РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7

дисциплина: Архитектура компьютера

Студент: Тимаков Макарий Владимирович

Группа: НПИбд 02-25

МОСКВА

2025г.

1.Цель работы

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

2. Выполнение лабораторной работы

2.1 Реализация переходов в NASM

Создаем каталог для программ ЛБ7 и в нем создаем файл (рис.2.1)

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

Рис. 2.1 Создаем каталог с помощью команды mkdir и файл с помощью команды touch

Открываем файл в Midnight Commander, копируем исходгый файл in_out (рис 2.2) и заполняем созданный файл lab7-1 в соответствии с листингом 7.1 (рис. 2.3)

Рис. 2.2 Копируем файл in_out

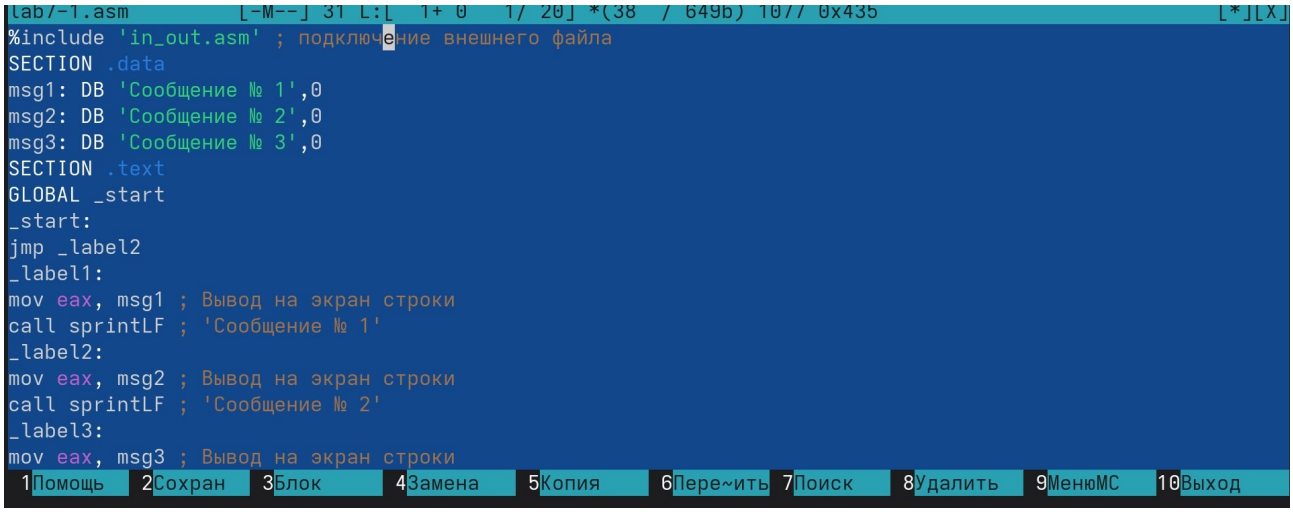

Рис. 2.3 Заполняем файл

Создаем исполняемый файл и запускаем его (рис. 2.4)

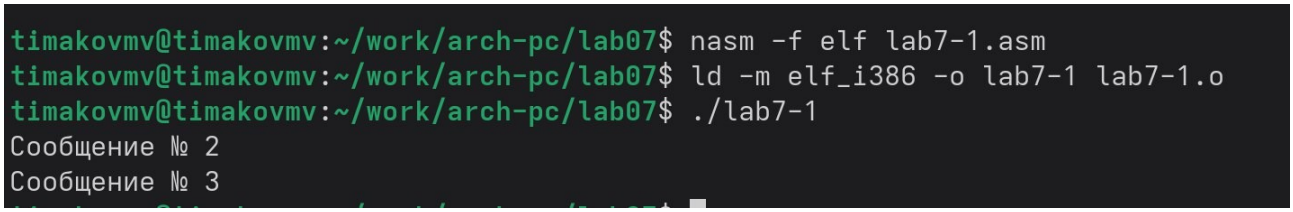

Рис. 2.4 Запускаем файл и смотрим на его работу

Снова открываем файл для редактирования и изменяем его в соответствии с листингом 7.2 (рис. 2.5)

