

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №

дисциплина: Архитектура компьютера

Студент: Татур Платон Андреевич

Группа: НКАбд-05-25

МОСКВА

2025 г.

8.1. Цель работы

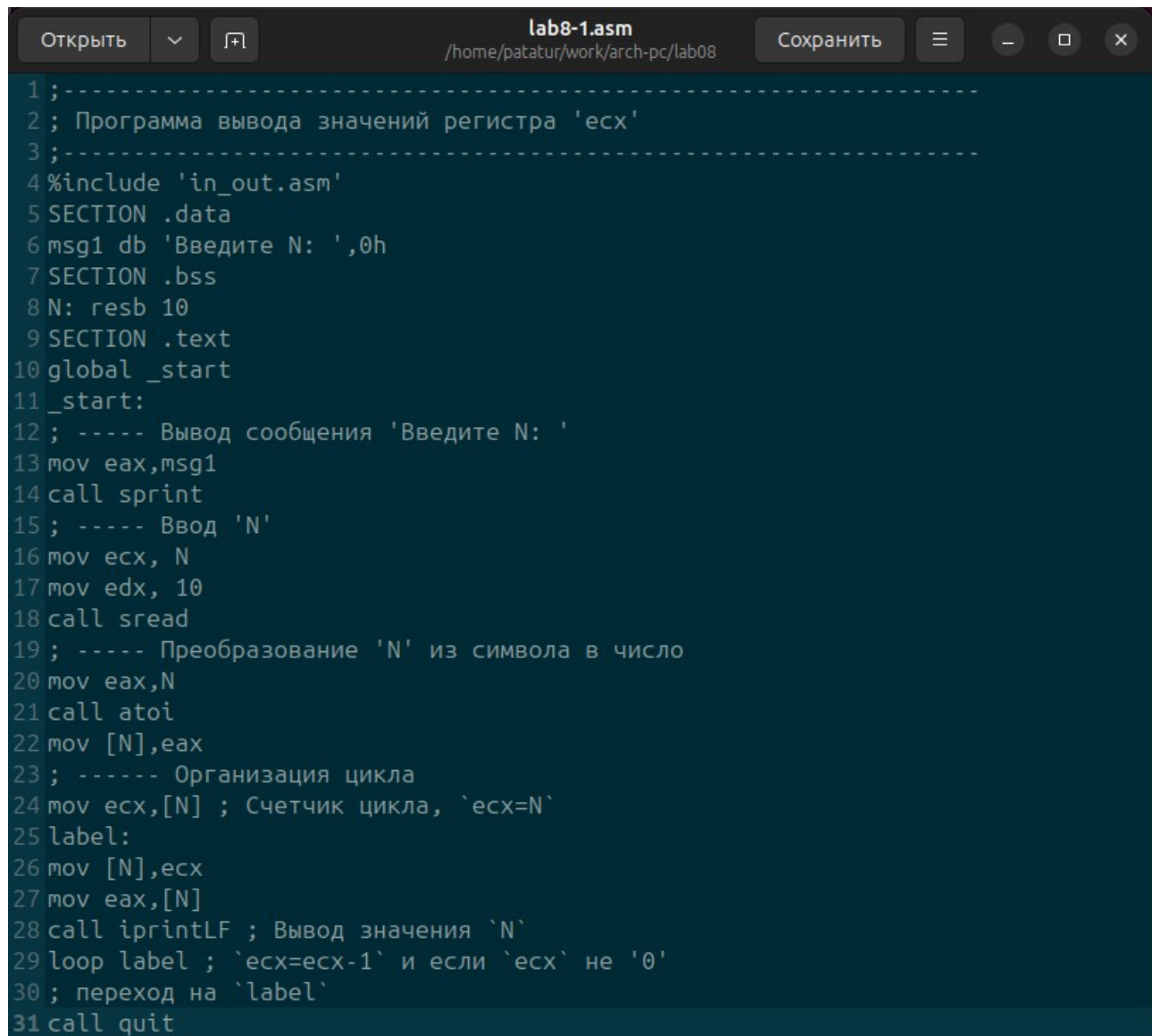
Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

8.3. Порядок выполнения лабораторной работы

8.3.1. Реализация циклов в NASM

```
patatur@platonpc:~$ mkdir ~/work/arch-pc/lab08
patatur@platonpc:~$ cd ~/work/arch-pc/lab08
patatur@platonpc:~/work/arch-pc/lab08$ touch lab8-1.asm
patatur@platonpc:~/work/arch-pc/lab08$
```

