

Отчёт по лабораторной работе №7

Дисциплина: архитектура компьютера

Репкина Елизавета Андреевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	16
	Список литературы	17

Список иллюстраций

4.1	Выполнение команд	8
4.2	Редактирование файла	8
4.3	Запуск файла	9
4.4	Редактирование файла	9
4.5	Запуск файла	9
4.6	Редактирование файла	10
4.7	Запуск файла	10
4.8	Создание файла	10
4.9	Редактирование файла	11
4.10	Запуск исполняемого файла	11
4.11	Создание файла	11
4.12	Открытие файла	12
4.13	Изменение файла	13
4.14	Редактирование файла	14
4.15	Запуск исполняемого файла	14
4.16	Редактирование файла	15
4.17	Запуск исполняемого файла	15

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. Выполнение заданий для самостоятельной 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

Команды безусловного перехода

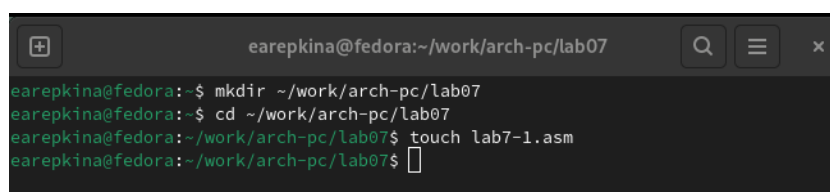
Безусловный переход выполняется инструкцией `jmp` (от англ. `jump` – прыжок), которая включает в себя адрес перехода, куда следует передать управление: `jmp Адрес перехода` Адрес перехода может быть либо меткой, либо адресом области памяти, в которую предварительно помещен указатель перехода. Кроме того, в качестве операнда можно использовать имя регистра, в таком случае переход будет осуществляться по адресу, хранящемуся в этом регистре.

Команды условного перехода

Как отмечалось выше, для условного перехода необходима проверка какого-либо условия. В ассемблере команды условного перехода вычисляют условие перехода анализируя флаги из регистра флагов

4 Выполнение лабораторной работы

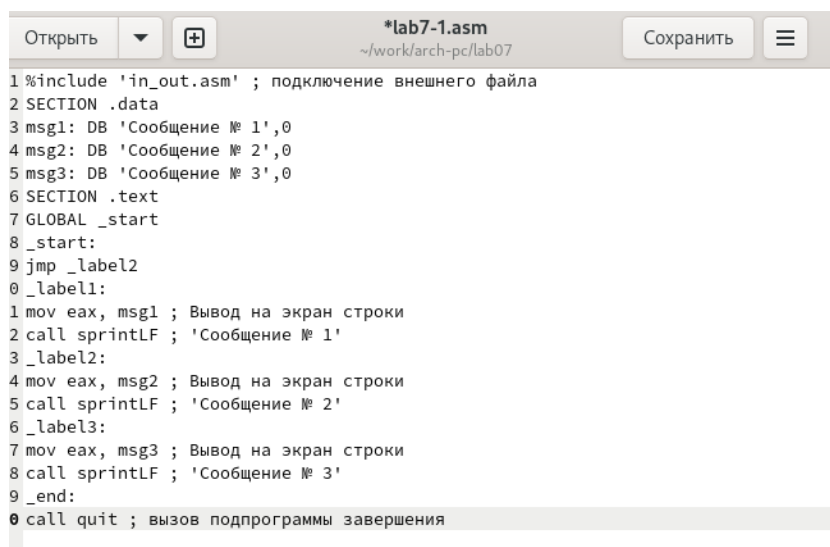
1. Реализация переходов в NASM Создаю каталог для программ лабораторной работы № 7, перехожу в него и создаю файл lab7-1.asm (рис. 4.1)



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

Рис. 4.1: Выполнение команд

Ввожу в файл lab7-1.asm текст программы из листинга 7.1. (рис. 4.2)



```
Открыть  *lab7-1.asm  Сохранить
~/work/arch-pc/lab07

1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label2
0 _label1:
1 mov eax, msg1 ; Вывод на экран строки
2 call sprintf ; 'Сообщение № 1'
3 _label2:
4 mov eax, msg2 ; Вывод на экран строки
5 call sprintf ; 'Сообщение № 2'
6 _label3:
7 mov eax, msg3 ; Вывод на экран строки
8 call sprintf ; 'Сообщение № 3'
9 _end:
0 call quit ; вызов подпрограммы завершения
```

