. 3.8).



```
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf -l lab7-2.lst lab7-2.asm
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ mcedit lab7-2.lst
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf -l lab7-2.lst lab7-2.asm
```

Рис. 3.8: Получение файла листинга.

Проанализировав файл, я поняла как он работает и какие значения выводит. Первая строка, которую я выбрала находится на 21 месте, ее адрес “00000101”, Машинный код - B8 [0A000000], а `mov eax,B` - исходный текст программы, означающий что в регистр `eax` мы вносим значения переменной `B`. (рис. 3.9).

A screenshot of a debugger window showing assembly code. The address is 21, the hex code is 00000101, the instruction is B8[0A000000], and the assembly comment is mov eax,B.

Рис. 3.9: Первая выбранная строка.

Вторая строка находится на 38 месте, ее адрес “00000106”, Машинный код - E891FFFFFF, а `call atoi` - исходный текст программы, означающий что символ лежащий в строке выше переводится в число. (рис. 3.10).


A screenshot of a debugger window showing assembly code. The address is 22, the hex code is 00000106, the instruction is E891FFFFFF, and the assembly comment is call atoi ; Вызов подпрограммы перевода символа в число.

Рис. 3.10: Вторая выбранная строка.

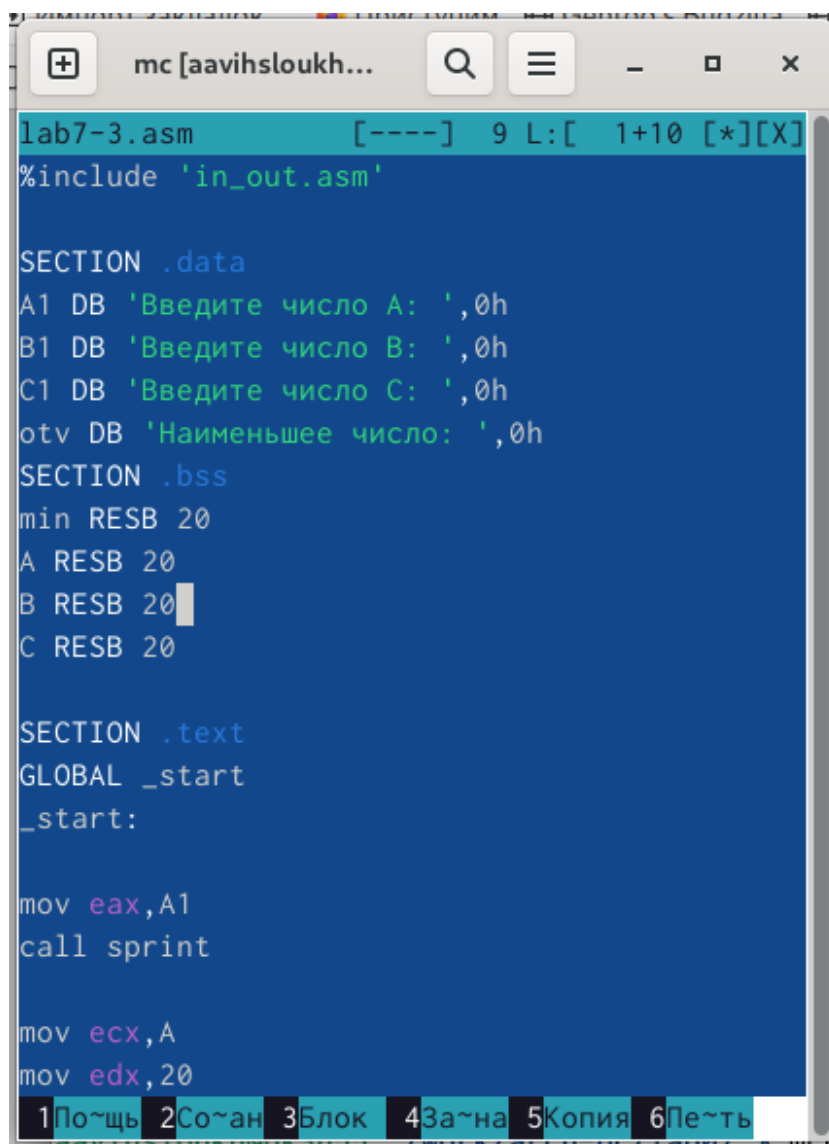
Третья строка находится на 50 месте, ее адрес “00000162”, Машинный код - A1[00000000], а `mov eax,[max]` - исходный текст программы, означающий что число хранившееся в переменной `max` записывается в регистр `eax`. (рис. 3.11).