Рис 1



```
lab8-1.asm
/home/patatur/work/arch-pc/lab08
Сохранить

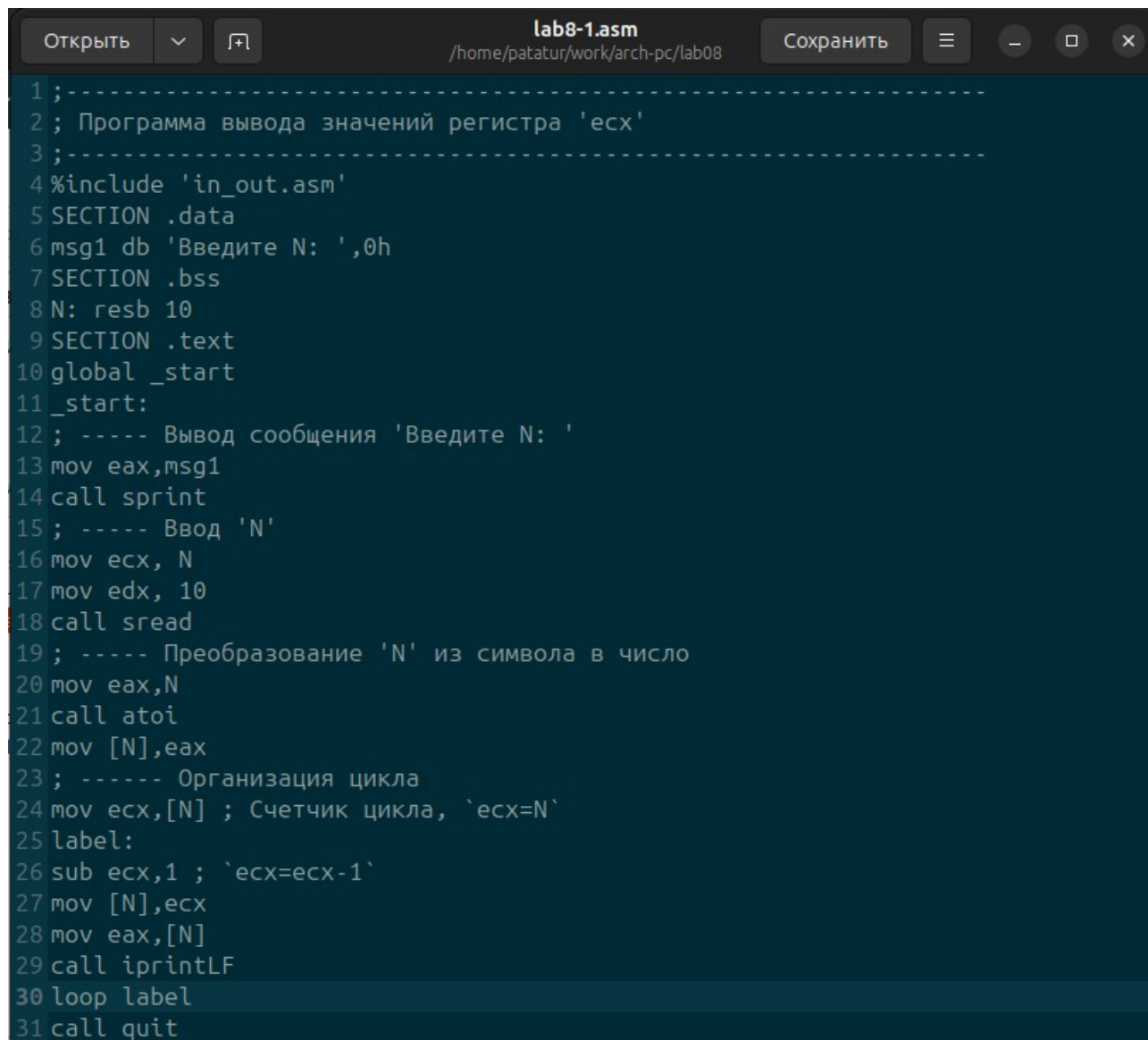
1 ;-----
2 ; Программа вывода значений регистра 'ecx'
3 ;-----
4 %include 'in_out.asm'
5 SECTION .data
6 msg1 db 'Введите N: ',0h
7 SECTION .bss
8 N: resb 10
9 SECTION .text
10 global _start
11 _start:
12 ; ----- Вывод сообщения 'Введите N: '
13 mov eax,msg1
14 call sprint
15 ; ----- Ввод 'N'
16 mov ecx, N
17 mov edx, 10
18 call sread
19 ; ----- Преобразование 'N' из символа в число
20 mov eax,N
21 call atoi
22 mov [N],eax
23 ; ----- Организация цикла
24 mov ecx,[N] ; Счетчик цикла, `ecx=N`
25 label:
26 mov [N],ecx
27 mov eax,[N]
28 call iprintLF ; Вывод значения `N`
29 loop label ; `ecx=ecx-1` и если `ecx` не '0'
30 ; переход на `label`
31 call quit
```

Рис 1.2

```
patatur@platonpc:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
patatur@platonpc:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
patatur@platonpc:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 5
5
4
3
2
1
patatur@platonpc:~/work/arch-pc/lab08$
```

