

Лабораторная работа №8

Отчет

Устинова Виктория Вадимовна

Содержание

| | | |
|----------|---------------------------------------|-----------|
| 1 | Цель работы | 5 |
| 2 | Задание | 6 |
| 3 | Выполнение лабораторной работы | 7 |
| 4 | Выводы | 16 |

Список иллюстраций

| | | |
|------|---|----|
| 3.1 | Переходим в каталог и создаем там файл lab8-1.asm | 7 |
| 3.2 | Заполняем данный файл | 8 |
| 3.3 | Смотрим как работает файл | 8 |
| 3.4 | Редактируем файл | 9 |
| 3.5 | Смотрим как работает файл | 9 |
| 3.6 | Редактируем данный файл | 10 |
| 3.7 | Сверяемся с нужным выводом, все верно | 10 |
| 3.8 | Используем команду touch, заполняем файл как указано в листинге | 11 |
| 3.9 | Смотрим как работает наш файл | 11 |
| 3.10 | Используем команду touch | 11 |
| 3.11 | Заполняем файл | 12 |
| 3.12 | Смотрим как работает наша программа | 12 |
| 3.13 | Изменяем программу | 13 |
| 3.14 | Проверяем работу файла, все корректно | 13 |
| 3.15 | Используем команду touch | 14 |
| 3.16 | Сама программа | 14 |
| 3.17 | Все работает корректно | 15 |

Список таблиц

1 Цель работы

Приобрести навыки написания программ с использованием циклов и обработкой аргументов командой строки.

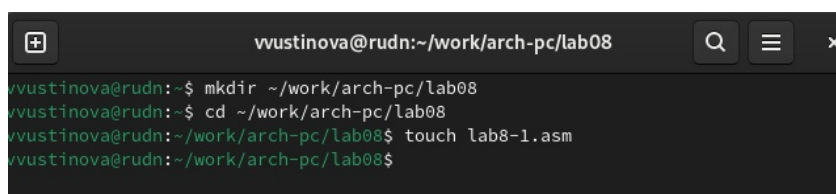
2 Задание

Выполнить лабораторную работу №8 и написать программы с использованием циклов и обработкой аргументов командной строки.

3 Выполнение лабораторной работы

Реализация циклов в NASM

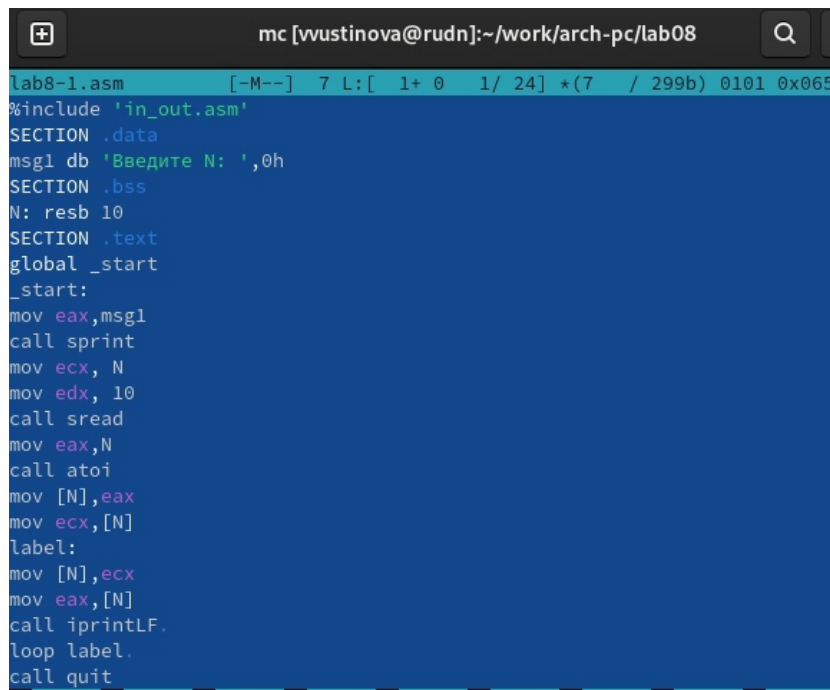
Создаем каталог для 8 лабораторной работы(рис. 3.1).

