

# **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №9**

**Дисциплина: Архитектура компьютера**

Обрезкова Анастасия Владимировна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
4.1	Задания для самостоятельной работы . . . . .	18
<b>5</b>	<b>Выводы</b>	<b>21</b>
	<b>Список литературы</b>	<b>22</b>

## Список иллюстраций

4.1	Создание, переход в lab09 . . . . .	8
4.2	Ввод текста . . . . .	9
4.3	Результат программы . . . . .	9
4.4	Изменения текста . . . . .	10
4.5	Результат изменений . . . . .	11
4.6	Изменила программу . . . . .	12
4.7	Вывела результат . . . . .	13
4.8	Ввела нужный текст . . . . .	13
4.9	Вывод результата . . . . .	14
4.10	Создание файла . . . . .	14
4.11	Открытие файла . . . . .	15
4.12	Текст программы . . . . .	16
4.13	Результат программы . . . . .	16
4.14	Текст программы . . . . .	17
4.15	Результат программы . . . . .	17
4.16	Текст программы . . . . .	19
4.17	Результат . . . . .	20

## **Список таблиц**

# 1 Цель работы

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

## 2 Задание

Приобрести навыки написания программ с использованием циклов и обработки аргументов командной строки.

### 3 Теоретическое введение

Стек — это структура данных, организованная по принципу LIFO («Last In — First Out» или «последним пришёл — первым ушёл»). Стек является частью архитектуры процессора и реализован на аппаратном уровне. Для работы со стеком в процессоре есть специальные регистры (ss, bp, sp) и команды.

Основной функцией стека является функция сохранения адресов возврата и передачи аргументов при вызове процедур. Кроме того, в нём выделяется память для локальных переменных и могут временно храниться значения регистров.

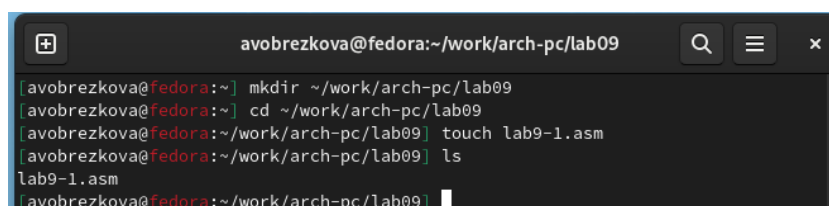
Команда push размещает значение в стеке, т.е. помещает значение в ячейку памяти, на которую указывает регистр esp, после этого значение регистра esp увеличивается на 4. Данная команда имеет один операнд — значение, которое необходимо поместить в стек.

Команда pop извлекает значение из стека, т.е. извлекает значение из ячейки памяти, на которую указывает регистр esp, после этого уменьшает значение регистра esp на 4. У этой команды также один операнд, который может быть регистром или переменной в памяти.

Для организации циклов существуют специальные инструкции. Для всех инструкций максимальное количество проходов задаётся в регистре ecx. Наиболее простой является инструкция loop.

## 4 Выполнение лабораторной работы

1. Создала каталог для программ лабораторной работы №9, перешла в него и создала файл lab9-1.asm. (рис. 4.1)