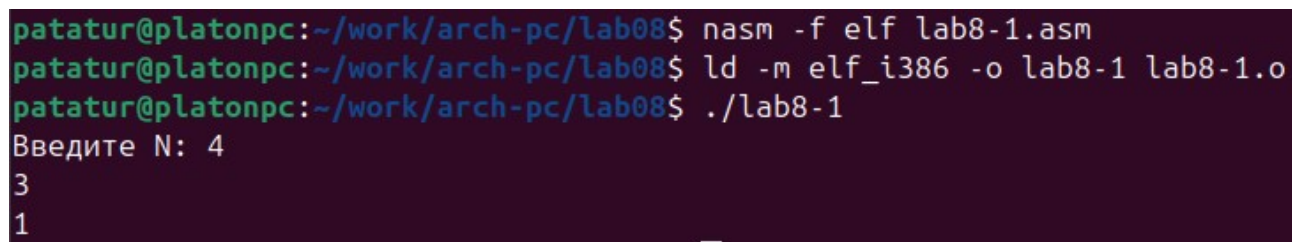
Рис 1.3

- 1) Создал каталог для программ лабораторной работы № 8, перешел в него и создал файл lab8-1.asm (см Рис 1)
 - 2) Ввел в файл lab8-1.asm текст программы из листинга 8.1. (см Рис 1.2)
 - 3) Создал исполняемый файл и проверил его работу (см Рис 1.3)
-



```
1 ;-----
2 ; Программа вывода значений регистра 'ecx'
3 ;-----
4 %include 'in_out.asm'
5 SECTION .data
6 msg1 db 'Введите N: ',0h
7 SECTION .bss
8 N: resb 10
9 SECTION .text
10 global _start
11 _start:
12 ; ----- Вывод сообщения 'Введите N: '
13 mov eax,msg1
14 call sprint
15 ; ----- Ввод 'N'
16 mov ecx, N
17 mov edx, 10
18 call sread
19 ; ----- Преобразование 'N' из символа в число
20 mov eax,N
21 call atoi
22 mov [N],eax
23 ; ----- Организация цикла
24 mov ecx,[N] ; Счетчик цикла, `ecx=N`
25 label:
26 sub ecx,1 ; `ecx=ecx-1`
27 mov [N],ecx
28 mov eax,[N]
29 call iprintLF
30 loop label
31 call quit
```

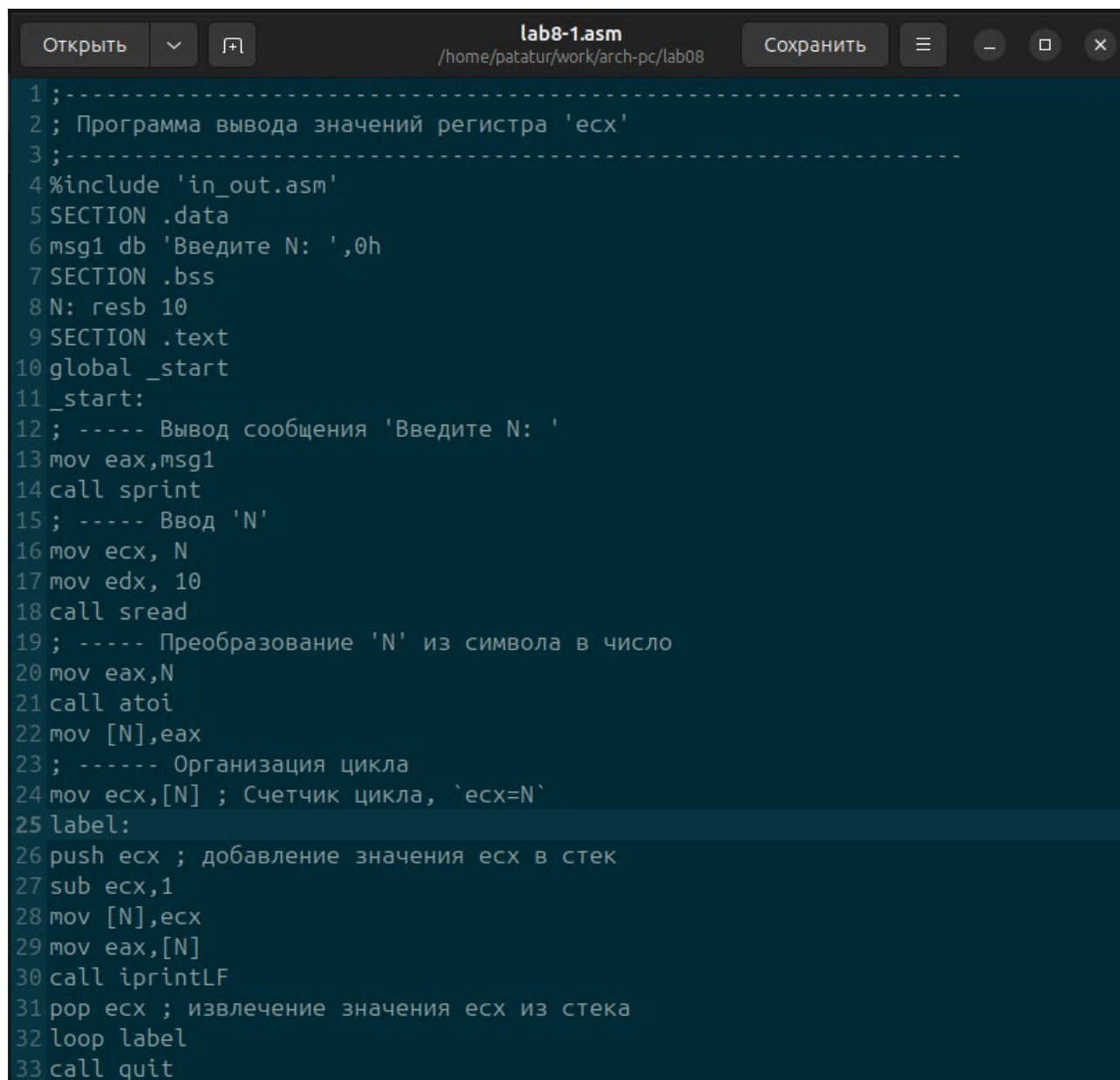
Рис 1.4



```
patatur@platonpc:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
patatur@platonpc:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
patatur@platonpc:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 4
3
1
```

