

Отчёт по лабораторной работе №8

дисциплина: Архитектура компьютера

Максим Александрович Мишонков

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
5	Выводы	19

Список иллюстраций

4.1	Создание каталога и файла	7
4.2	Введение текста программы	8
4.3	Проверка работы исполняемого файла	8
4.4	Изменение текста программы	9
4.5	Проверка работы исполняемого файла	9
4.6	Изменение текста программы	10
4.7	Проверка работы исполняемого файла	10
4.8	Создание файла	10
4.9	Введение текста программы	11
4.10	Введение текста программы	12
4.11	Проверка работы исполняемого файла	12
4.12	Создание файла листинга и открытие его в редакторе	12
4.13	Открытие файла в редакторе	13
4.14	Объяснение первой строки	13
4.15	Объяснение второй строки	13
4.16	Объяснение третьей строки	13
4.17	Удаление операнда в файле	14
4.18	Выполнение трансляции с получением файла листинга	14
4.19	Ошибка в тексте программы	14
4.20	Текст программы	15
4.21	Текст программы	16
4.22	Проверка работы исполняемого файла	16
4.23	Текст программы	17
4.24	Текст программы	18
4.25	Проверка работы исполняемого файла	18

1 Цель работы

Целью данной лабораторной работы является изучение команд условного и безусловного переходов, приобретение навыков написания программ с использованием переходов, знакомство с назначением и структурой файла листинга.

2 Задание

Изучить команды условного и безусловного переходов, приобрести навыки написания программ с использованием переходов, познакомиться с назначением и структурой файла листинга.

3 Теоретическое введение

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

Условный переход – это выполнение перехода в определенную точку программы в зависимости от проверки условия.

Безусловный переход – это выполнение передачи управления в определенную точку программы без каких-либо условий.

4 Выполнение лабораторной работы

1. Создал каталог для программ лабораторной работы №8, перешёл в него и создал файл lab8-1.asm. (рис. 4.1)

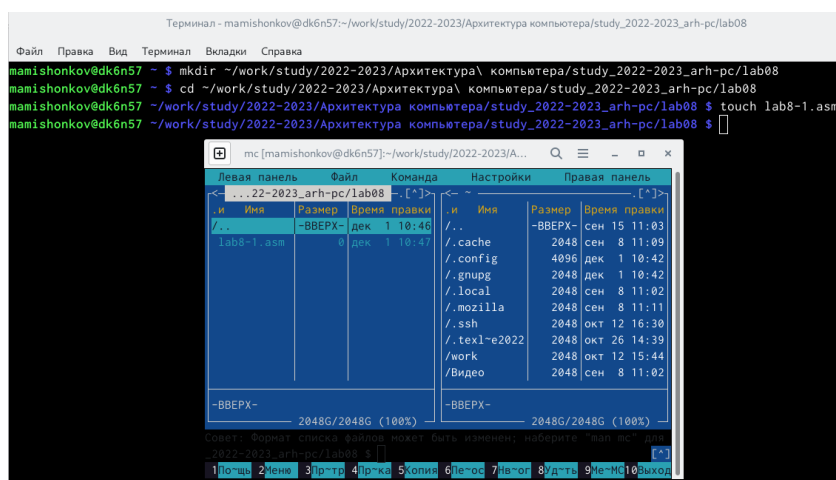
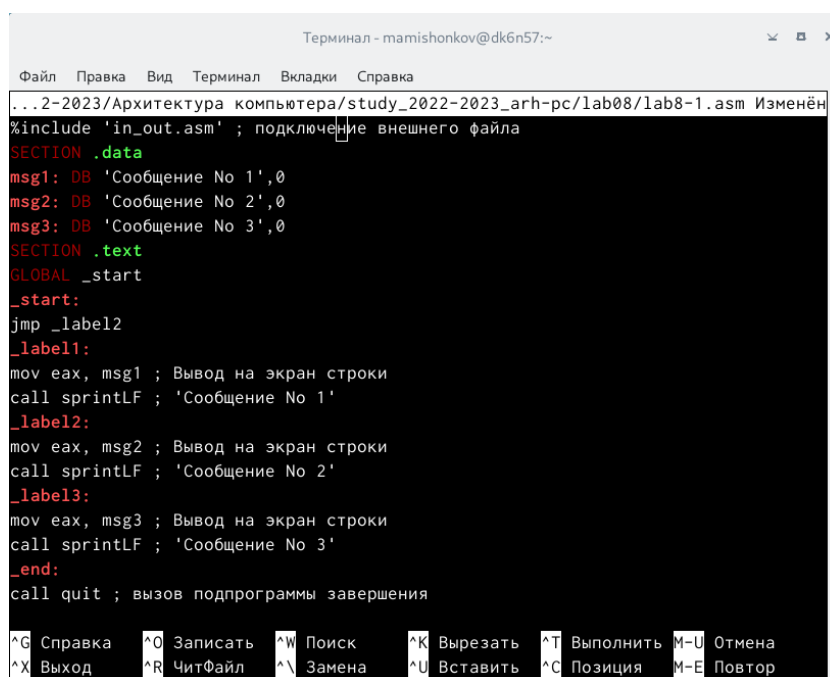