A screenshot of a debugger window showing assembly code. The address is 47, the hex code is 00000163, the instruction is A1[00000000], and the assembly comment is mov eax,[max].

Рис. 3.11: Третья выбранная строка.

3.3 Задание для самостоятельной работы.

Создаю файл 7-3 и пишу программу для нахождения меньшего из введенных чисел. (рис. 3.12).



```
lab7-3.asm [----] 9 L:[ 1+10 [*][X]
%include 'in_out.asm'

SECTION .data
A1 DB 'Введите число A: ',0h
B1 DB 'Введите число B: ',0h
C1 DB 'Введите число C: ',0h
otv DB 'Наименьшее число: ',0h
SECTION .bss
min RESB 20
A RESB 20
B RESB 20
C RESB 20

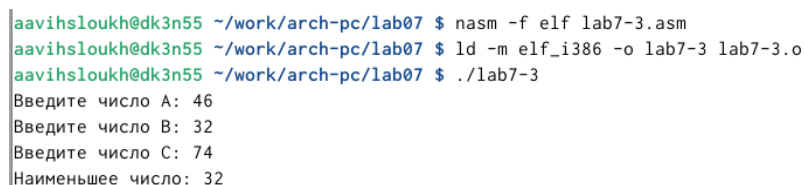
SECTION .text
GLOBAL _start
_start:

mov eax,A1
call sprint

mov ecx,A
mov edx,20
```

Рис. 3.12: Ввод программы.