Рис 1.5

- 1) Изменил текст программы добавив изменение значение регистра есх в цикле (см Рис 1.4)
 - 2) Создал исполняемый файл и проверил его работу (см Рис 1.5)
 - 3) есх принимает значения 3 2 1 0, но выводится только 3 и 1 так как есх является счетчиком цикла и понижается с каждым проходом на 2
 - 4) Число проходов цикла не соответствует введенному числу
-



```
1 ;-----
2 ; Программа вывода значений регистра 'есх'
3 ;-----
4 %include 'in_out.asm'
5 SECTION .data
6 msg1 db 'Введите N: ',0h
7 SECTION .bss
8 N: resb 10
9 SECTION .text
10 global _start
11 _start:
12 ; ----- Вывод сообщения 'Введите N: '
13 mov eax,msg1
14 call sprint
15 ; ----- Ввод 'N'
16 mov ecx, N
17 mov edx, 10
18 call sread
19 ; ----- Преобразование 'N' из символа в число
20 mov eax,N
21 call atoi
22 mov [N],eax
23 ; ----- Организация цикла
24 mov ecx,[N] ; Счетчик цикла, `ecx=N`
25 label:
26 push ecx ; добавление значения ecx в стек
27 sub ecx,1
28 mov [N],ecx
29 mov eax,[N]
30 call iprintLF
31 pop ecx ; извлечение значения ecx из стека
32 loop label
33 call quit
```

Рис 1.6

```

patatur@platonpc:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
patatur@platonpc:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
patatur@platonpc:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 5
4
3
2
1
0
patatur@platonpc:~/work/arch-pc/lab08$ 

```

Рис 1.7

- 1) Внес изменения в текст программы добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop (см Рис 1.6)
- 2) Создал исполняемый файл и проверил его работу (см Рис 1.7)
- 3) Число проходов цикла соответствует введенному числу

8.3.2. Обработка аргументов командной строки

```

patatur@platonpc:~/work/arch-pc/lab08$ touch lab8-2.asm
patatur@platonpc:~/work/arch-pc/lab08$ 

```

Рис 2

```

*lab8-2.asm
/home/patatur/work/arch-pc/lab08
Сохранить

1 ;-----
2 ; Обработка аргументов командной строки
3 ;-----
4 %include 'in_out.asm'
5 SECTION .text
6 global _start
7 _start:
8 pop ecx ; Извлекаем из стека в `ecx` количество
9 ; аргументов (первое значение в стеке)
10 pop edx ; Извлекаем из стека в `edx` имя программы
11 ; (второе значение в стеке)
12 sub ecx, 1 ; Уменьшаем `ecx` на 1 (количество
13 ; аргументов без названия программы)
14 next:
15 cmp ecx, 0 ; проверяем, есть ли еще аргументы
16 jz _end ; если аргументов нет выходим из цикла
17 ; (переход на метку `_end`)
18 pop eax ; иначе извлекаем аргумент из стека
19 call sprintf ; вызываем функцию печати
20 loop next ; переход к обработке следующего
21 ; аргумента (переход на метку `next`)
22 _end:
23 call quit