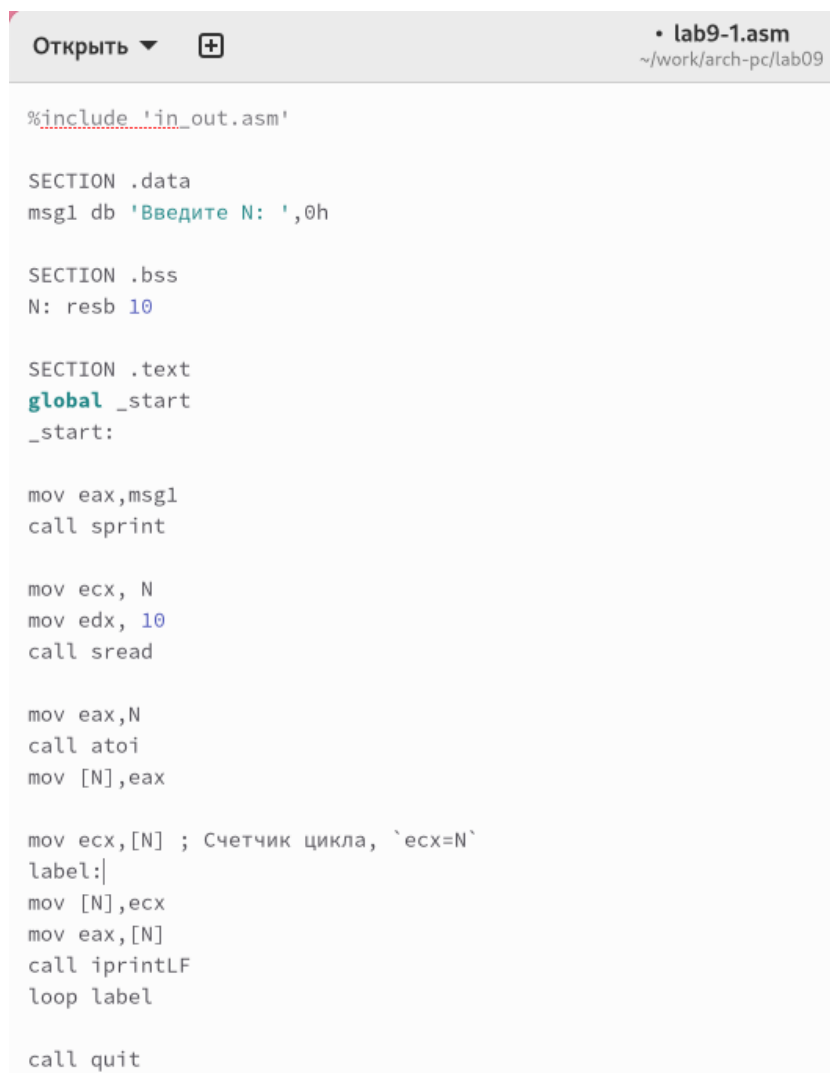


```
avobrezkova@fedora:~/work/arch-pc/lab09
[avobrezkova@fedora:~] mkdir ~/work/arch-pc/lab09
[avobrezkova@fedora:~] cd ~/work/arch-pc/lab09
[avobrezkova@fedora:~/work/arch-pc/lab09] touch lab9-1.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] ls
lab9-1.asm
[avobrezkova@fedora:~/work/arch-pc/lab09]
```

Рис. 4.1: Создание, переход в lab09

2. Ввела в файл lab9-1 нужный текст программы из листинга 9.1., создала исполняемый файл и вывела результат. (рис. 4.2; рис. 4.3)






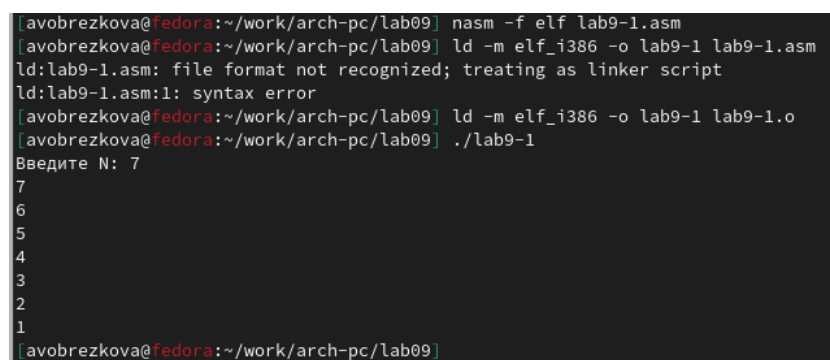
```
Открыть ▾  • lab9-1.asm  
~/work/arch-pc/lab09  
  
%include 'in_out.asm'  
  
SECTION .data  
msg1 db 'Введите N: ',0h  
  
SECTION .bss  
N: resb 10  
  
SECTION .text  
global _start  
_start:  
  
mov eax,msg1  
call sprint  
  
mov ecx, N  
mov edx, 10  
call sread  
  
mov eax,N  
call atoi  
mov [N],eax  
  
mov ecx,[N] ; Счетчик цикла, `ecx=N`  
label:|  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
loop label  
  
call quit
```