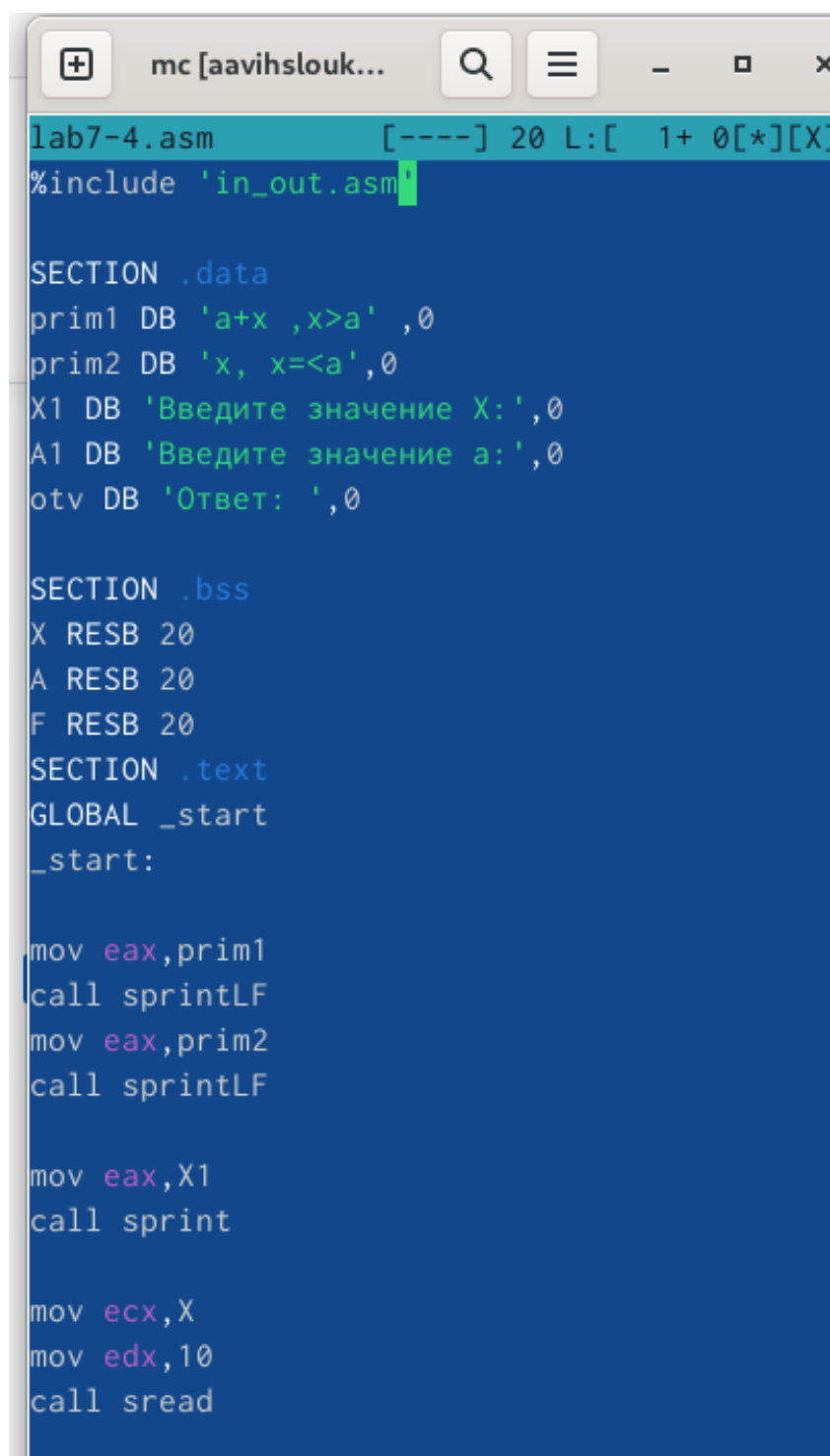
Создаю, исполняю и запуска файл. Ввожу числа из моего варианта. Мой вариант 19. Программа написано верно. (рис. 3.13).



```
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-3.asm
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-3 lab7-3.o
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-3
Введите число A: 46
Введите число B: 32
Введите число C: 74
Наименьшее число: 32
```

Рис. 3.13: Создание и запуск файл.

Я написала программу, чтобы она вычисляла выражение при введенных X и A . Для большего удобства, выражение которое будет вычисляться я вывожу в начале работы программы. (рис. 3.14).



```
lab7-4.asm [----] 20 L:[ 1+ 0[*]][X]
#include 'in_out.asm'

SECTION .data
prim1 DB 'a+x ,x>a' ,0
prim2 DB 'x, x=<a',0
X1 DB 'Введите значение X:',0
A1 DB 'Введите значение a:',0
otv DB 'Ответ: ',0

SECTION .bss
X RESB 20
A RESB 20
F RESB 20
SECTION .text
GLOBAL _start
_start:

mov eax,prim1
call sprintf
mov eax,prim2
call sprintf

mov eax,X1
call sprintf

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

Рис. 3.14: написание программы.

Создаю, исполняю и запускаю созданный файл, проверяю работу на числах 1 и 2. (рис. 3.15).

```
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-4.asm
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-4 lab7-4.o
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-4
a+x ,x>a
x, x=<a
Введите значение X:1
Введите значение a:2
Ответ: 3
aavihsloukh@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-4
a+x ,x>a
x, x=<a
Введите значение X:2
Введите значение a:1
Ответ: 8
```

Рис. 3.15: Создание и запуск файла.

4 Выводы

Я изучила команды условного и безусловного перехода. Приобрела навыки написания программ с переходами.