```

Рис 2.2

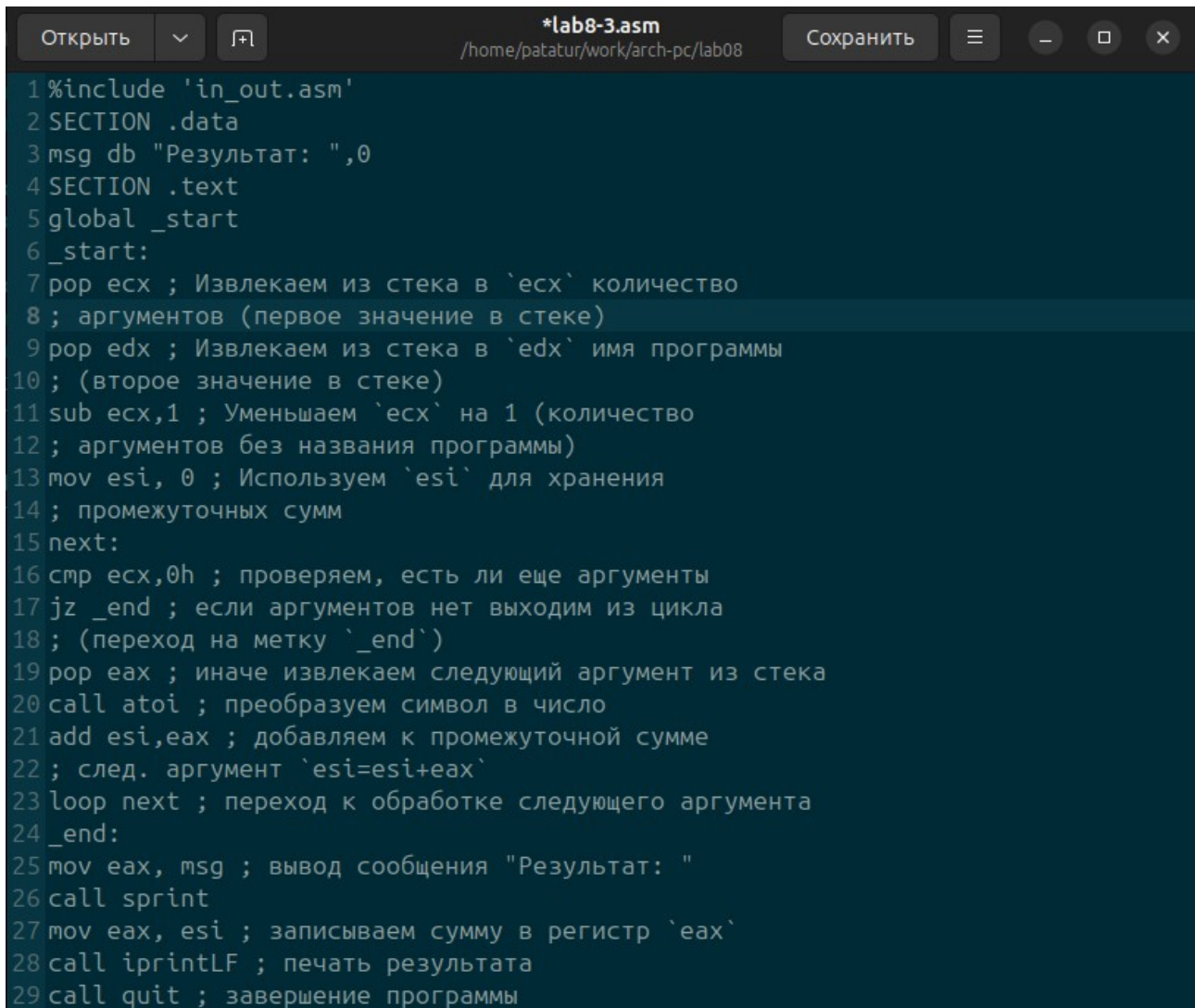
```
patatur@platonpc:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
patatur@platonpc:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
patatur@platonpc:~/work/arch-pc/lab08$
```

Рис 2.3

- 1) Создал файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 (см Рис 2)
 - 2) Ввел в него текст программы из листинга 8.2. (см Рис 2.2)
 - 3) Создал исполняемый файл и запустил его, указав аргументы (см Рис 2.3)
 - 4) Программой было обработано 4 аргумента
-


```
patatur@platonpc:~/work/arch-pc/lab08$ touch lab8-3.asm
patatur@platonpc:~/work/arch-pc/lab08$
```