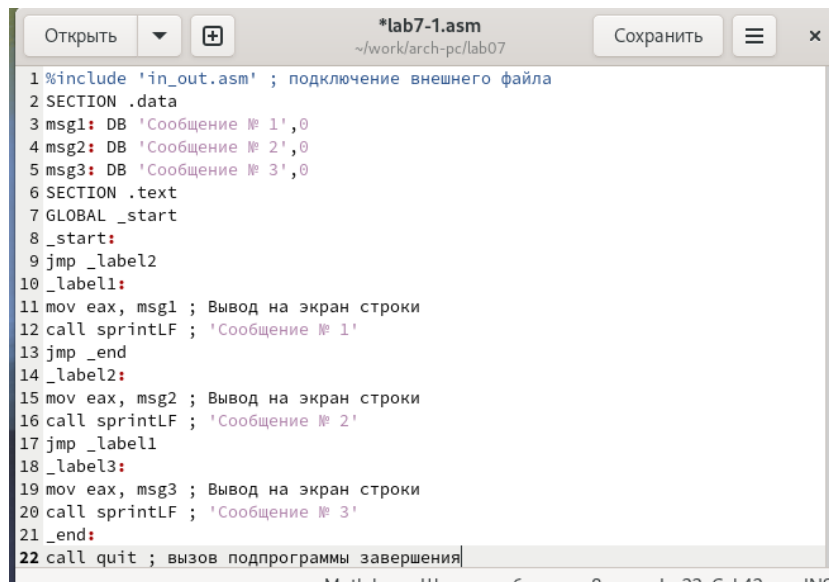
Рис. 4.2: Редактирование файла

Создаю исполняемый файл и запускаю его.(рис. 4.3)


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

Рис. 4.3: Запуск файла

Изменяю текст программы в соответствии с листингом 7.2.(рис. 4.4)



```
*lab7-1.asm
~/work/arch-pc/lab07
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label2
10 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 jmp _end
14 _label2:
15 mov eax, msg2 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 2'
17 jmp _label1
18 _label3:
19 mov eax, msg3 ; Вывод на экран строки
20 call sprintf ; 'Сообщение № 3'
21 _end:
22 call quit ; вызов подпрограммы завершения
```

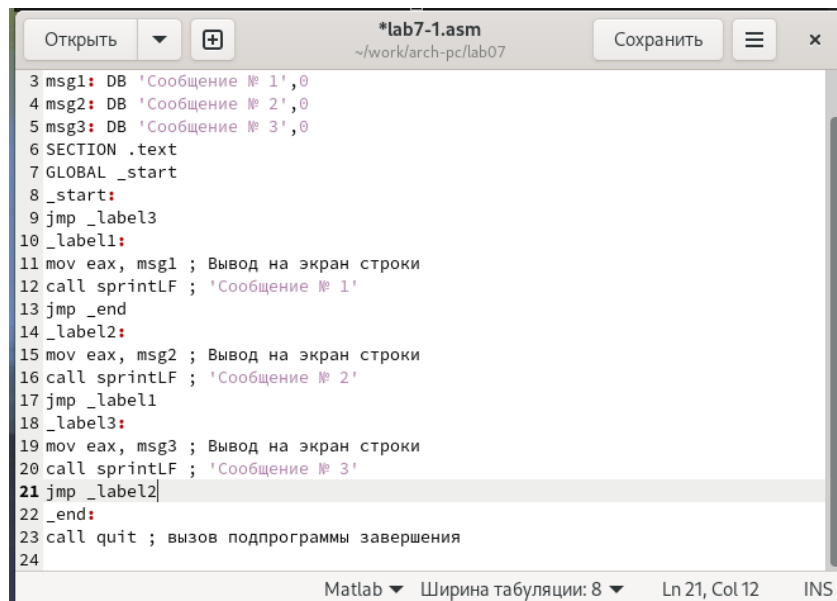
Рис. 4.4: Редактирование файла

Создаю исполняемый файл и запускаю его.(рис. 4.5)