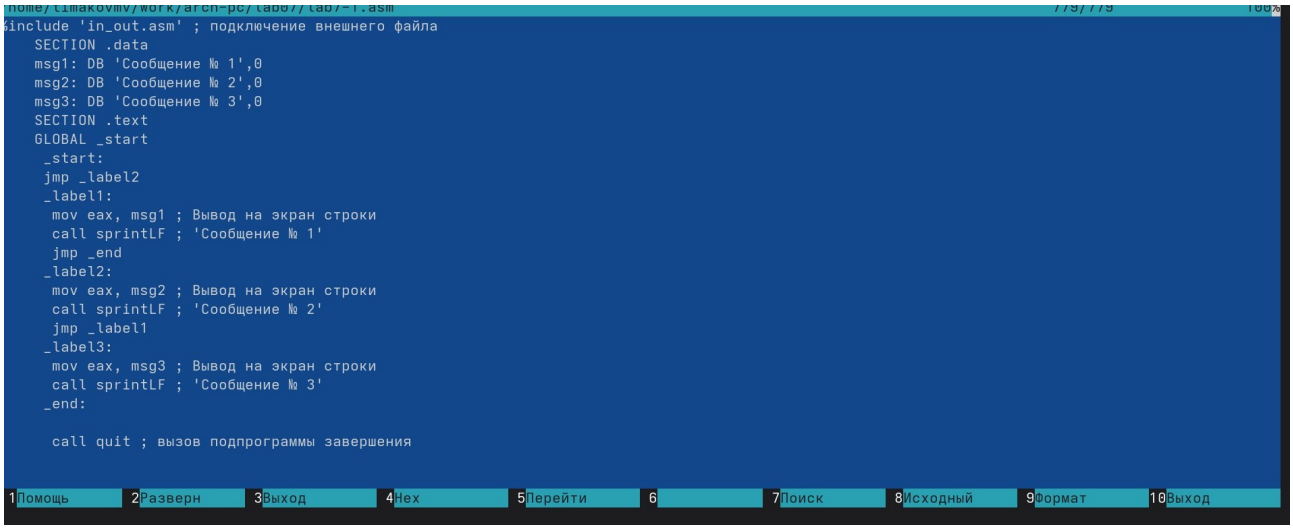

Рис. 2.5 Изменяем файл

Создаем исполняемый файл и запускаем его (рис. 2.6).

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

Рис. 2.6 Запускаем файл и смотрим на его работу

Снова открываем файл для редактирования и изменяем его чтобы произошел данный вывод (рис. 2.7)

```
%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 _label3
     _label1:
      mov eax, msg1 ; Вывод на экран строки
      call sprintLF ; 'Сообщение № 1'
      jmp _end
     _label2:
      mov eax, msg2 ; Вывод на экран строки
      call sprintLF ; 'Сообщение № 2'
      jmp _label1
     _label3:
      mov eax, msg3 ; Вывод на экран строки
      call sprintLF ; 'Сообщение № 3'
      jmp _label2
     _end:

      call quit ; вызов подпрограммы завершения

1Помощь 2Разверн 3Выход 4Hex 5Перейти 6 7Поиск 8Исходный 9Формат 10Выход
```

Рис. 2.7 Редактируем файл

Создаем исполняемый файл и запускаем его (рис. 2.8).

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

Рис. 2.8 Проверяем, сошелся ли наш вывод с данным в условии выводом

Создаем новый файл (рис. 2.9)

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

Рис. 2.9 Создаем файл командой touch

Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 7.3 (рис. 2.10)

```
%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
80 Демидова А. В.
Архитектура ЭВМ
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',
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Переместить 7Поиск 8Удалить 9МенюMC 10Выход
```

Рис. 2.10 Заполняем файл

Создаем исполняемый файл и проверяем его работу, вводя разные значения B (рис. 2.11).

```
timakovmv@timakovmv:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
timakovmv@timakovmv:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
timakovmv@timakovmv:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 4
Наибольшее число: 50
timakovmv@timakovmv:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
timakovmv@timakovmv:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
timakovmv@timakovmv:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 5
Наибольшее число: 50
timakovmv@timakovmv:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
timakovmv@timakovmv:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
timakovmv@timakovmv:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 1
Наибольшее число: 50
timakovmv@timakovmv:~/work/arch-pc/lab07$
```