Рис 3



```
*lab8-3.asm
/home/patatur/work/arch-pc/lab08
Сохранить

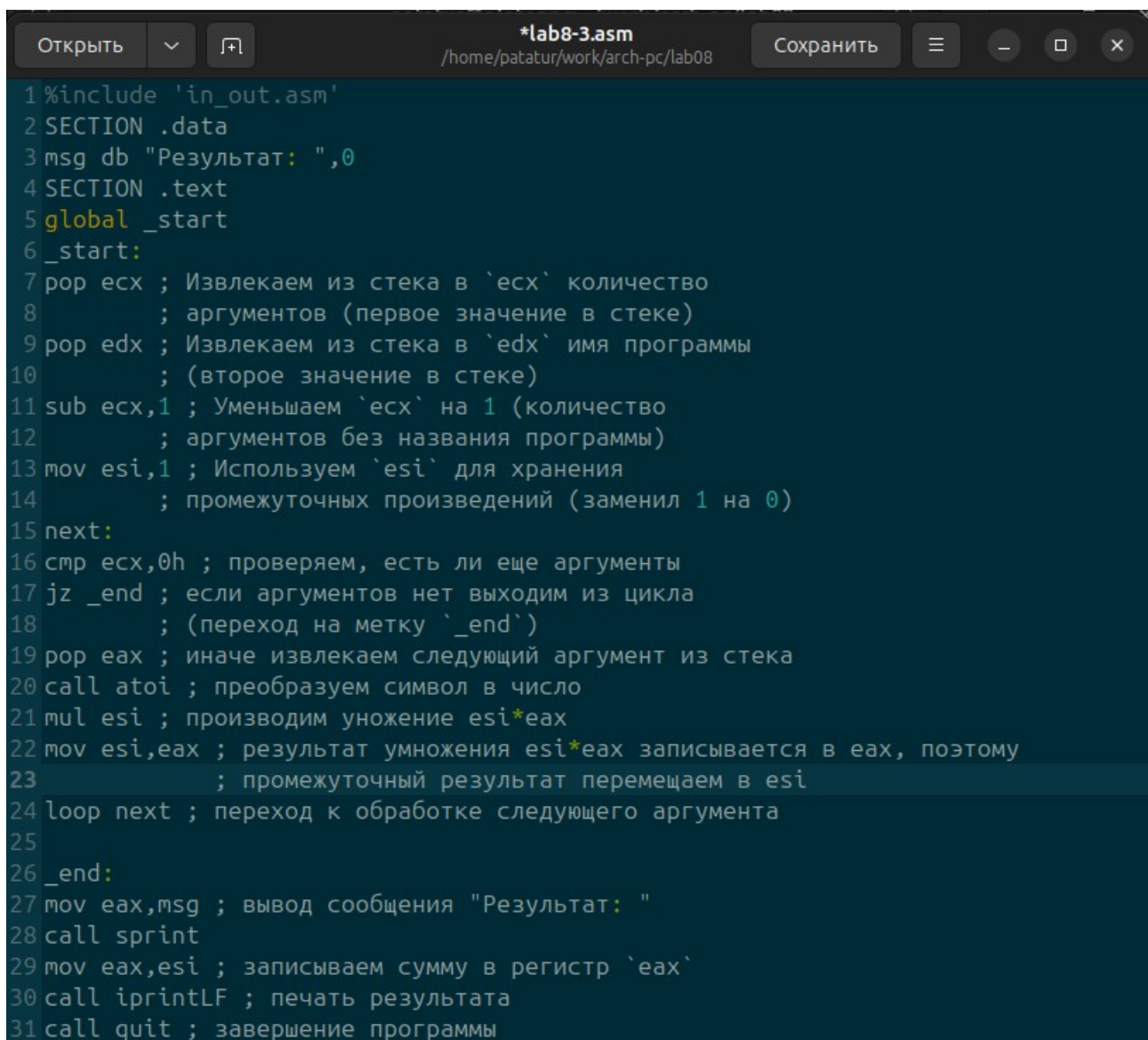
1 %include 'in_out.asm'
2 SECTION .data
3 msg db "Результат: ",0
4 SECTION .text
5 global _start
6 _start:
7 pop ecx ; Извлекаем из стека в `ecx` количество
8 ; аргументов (первое значение в стеке)
9 pop edx ; Извлекаем из стека в `edx` имя программы
10 ; (второе значение в стеке)
11 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
12 ; аргументов без названия программы)
13 mov esi, 0 ; Используем `esi` для хранения
14 ; промежуточных сумм
15 next:
16 cmp ecx,0h ; проверяем, есть ли еще аргументы
17 jz _end ; если аргументов нет выходим из цикла
18 ; (переход на метку `_end`)
19 pop eax ; иначе извлекаем следующий аргумент из стека
20 call atoi ; преобразуем символ в число
21 add esi,eax ; добавляем к промежуточной сумме
22 ; след. аргумент `esi=esi+eax`
23 loop next ; переход к обработке следующего аргумента
24 _end:
25 mov eax, msg ; вывод сообщения "Результат: "
26 call sprint
27 mov eax, esi ; записываем сумму в регистр `eax`
28 call iprintLF ; печать результата
29 call quit ; завершение программы
```

Рис 3.2

```
patatur@platonpc:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
patatur@platonpc:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
patatur@platonpc:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
patatur@platonpc:~/work/arch-pc/lab08$
```