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

Рис. 4.5: Запуск файла

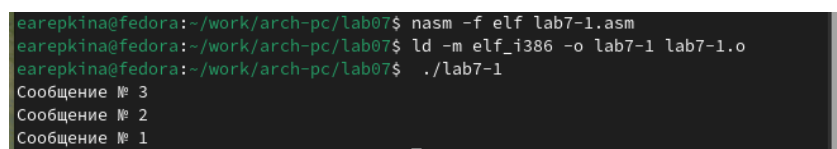
Изменяю текст программы добавив или изменив инструкции jmp (рис. 4.6)



```
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label3
10 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 jmp _end
14 _label2:
15 mov eax, msg2 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 2'
17 jmp _label1
18 _label3:
19 mov eax, msg3 ; Вывод на экран строки
20 call sprintf ; 'Сообщение № 3'
21 jmp _label2
22 _end:
23 call quit ; вызов подпрограммы завершения
24
```

Рис. 4.6: Редактирование файла

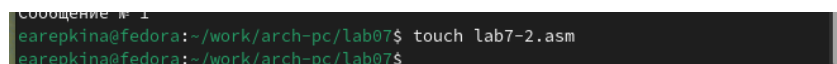
Создаю исполняемый файл и запускаю его.(рис. 4.7)



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

Рис. 4.7: Запуск файла

Создаю файл с названием lab7-2.asm (рис. 4.8)



```
Сообщение № 1
earepkina@fedora:~/work/arch-pc/lab07$ touch lab7-2.asm
earepkina@fedora:~/work/arch-pc/lab07$
```

Рис. 4.8: Создание файла

ввожу в него текст программы из листинга 7.3 (рис. 4.9)

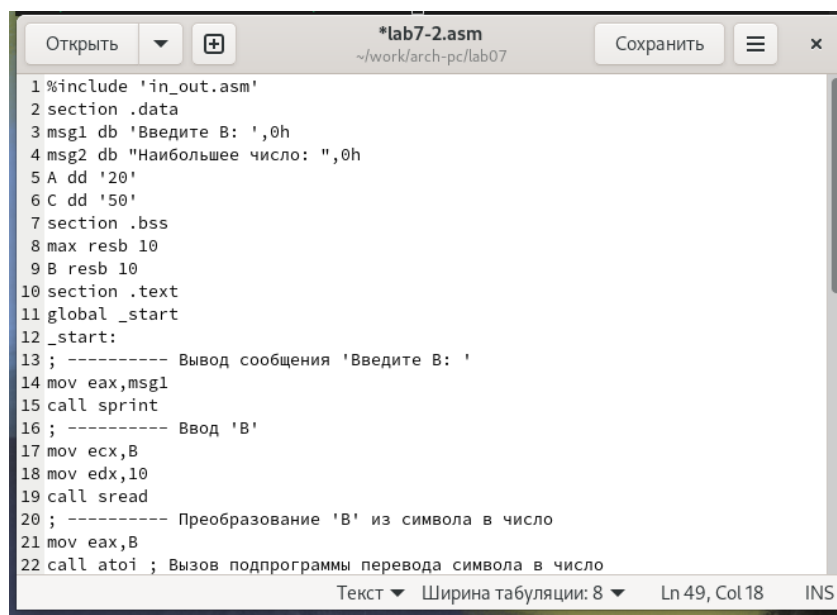


Рис. 4.9: Редактирование файла

Создаю исполняемый файл и запускаю его. (рис. 4.10)

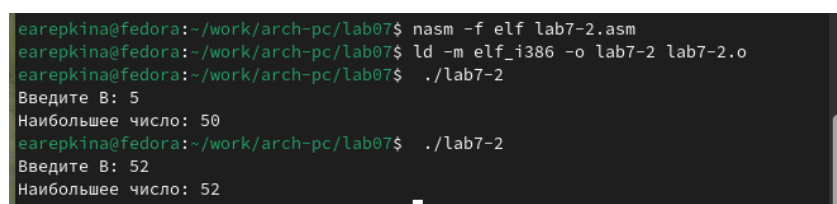


Рис. 4.10: Запуск исполняемого файла

При введении числа до 50, программа выводит наибольшее число 50, при введении числа больше 50, программа выводит введенное нами число.