Рис. 2.11 Смотрим на работу программ

2.2 Изучение структуры файла листинга

Создаем файл листинга для программы lab7-2.asm (рис. 2.12)

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

Рис.2.12 Создаем файл листинга

Открываем файл листинга с помощью команды mcedit и изучаем его (рис. 2.13)

```
lab7-2.lst        [---O]  0 L:[ 27+ 0  27/225] *(1706/14456b) 0032 0x020                [*][X]
   26 00000012 50                  <1>     push    eax
   27 00000013 E8E8FFFFFF          <1>     call    slen
   28                              <1>.....
   29 00000018 89C2                <1>     mov     edx, eax
   30 0000001A 58                  <1>     pop     eax
   31                              <1>.....
   32 0000001B 89C1                <1>     mov     ecx, eax
   33 0000001D BB01000000          <1>     mov     ebx, 1
   34 00000022 B804000000          <1>     mov     eax, 4
   35 00000027 CD80                <1>     int     80h
   36                              <1>.
   37 00000029 5B                  <1>     pop     ebx
   38 0000002A 59                  <1>     pop     ecx
   39 0000002B 5A                  <1>     pop     edx
   40 0000002C C3                  <1>     ret
   41                              <1>.
   42                              <1>.
 1Помощь  2Сохран  3Блок  4Замена  5Копия  6Пере~ить  7Поиск  8Удалить  9МенюMC  10Выход
```

Рис. 2.13 Изучаем файл

Строка 33: 0000001D-адрес в сегменте кода,BB01000000-машинный код,mov ebx,1-присвоение переменной ecx 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.

Открываем файл и удаляем один операндум (рис.2.14).

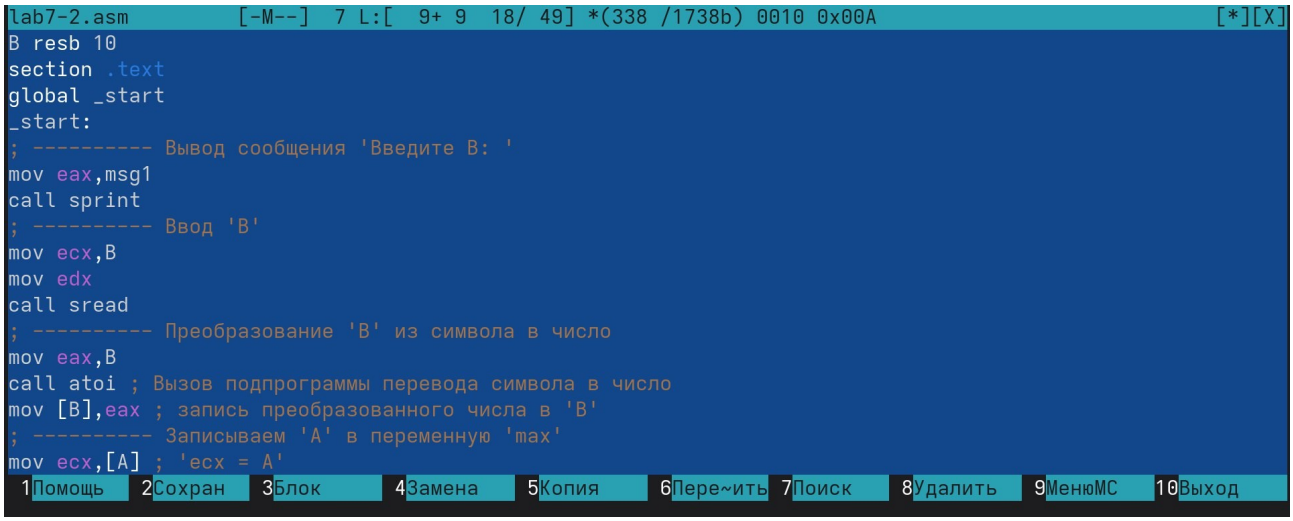

Рис. 2.14 Удаляем операндум из файла

Транслируем с получением файла листинга (рис. 2.15).