Рис. 4.1: Создание каталога и файла

2. Рассмотрел пример программы с использованием инструкции `jump`, ввёл в файл lab8-1.asm текст программы из листинга. (рис. 4.2)

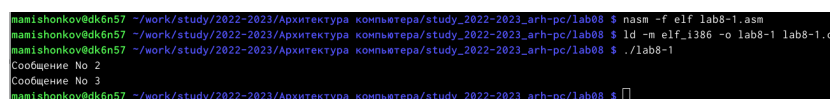


```
Терминал - mamishonkov@dk6n57:~
Файл  Правка  Вид  Терминал  Вкладки  Справка
...2-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08/lab8-1.asm Изменён
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение No 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение No 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение No 3'
_end:
call quit ; вызов подпрограммы завершения

^C Справка  ^O Записать  ^W Поиск    ^K Вырезать  ^T Выполнить  M-U Отмена
^X Выход    ^R ЧитФайл  ^\ Замена  ^U Вставить  ^C Поция     M-E Повтор
```

Рис. 4.2: Введение текста программы

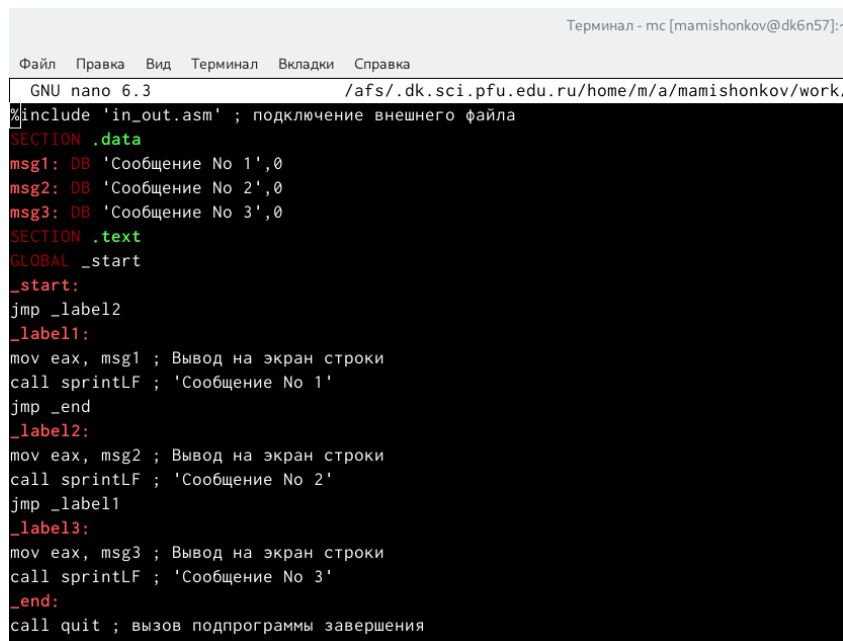
3. Создал исполняемый файл и проверил его работу. (рис. 4.3)



```
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf lab8-1.asm
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 3
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $
```

Рис. 4.3: Проверка работы исполняемого файла

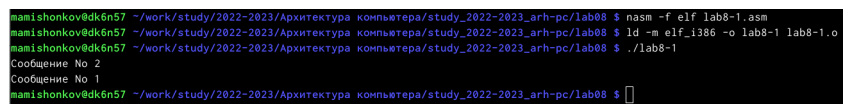
4. Изменил текст программы таким образом, чтобы программа выводила сначала 'Сообщение №2', потом 'Сообщение №1'. Для этого в текст программы после вывода сообщения №2 добавил инструкцию jmp с меткой _label1, а после вывода сообщения №1 добавил инструкцию jmp с меткой _end. (рис. 4.4)



```
Терминал - mc [mamishonkov@dk6n57]:-
Файл  Правка  Вид  Терминал  Вкладки  Справка
GNU nano 6.3 /afs/.dk.sci.pfu.edu.ru/home/m/a/mamishonkov/work.
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение No 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение No 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение No 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 4.4: Изменение текста программы

5. Создал исполняемый файл и проверил его работу. (рис. 4.5)



```
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf lab8-1.asm
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 1
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $
```

Рис. 4.5: Проверка работы исполняемого файла

6. Изменил текст программы, изменив инструкции `jmp`, чтобы вывод программы был следующим: (рис. 4.6, 4.7)