2. Изучение структуры файлы листинга

Создаю файл листинга для программы из файла (рис. 4.11)

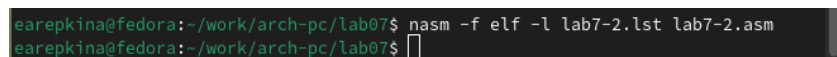
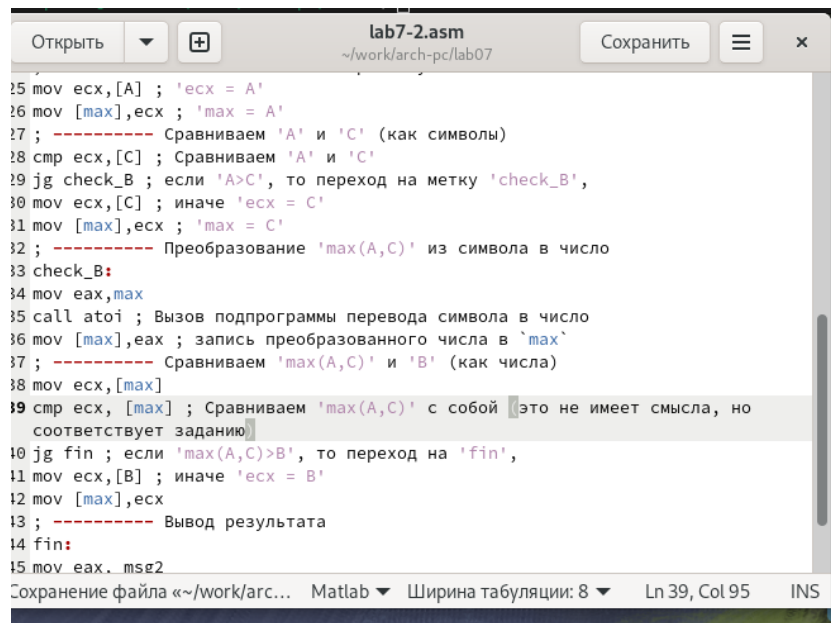


Рис. 4.11: Создание файла

Открываю файл листинга lab7-2.lst с помощью любого текстового редактора, например mcedit (рис. 4.12)



```
25 mov ecx,[A] ; 'ecx = A'
26 mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 jg check_B ; если 'A>C', то переход на метку 'check_B',
30 mov ecx,[C] ; иначе 'ecx = C'
31 mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 mov eax,max
35 call atoi ; Вызов подпрограммы перевода символа в число
36 mov [max],eax ; запись преобразованного числа в 'max'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 mov ecx,[max]
39 cmp ecx,[max] ; Сравниваем 'max(A,C)' с собой [это не имеет смысла, но
соответствует заданию]
40 jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 mov ecx,[B] ; иначе 'ecx = B'
42 mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 mov_eax_msg2
Сохранение файла «~/work/arc... Matlab ▾ Ширина табуляции: 8 ▾ Ln 39, Col 95 INS
```

Рис. 4.13: Изменение файла

Пытаюсь создать файл листинга, но он не создается из-за ошибки

Задание для самостоятельной работы

1. Создаю файл lab7-3.asm, пишу программу для нахождения наименьшего из 3 переменных (14 вариантов исходя из 6 лабораторной) (рис. 4.14)

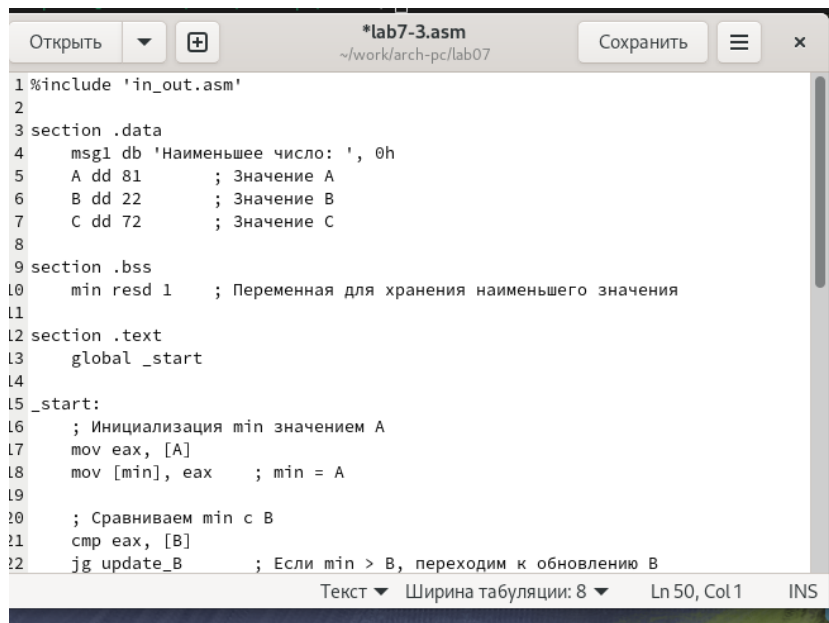


Рис. 4.14: Редактирование файла

Проверяю работу программы, программа работает верно. (рис. 4.15)