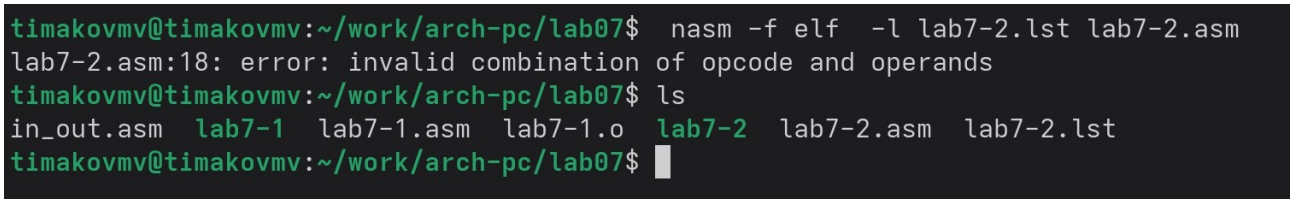

Рис. 2.15 Транслируем файл

При трансляции файла выдается ошибка, но создаются исполнительный файл lab7-2 и lab7-2.lst

Снова открываем файл листинга и изучаем его (рис. 2.16).

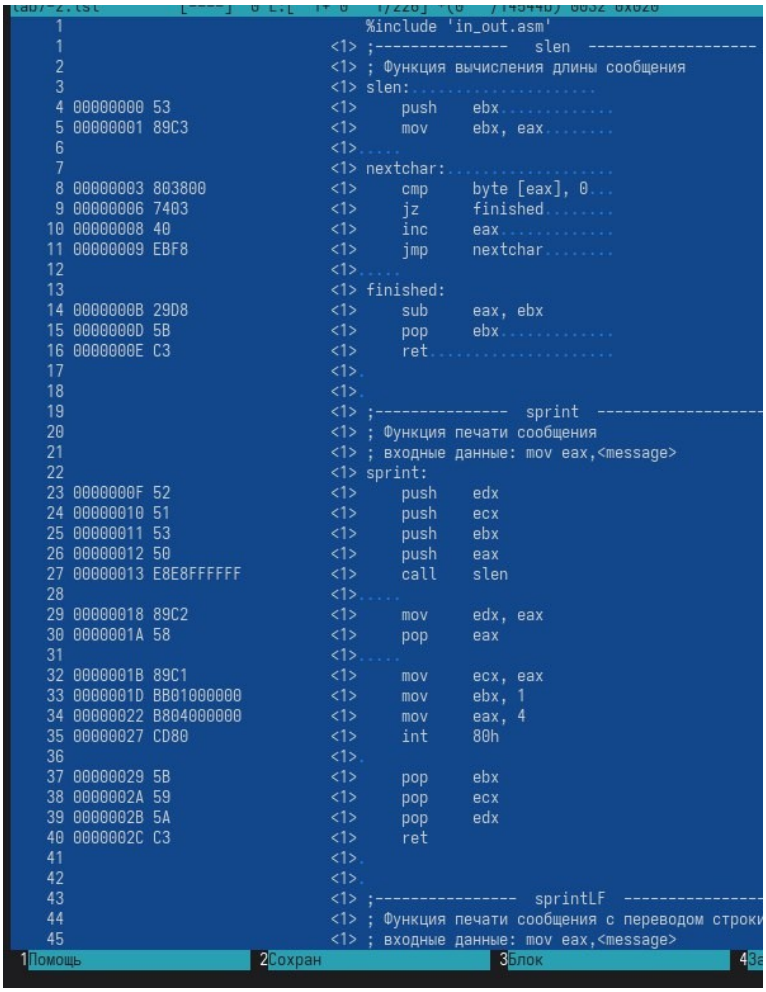

Рис. 2.16 Анализируем файл с ошибкой

Создается ТОЛЬКО файл листинга `lab7-2.lst`. Файл объектного кода lab7-2.o НЕ создается, потому что ассемблер обнаружил ошибку и остановил компиляцию. При наличии синтаксических ошибок в исходном коде: - Объектный файл (.o) не создается - компиляция прерывается - Листинг создается, но содержит сообщения об ошибках - это помогает локализовать проблему - Ассемблер указывает точное место и характер ошибки - в данном случае "ожидалась инструкция" из-за неполной команды

3. Задание для самостоятельной работы

ВАРИАНТ 19

1.Напишите программу нахождения наименьшей из 3 целочисленных переменных a,b и c. Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 6. Создайте исполняемый файл и проверьте его работу.