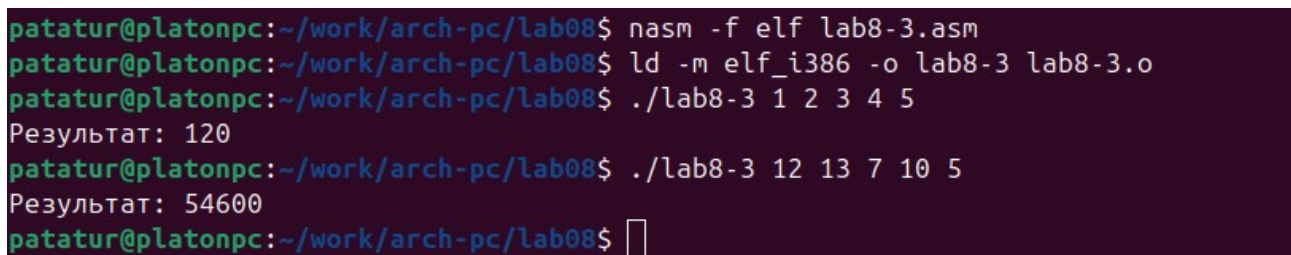
Рис 3.3

- 1) Создал файл lab8-3.asm в каталоге ~/work/arch-pc/lab08 (см Рис 3)
 - 2) Ввел в него текст программы из листинга 8.3. (см Рис 3.2)
 - 3) Создал исполняемый файл и запустил его, указав аргументы
-



```
1 %include 'in_out.asm'
2 SECTION .data
3 msg db "Результат: ",0
4 SECTION .text
5 global _start
6 _start:
7 pop ecx ; Извлекаем из стека в `ecx` количество
8          ; аргументов (первое значение в стеке)
9 pop edx ; Извлекаем из стека в `edx` имя программы
10         ; (второе значение в стеке)
11 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
12          ; аргументов без названия программы)
13 mov esi,1 ; Используем `esi` для хранения
14         ; промежуточных произведений (заменяем 1 на 0)
15 next:
16 cmp ecx,0h ; проверяем, есть ли еще аргументы
17 jz _end ; если аргументов нет выходим из цикла
18         ; (переход на метку `_end`)
19 pop eax ; иначе извлекаем следующий аргумент из стека
20 call atoi ; преобразуем символ в число
21 mul esi ; производим умножение esi*eax
22 mov esi,eax ; результат умножения esi*eax записывается в eax, поэтому
23         ; промежуточный результат перемещаем в esi
24 loop next ; переход к обработке следующего аргумента
25
26 _end:
27 mov eax,msg ; вывод сообщения "Результат: "
28 call sprint
29 mov eax,esi ; записываем сумму в регистр `eax`
30 call iprintLF ; печать результата
31 call quit ; завершение программы
```

Рис 3.4

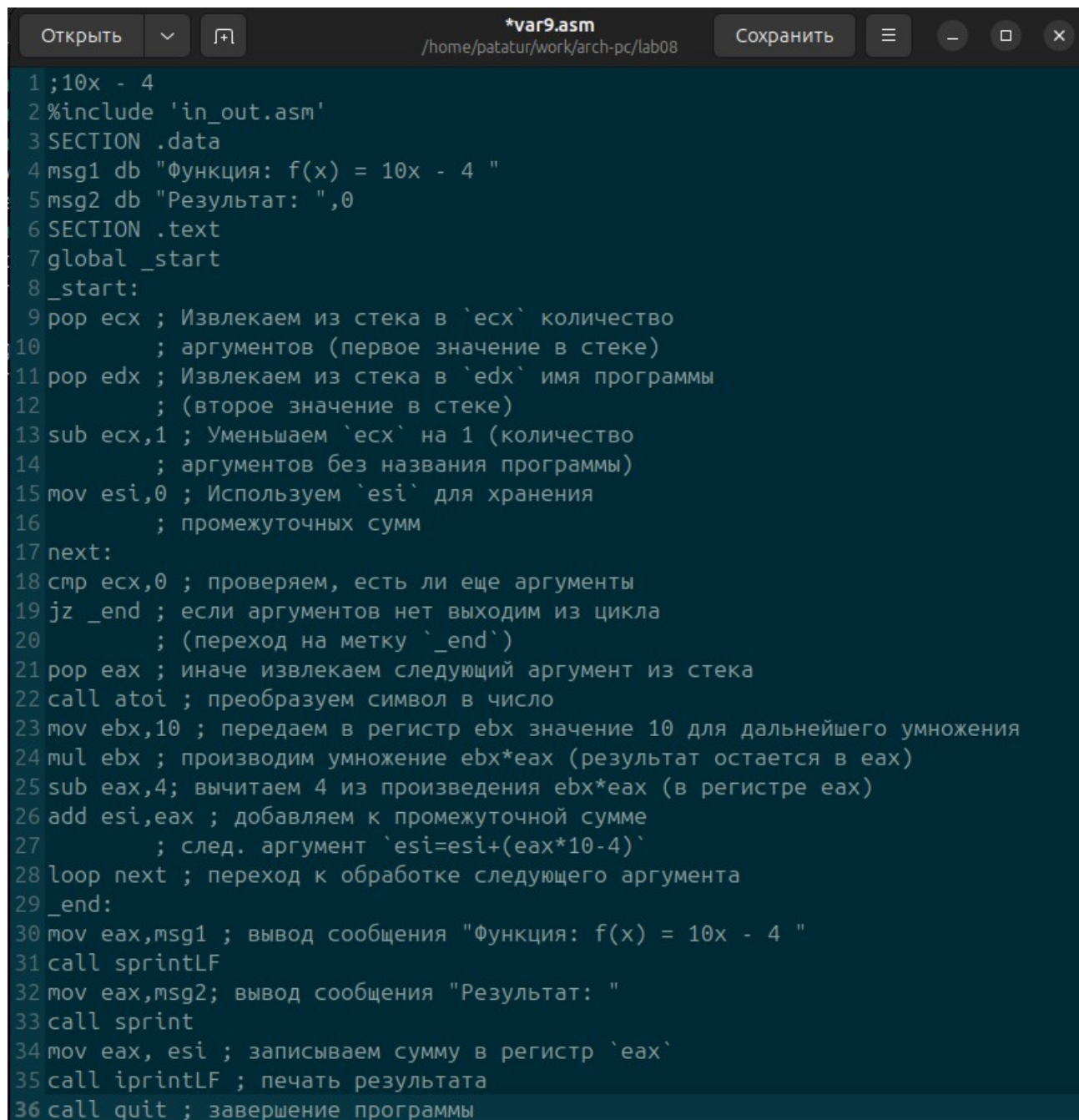


```
patatur@platonpc:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
patatur@platonpc:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
patatur@platonpc:~/work/arch-pc/lab08$ ./lab8-3 1 2 3 4 5
Результат: 120
patatur@platonpc:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 54600
patatur@platonpc:~/work/arch-pc/lab08$
```