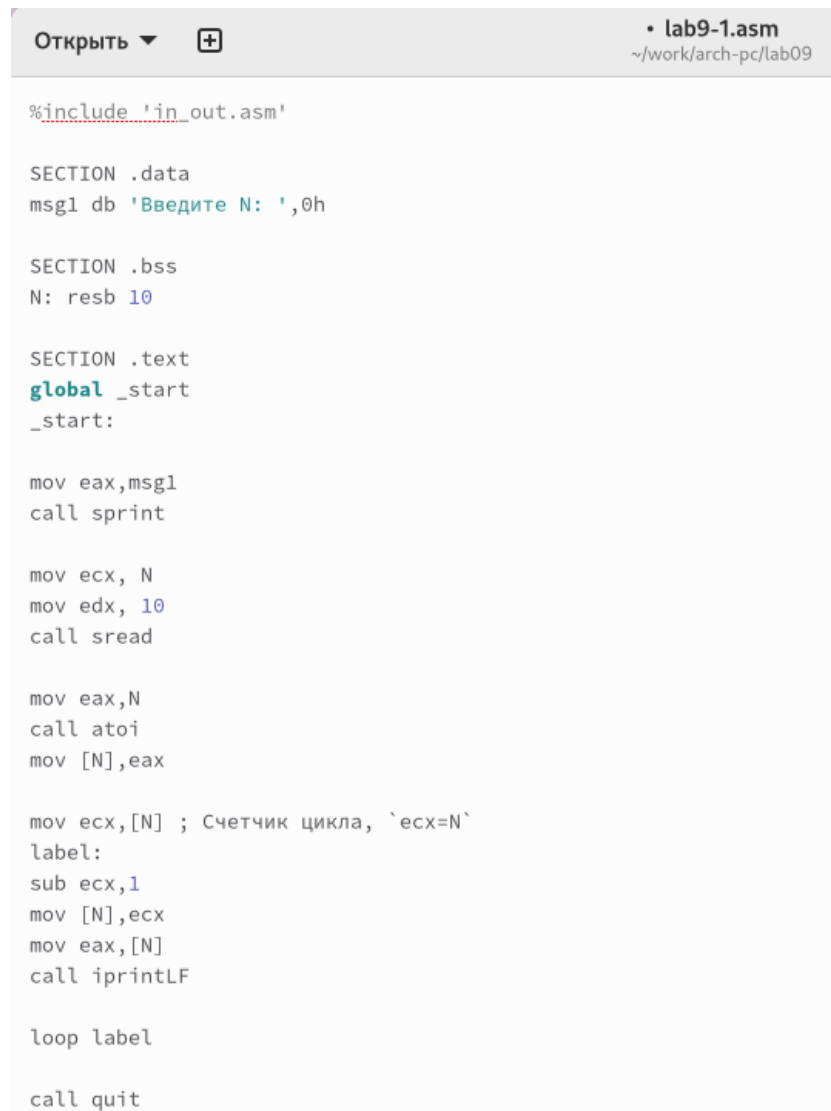
Рис. 4.2: Ввод текста



```
[avobrezkova@fedora:~/work/arch-pc/lab09] nasm -f elf lab9-1.asm  
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o lab9-1 lab9-1.asm  
ld:lab9-1.asm: file format not recognized; treating as linker script  
ld:lab9-1.asm:1: syntax error  
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o lab9-1 lab9-1.o  
[avobrezkova@fedora:~/work/arch-pc/lab09] ./lab9-1  
Введите N: 7  
7  
6  
5  
4  
3  
2  
1  
[avobrezkova@fedora:~/work/arch-pc/lab09]
```

Рис. 4.3: Результат программы

3. Изменила текст программы, добавив изменения значения регистра ecx в цикле. Цикл закольцевался и стал бесконечным. (рис. 4.4; рис. 4.5)