```

Терминал - mc [mamishonkov@dk6n57]:
Файл  Правка  Вид  Терминал  Вкладки  Справка
GNU nano 6.3 /afs/.dk.sci.pfu.edu.ru/home/m/a/mamishonkov/work
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение No 1',0
msg2: DB 'Сообщение No 2',0
msg3: DB 'Сообщение No 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение No 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение No 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение No 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения

```

Рис. 4.6: Изменение текста программы

```

mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf lab8-1.asm
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $

```

Рис. 4.7: Проверка работы исполняемого файла

7. Создал в каталоге lab08 файл lab8-2.asm, ввёл текст программы из листинга, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: А,В и С. (рис. 4.8, 4.9, 4.10)

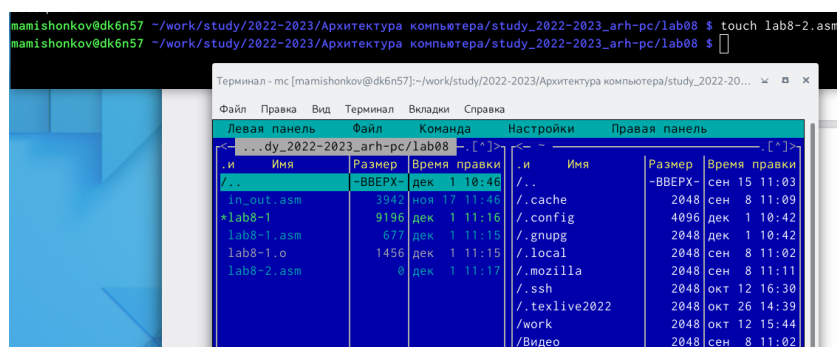


Рис. 4.8: Создание файла

```
Терминал - mc [mamishonkov@dk6n57]:  
Файл Правка Вид Терминал Вкладки Справка  
GNU nano 6.3 /afs/.dk.sci.pfu.edu.ru/home/m/a/mamishonkov/work  
%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'  
; ----- Сравниваем 'A' и 'C' (как символы)  
cmp 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
```

Рис. 4.9: Введение текста программы

```

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]
cmp ecx,[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,msg2
call sprintf ; Вывод сообщения 'Наибольшее число: '
mov eax,[max]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход

```

Рис. 4.10: Введение текста программы

8. Создал исполняемый файл и проверил его работу. (рис. 4.11)

```

mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ touch lab8-2.asm
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf lab8-2.asm
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-2
Введите B: 1
Наибольшее число: 50
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-2
Введите B: 52
Наибольшее число: 52
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $

```

Рис. 4.11: Проверка работы исполняемого файла

9. Создал файл листинга для программы из файла lab8-2.asm, указав ключ -l и задав имя файла листинга в командной строке, открыл этот файл при помощи редактора mcedit. (рис. 4.12, 4.13)

```

mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf -l lab8-2.lst lab8-2.asm
mamishonkov@dk6n57 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ mcedit lab8-2.lst

```

Рис. 4.12: Создание файла листинга и открытие его в редакторе

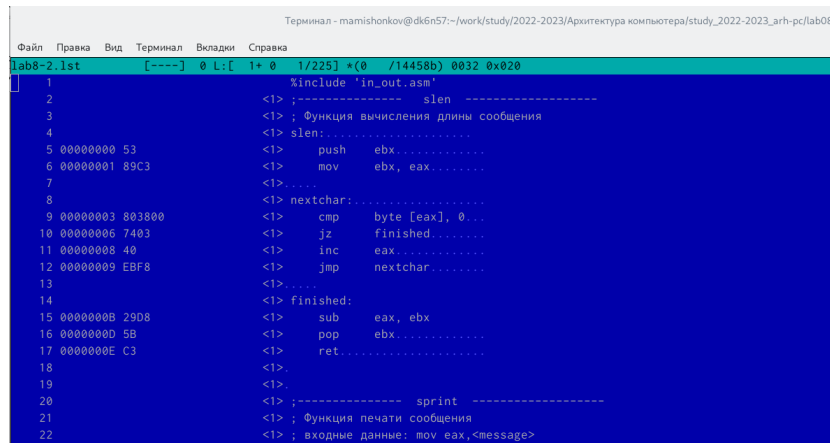


Рис. 4.13: Открытие файла в редакторе

10. Данная строка находится на 21 месте, её адрес - 00000101, машинный код - B8[0A000000], а mov eax,B - исходный текст программы, который означает, что мы в регистр eax вносим значение переменной B. (рис. 4.14)



Рис. 4.14: Объяснение первой строки

11. Данная строка находится на 35 месте, её адрес - 00000130, машинный код - E867FFFFFF, а call atoi - исходный текст программы, который означает, что символ, лежащий в строке выше, переводится в число. (рис. 4.15)

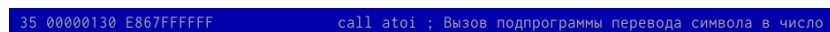


Рис. 4.15: Объяснение второй строки

12. Данная строка находится на 38 месте, её адрес - 0000013A, машинный код - 8B0D[00000000], а mov ecx,[max] - исходный текст программы, означающий, что число, хранившееся в переменной max, записывается в регистр ecx. (рис. 4.16)



Рис. 4.16: Объяснение третьей строки

13. Удалил один из операндов в файле с программой lab8-2.asm. (рис. 4.17)