Рис 3.5

- 1) Изменил текст программы из листинга 8.3 для вычисления произведения аргументов командной строки.
 - 2) Создал исполняемый файл и запустил его, указав аргументы
-

8.4. Задание для самостоятельной работы



```
1 ;10x - 4
2 %include 'in_out.asm'
3 SECTION .data
4 msg1 db "Функция: f(x) = 10x - 4 "
5 msg2 db "Результат: ",0
6 SECTION .text
7 global _start
8 _start:
9 pop ecx ; Извлекаем из стека в `ecx` количество
10 ; аргументов (первое значение в стеке)
11 pop edx ; Извлекаем из стека в `edx` имя программы
12 ; (второе значение в стеке)
13 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
14 ; аргументов без названия программы)
15 mov esi,0 ; Используем `esi` для хранения
16 ; промежуточных сумм
17 next:
18 cmp ecx,0 ; проверяем, есть ли еще аргументы
19 jz _end ; если аргументов нет выходим из цикла
20 ; (переход на метку `_end`)
21 pop eax ; иначе извлекаем следующий аргумент из стека
22 call atoi ; преобразуем символ в число
23 mov ebx,10 ; передаем в регистр ebx значение 10 для дальнейшего умножения
24 mul ebx ; производим умножение ebx*eax (результат остается в eax)
25 sub eax,4; вычитаем 4 из произведения ebx*eax (в регистре eax)
26 add esi,eax ; добавляем к промежуточной сумме
27 ; след. аргумент `esi=esi+(eax*10-4)`
28 loop next ; переход к обработке следующего аргумента
29 _end:
30 mov eax,msg1 ; вывод сообщения "Функция: f(x) = 10x - 4 "
31 call printf
32 mov eax,msg2; вывод сообщения "Результат: "
33 call printf
34 mov eax,esi ; записываем сумму в регистр `eax`
35 call printf ; печать результата
36 call quit ; завершение программы
```

Рис 4

```
patatur@platonpc:~/work/arch-pc/lab08$ nasm -f elf var9.asm
patatur@platonpc:~/work/arch-pc/lab08$ ld -m elf_i386 -o var9 var9.o
patatur@platonpc:~/work/arch-pc/lab08$ ./var9 1 2 3 4
Функция:  $f(x) = 10x - 4$  Результат:
Результат: 84
patatur@platonpc:~/work/arch-pc/lab08$ ./var9 6 8 9 11
Функция:  $f(x) = 10x - 4$  Результат:
Результат: 324
patatur@platonpc:~/work/arch-pc/lab08$ ./var9 54 66
Функция:  $f(x) = 10x - 4$  Результат:
Результат: 1192
patatur@platonpc:~/work/arch-pc/lab08$
```

Рис 4.2

- 1) Написал код программы (см Рис 4)
- 2) Проверил ее работу (см Рис 4.2) все вычисления выполнены верно

Вывод

В ходе выполнения лабораторной работы я приобрел навыки написания программ с использованием циклов и обработкой аргументов командной строки.