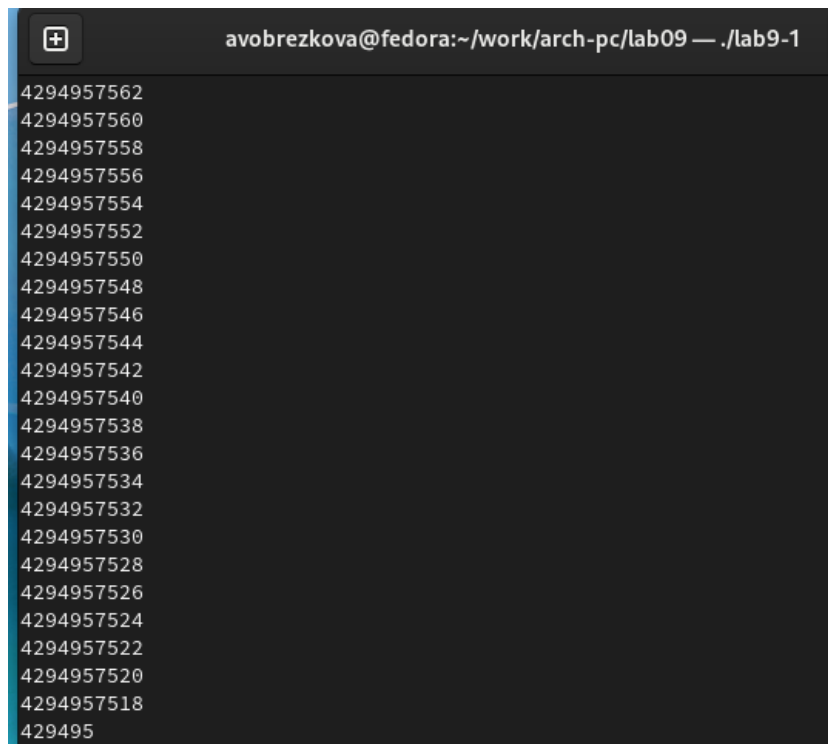
```
Открыть ▼  • lab9-1.asm  
~/work/arch-pc/lab09  
  
%include 'in_out.asm'  
  
SECTION .data  
msg1 db 'Введите N: ',0h  
  
SECTION .bss  
N: resb 10  
  
SECTION .text  
global _start  
_start:  
  
mov eax,msg1  
call sprint  
  
mov ecx, N  
mov edx, 10  
call sread  
  
mov eax,N  
call atoi  
mov [N],eax  
  
mov ecx,[N] ; Счетчик цикла, `ecx=N`  
label:  
sub ecx,1  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
  
loop label  
  
call quit
```


Рис. 4.4: Изменения текста



```
avobrezkova@fedora:~/work/arch-pc/lab09 — ./lab9-1
4294957562
4294957560
4294957558
4294957556
4294957554
4294957552
4294957550
4294957548
4294957546
4294957544
4294957542
4294957540
4294957538
4294957536
4294957534
4294957532
4294957530
4294957528
4294957526
4294957524
4294957522
4294957520
4294957518
429495
```

Рис. 4.5: Результат изменений

4. Изменила текст программы, добавив команды `push` и `pop` (добавление строк и извлечение из стека) для сохранения значения счетчика цикла `loop`. После изменения программы, число проходов циклов стал соответствовать числу введенному с клавиатуры. (рис. 4.6; рис. 4.7)