```
check_B:
mov eax,
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]
cmp ecx,[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, msg2
call sprintf ; Вывод сообщения 'Наибольшее число: '
mov eax,[max]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход
```

Рис. 4.17: Удаление операнда в файле

14. Выполнил трансляцию с получением файла листинга, программа выдала ошибку, как и должно быть, потому что один из операндов был удалён. В файле листинга изображается, где именно ошибка и с чем она связана. (рис. 4.18, 4.19)

```
namishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:34: error: invalid combination of opcode and operands
namishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $
```

Рис. 4.18: Выполнение трансляции с получением файла листинга

```
34                                     mov eax
34  ***** error: invalid combination of opcode and operands
```

Рис. 4.19: Ошибка в тексте программы

Самостоятельная работа

1. Написал программу нахождения наименьшей из 3 целочисленных переменных, соответствующих варианту 14. (рис. 4.20, 4.21)

```

GNU nano 6.3 /afs/.dk.sci.pfu.edu.ru/home/m/a/mamishonkov
%include 'in_out.asm'
section .data
msg1 db 'Введите B: ',0h
msg2 db "Наименьшее число: ",0h
A dd '81'
C dd '72'

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]
jb check_B
mov ecx,[C]
mov [min],ecx

```

Рис. 4.20: Текст программы

```

check_B:
mov eax,min
call atoi
mov [min],eax

mov ecx,[min]
cmp ecx,[B]
jb fin
mov ecx,[B]
mov [min],ecx

fin:
mov eax, msg2
call sprint
mov eax,[min]
call iprintLF
call quit

```

Рис. 4.21: Текст программы

2. Создал исполняемый файл и проверил его работу. (рис. 4.22)

```

mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf lab8-3.asm
mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-3
Введите B: 22
Наименьшее число: 22
mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $

```

Рис. 4.22: Проверка работы исполняемого файла

3. Написал программу, которая для введенных с клавиатуры значений x и a вычисляет значение функции, соответствующей варианту 14, и выводит результат вычислений. (рис. 4.23, 4.24)


```
GNU nano 6.3 /afs/.dk.sci.pfu.edu.ru/home/m/a/mamishonkov.  
%include 'in_out.asm'  
  
SECTION .data  
msg1 db 'Введите X: ',0h  
msg2 db 'Введите a: ',0h  
msg3 db 'Ответ: ',0h  
  
SECTION .bss  
x resb 10  
a resb 10  
o resb 10  
  
SECTION .text  
global _start  
_start:  
  
mov eax,msg1  
call sprint  
  
mov ecx,x  
mov edx,10  
call sread  
  
mov eax,x  
call atoi  
mov [x],eax  
  
mov eax,msg2  
call sprint  
  
mov ecx,a  
mov edx,10  
call sread
```

Рис. 4.23: Текст программы

```

mov eax,a
call atoi
mov [a],eax

mov ecx,[x]
mov [o],ecx

mov ebx,3
cmp ecx,[a]
jle fin
mov eax,[x]
mul ebx
add eax,1
mov [o],eax
jmp otv

fin:
mov eax,[a]
mul ebx
add eax,1
mov [o],eax

otv:
mov eax,msg3
call sprint
mov eax,[o]
call iprintLF
call quit

```

Рис. 4.24: Текст программы

4. Создал исполняемый файл и проверил его работу. (рис. 4.25)

```

mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ nasm -f elf lab8-4.asm
mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-4
Введите X: 2
Ответ: 3
mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $ ./lab8-4
Введите X: 4
Введите a: 2
Ответ: 13
mamishonkov@dk4n65 ~/work/study/2022-2023/Архитектура компьютера/study_2022-2023_arh-pc/lab08 $

```

Рис. 4.25: Проверка работы исполняемого файла

5 Выводы

В ходе выполнения данной лабораторной работы я изучил команды условного и безусловного переходов, приобрёл навыки написания программ с использованием переходов, познакомился с назначением и структурой файла листинга.