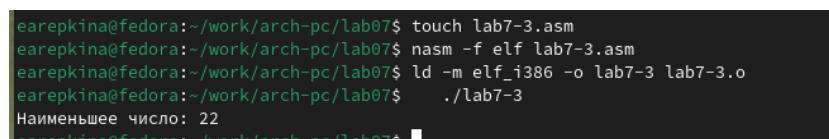
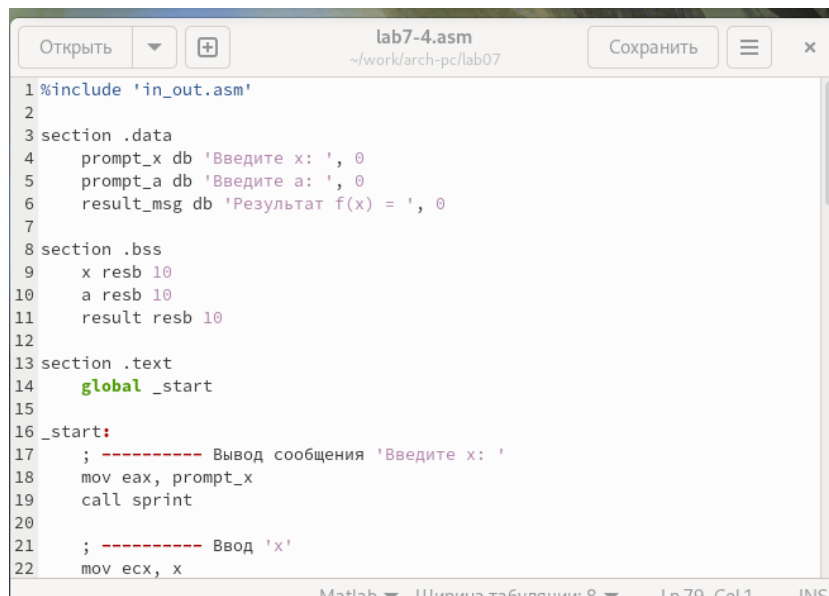


Рис. 4.15: Запуск исполняемого файла

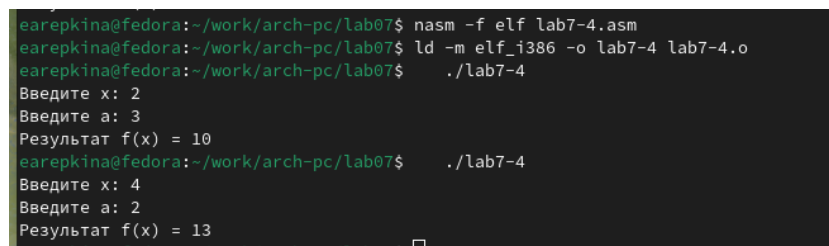
2. Создаю файл с названием lab7-4.asm, пишу программу для вычисления $f(x)$, пишу программу для функции исходя из своего варианта, полученного в ходе лабораторной работы номер 6, номер моего варианта 14 (рис. 4.16)



```
1 %include 'in_out.asm'
2
3 section .data
4     prompt_x db 'Введите x: ', 0
5     prompt_a db 'Введите a: ', 0
6     result_msg db 'Результат f(x) = ', 0
7
8 section .bss
9     x resb 10
10    a resb 10
11    result resb 10
12
13 section .text
14     global _start
15
16 _start:
17     ; ----- Вывод сообщения 'Введите x: '
18     mov eax, prompt_x
19     call sprint
20
21     ; ----- Ввод 'x'
22     mov ecx, x
```

Рис. 4.16: Редактирование файла

Создаю исполняемый файл и запускаю его. (рис. 4.17)



```
earepkina@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
earepkina@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
earepkina@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 2
Введите a: 3
Результат f(x) = 10
earepkina@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 4
Введите a: 2
Результат f(x) = 13
```

Рис. 4.17: Запуск исполняемого файла

Произведя несложные математические вычисления, делаю вывод, что программа работает верно

5 Выводы

В результате выполнения данной лабораторной работы, я изучила команды условного и безусловного переходов, приобрела навыки написания программ с использованием переходов и познакомилась с назначением и структурой файла листинга

Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 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/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс,
- 11.
12. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
13. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
14. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
15. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-

- е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.
16. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
17. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер,
18. — 1120 с. — (Классика Computer Science).