Открыть ▾ 

• lab9-1.asm  
~/work/arch-pc/lab09

```
%include 'in_out.asm'

SECTION .data
msg1 db 'Введите N: ',0h

SECTION .bss
N: resb 10

SECTION .text
global _start
_start:

mov eax,msg1
call sprint

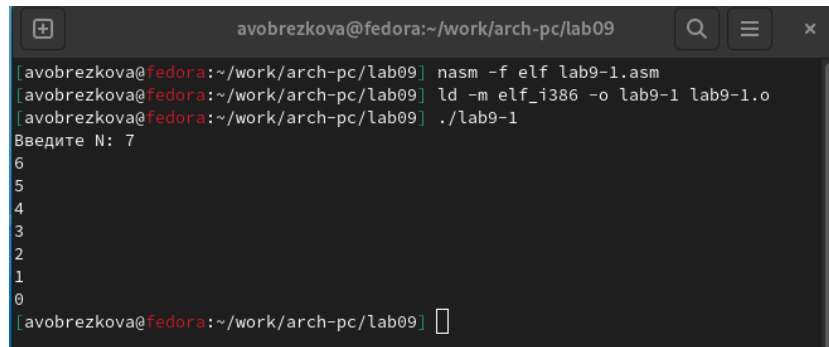
mov ecx, N
mov edx, 10
call sread

mov eax,N
call atoi
mov [N],eax

mov ecx,[N]
label:
push ecx
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
pop ecx
loop label

call quit
```

Рис. 4.6: Изменила программу



```
avobrezkova@fedora:~/work/arch-pc/lab09
[avobrezkova@fedora:~/work/arch-pc/lab09] nasm -f elf lab9-1.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o lab9-1 lab9-1.o
[avobrezkova@fedora:~/work/arch-pc/lab09] ./lab9-1
Введите N: 7
6
5
4
3
2
1
0
[avobrezkova@fedora:~/work/arch-pc/lab09]
```

Рис. 4.7: Вывела результат

5. Создала файл lab9-2.asm в нужном каталоге, ввела нужный текст и вывела результат. Программа выводит все аргументы, введенные при запуске программы. (рис. 4.8; рис. 4.9)



```
Открыть ▾ + lab9-2.asm
~/work/arch-pc/lab09

%include 'in_out.asm'

SECTION .text
global _start

_start:
pop ecx

pop edx

sub ecx, 1

next:
cmp ecx, 0
jz _end

pop eax
call sprintLF
loop next
|
_end:
call quit
```

Рис. 4.8: Ввела нужный текст

```

[avobrezkova@fedora:~/work/arch-pc/lab09] touch lab9-2.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] nasm -f elf lab9-2.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o lab9-2 lab9-2.o
[avobrezkova@fedora:~/work/arch-pc/lab09] ./lab9-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
[avobrezkova@fedora:~/work/arch-pc/lab09]