Создаем новый файл (рис. 3.1)

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

Рис. 3.1 Создаем файл командой touch и редактируем в Midnight Commander

Открываем его и пишем программу, которая выберет наименьшее число из трех (2 числа уже в программе, третье вводится с консоли)(рис. 3.2)

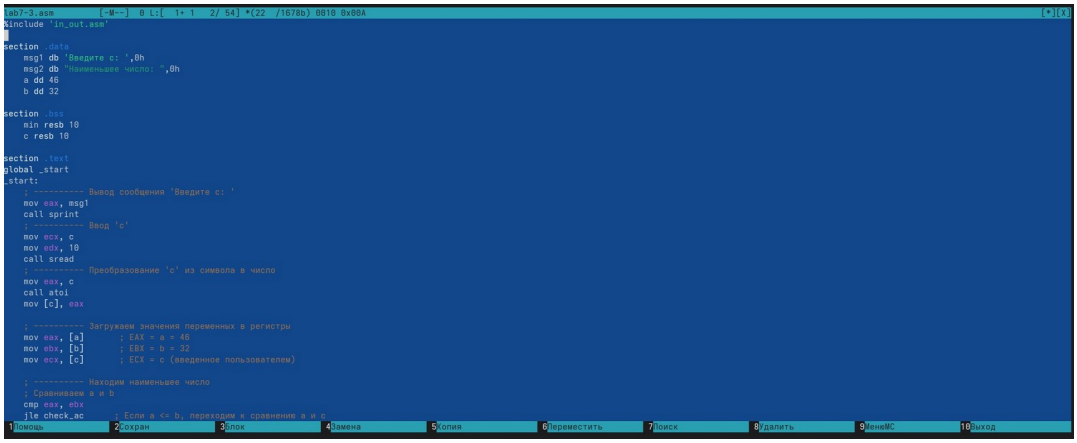

Рис. 3.2 Прописываем программу

Транслируем файл и смотрим на работу программы (рис. 3.3)

```
timakovmv@timakovmv:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
timakovmv@timakovmv:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
timakovmv@timakovmv:~/work/arch-pc/lab07$ ./lab7-3
Введите c: 74
Наименьшее число: 32
timakovmv@timakovmv:~/work/arch-pc/lab07$
```

Рис. 3.3 Смотрим на работу выполненного файла (всё верно)

2. Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 6. Создайте исполняемый файл и проверьте его работу для значений x и a из 7.6.

Создаем новый файл (рис. 3.4).

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

Рис. 3.4 Создаем файл командой touch

Открываем его и пишем программу, которая решит систему уравнений при данных, введенных в консоль (рис. 3.5)

Рис. 3.5 Прописываем программу

Транслируем файл и проверяем его работу при x = 4 и a = 5 (рис. 3.6).

```
timakovmv@timakovmv:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
timakovmv@timakovmv:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
timakovmv@timakovmv:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 4
Введите a: 5
Результат f(x) = 4 (x <= a, f(x) = x)
timakovmv@timakovmv:~/work/arch-pc/lab07$
```

Рис 3.6 Проверяем работу программы (работает верно)

Транслируем файл и проверяем его работу при x = 5 и a = 2 (рис 3.7)

```
timakovmv@timakovmv:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
timakovmv@timakovmv:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
timakovmv@timakovmv:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 5
Введите a: 2
Результат f(x) = 7 (x > a, f(x) = a + x)
timakovmv@timakovmv:~/work/arch-pc/lab07$
```

Рис. 3.7 Проверяем работу программы (работает верно)

4 Выводы

Мы научились решать программы с использованием циклов и обработкой аргументов командной строки.