```

Рис. 4.9: Вывод результата

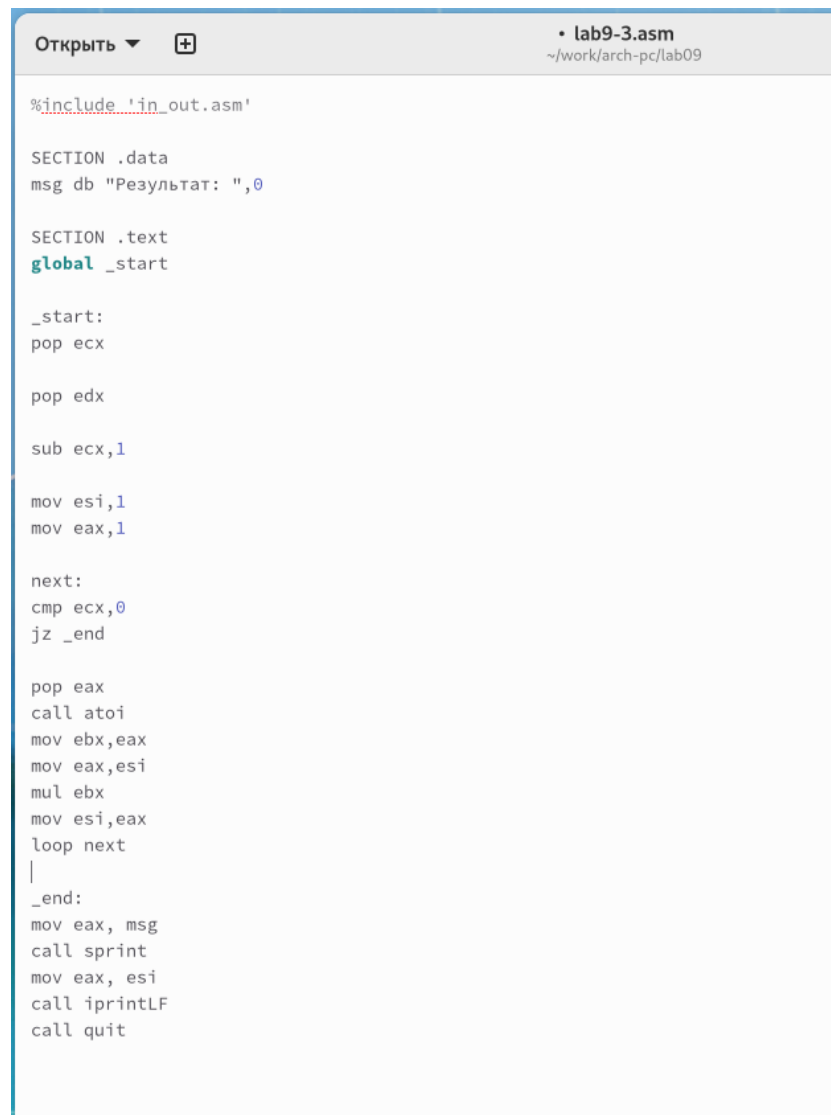
6. Создала файл листинга для программы из файла lab8-2.asm, открыла созданный файл с помощью команды, ознакомилась с его форматом и содержанием. (рис. 4.10; рис. 4.11)

```

[avobrezkova@fedora:~/work/arch-pc/lab09] touch lab9-3.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] nasm -f elf lab9-3.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o lab9-3 lab9-3.o
[avobrezkova@fedora:~/work/arch-pc/lab09] ./lab9-3 12 13 7 10 5
Результат: 47
[avobrezkova@fedora:~/work/arch-pc/lab09]

```

Рис. 4.10: Создание файла



```
%include 'in_out.asm'

SECTION .data
msg db "Результат: ",0

SECTION .text
global _start

_start:
pop ecx

pop edx

sub ecx,1

mov esi,1
mov eax,1

next:
cmp ecx,0
jz _end

pop eax
call atoi
mov ebx,eax
mov eax,esi
mul ebx
mov esi,eax
loop next
|
_end:
mov eax, msg
call sprint
mov eax, esi
call iprintLF
call quit
```

Рис. 4.11: Открытие файла

7. Создала файл lab9-3.asm, ввела в него нужный текст и вывела результат.  
(рис. 4.12; рис. 4.13)



```
Открыть ▾ + • lab9-3.asm
~/work/arch-pc/lab09

%include 'in_out.asm'

SECTION .data
msg db "Результат: ",0

SECTION .text
global _start

_start:
pop ecx

pop edx

sub ecx,1

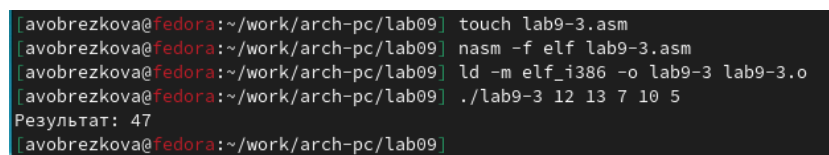
mov esi, 0

next:
cmp ecx,0h
jz _end

pop eax
call atoi
add esi,eax

loop next
_end:
mov eax, msg
call sprint
mov eax, esi
call iprintLF |
call quit
```

Рис. 4.12: Текст программы



```
[avobrezkova@fedora:~/work/arch-pc/lab09] touch lab9-3.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] nasm -f elf lab9-3.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o lab9-3 lab9-3.o
[avobrezkova@fedora:~/work/arch-pc/lab09] ./lab9-3 12 13 7 10 5
Результат: 47
[avobrezkova@fedora:~/work/arch-pc/lab09]
```

Рис. 4.13: Результат программы

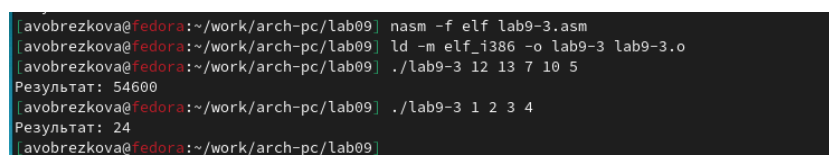
8. Изменила текст программы для вычисления произведения аргументов командной строки и вывела результат. (рис. 4.14; рис. 4.15)





```
Открыть ▾  • lab9-3.asm  
~/work/arch-pc/lab09  
  
%include 'in_out.asm'  
  
SECTION .data  
msg db "Результат: ",0  
  
SECTION .text  
global _start  
  
_start:  
pop ecx  
  
pop edx  
  
sub ecx,1  
  
mov esi,1  
mov eax,1  
  
next:  
cmp ecx,0  
jz _end  
  
pop eax  
call atoi  
mov ebx,eax  
mov eax,esi  
mul ebx  
mov esi,eax  
loop next  
|  
_end:  
mov eax, msg  
call sprint  
mov eax, esi  
call iprintLF  
call quit
```

Рис. 4.14: Текст программы

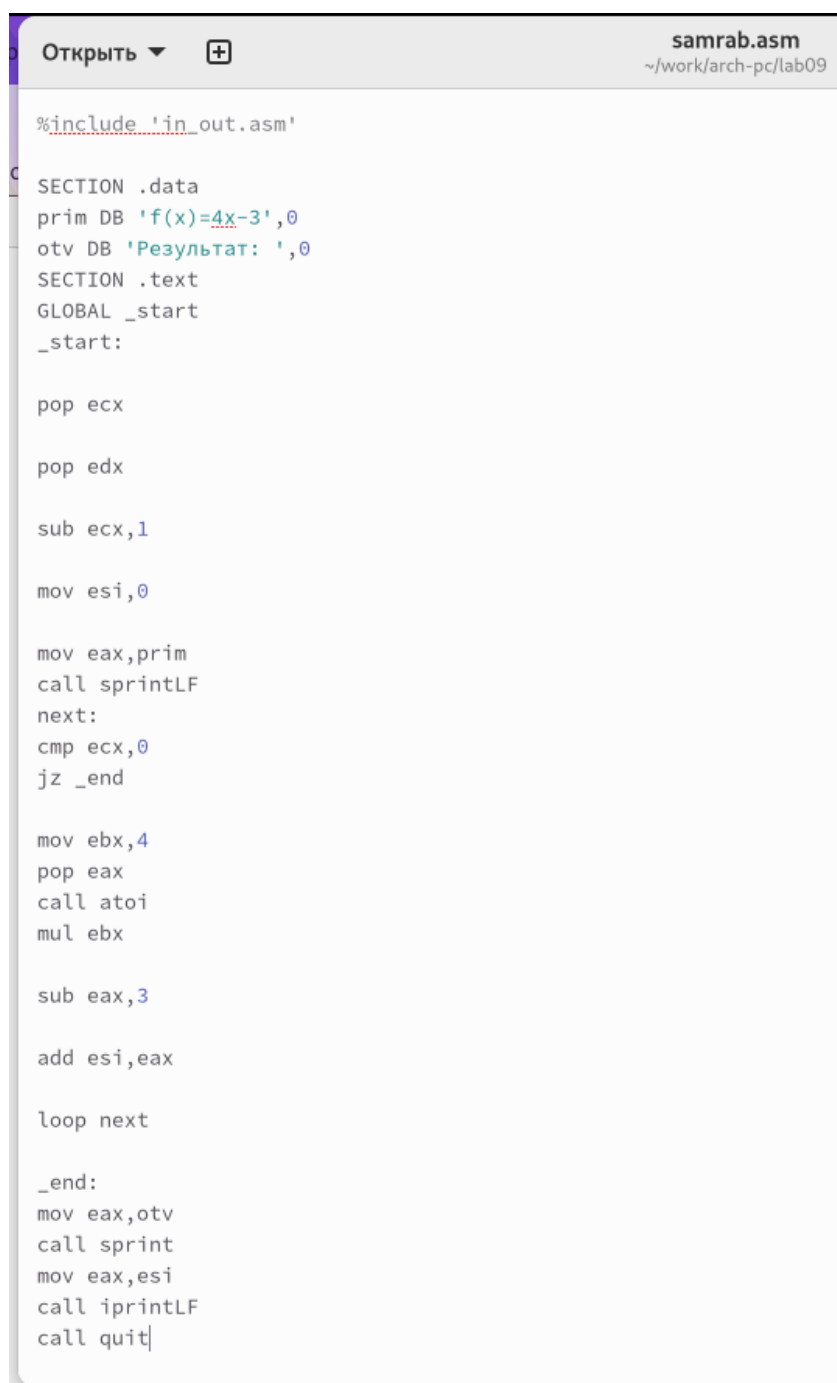


```
[avobrezkova@fedora:~/work/arch-pc/lab09] nasm -f elf lab9-3.asm  
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o lab9-3 lab9-3.o  
[avobrezkova@fedora:~/work/arch-pc/lab09] ./lab9-3 12 13 7 10 5  
Результат: 54600  
[avobrezkova@fedora:~/work/arch-pc/lab09] ./lab9-3 1 2 3 4  
Результат: 24  
[avobrezkova@fedora:~/work/arch-pc/lab09]
```

Рис. 4.15: Результат программы

## 4.1 Задания для самостоятельной работы

1. Я написала программу, которая находит сумму значений функции  $f(x)$  для  $x = x_1, x_2, \dots, x_n$ , т.е. программа должна выводить значение  $f(x_1) + f(x_2) + \dots + f(x_n)$ . Значения  $x_i$  передаются как аргументы. Вариант задания: №6. (рис. 4.14; рис. 4.15)



```
Открыть ▾ [+]samrab.asm  
~/work/arch-pc/lab09  
  
%include 'in_out.asm'  
  
SECTION .data  
prim DB 'f(x)=4x-3',0  
otv DB 'Результат: ',0  
SECTION .text  
GLOBAL _start  
_start:  
  
pop ecx  
  
pop edx  
  
sub ecx,1  
  
mov esi,0  
  
mov eax,prim  
call sprintf  
next:  
cmp ecx,0  
jz _end  
  
mov ebx,4  
pop eax  
call atoi  
mul ebx  
  
sub eax,3  
  
add esi,eax  
  
loop next  
  
_end:  
mov eax,otv  
call sprintf  
mov eax,esi  
call iprintLF  
call quit
```

Рис. 4.16: Текст программы

```

[avobrezkova@fedora:~/work/arch-pc/lab09] nasm -f elf samrab.asm
[avobrezkova@fedora:~/work/arch-pc/lab09] ld -m elf_i386 -o samrab samrab.o
[avobrezkova@fedora:~/work/arch-pc/lab09] ./samrab 1 2 3
f(x)=4x-3
Результат: 15
[avobrezkova@fedora:~/work/arch-pc/lab09] ./samrab 1 2 3 4
f(x)=4x-3
Результат: 28
[avobrezkova@fedora:~/work/arch-pc/lab09] ./samrab 1 2 3 4 5
f(x)=4x-3
Результат: 45
[avobrezkova@fedora:~/work/arch-pc/lab09]

```

Рис. 4.17: Результат

Данные изменения можно проверить по ссылке: [https://github.com/avobrezkova/study\\_2022-2023\\_arch-pc/tree/master/labs/lab09](https://github.com/avobrezkova/study_2022-2023_arch-pc/tree/master/labs/lab09)

## 5 Выводы

Приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.

## Список литературы

1. [https://esystem.rudn.ru/pluginfile.php/1584393/mod\\_resource/content/1/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%969.pdf](https://esystem.rudn.ru/pluginfile.php/1584393/mod_resource/content/1/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%969.pdf)