A terminal window with a dark background. The title bar shows 'vvustinova@rudn:~/work/arch-pc/lab08'. The terminal contains the following commands and their outputs:

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

Рис. 3.1: Переходим в каталог и создаем там файл lab8-1.asm

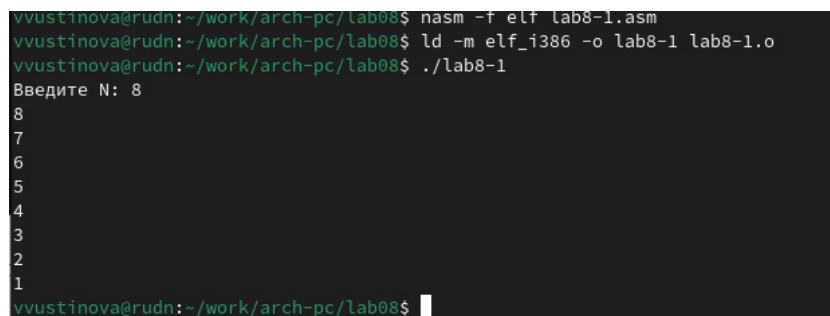
Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 8.1(рис. 3.2).



```
lab8-1.asm [-M--] 7 L: [ 1+ 0 1/ 24] *(7 / 299b) 0101 0x065
%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:
mov [N],ecx
mov eax,[N]
call iprintLF.
loop label.
call quit
```

Рис. 3.2: Заполняем данный файл

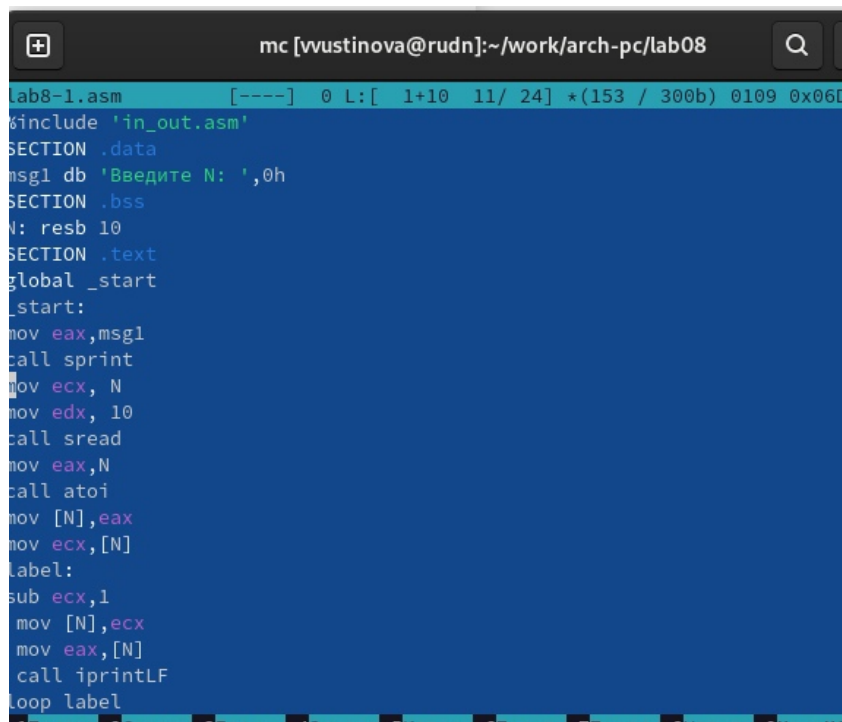
Запускаем файл(рис. 3.3).



```
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 8
8
7
6
5
4
3
2
1
vvustinova@rudn:~/work/arch-pc/lab08$
```

Рис. 3.3: Смотрим как работает файл

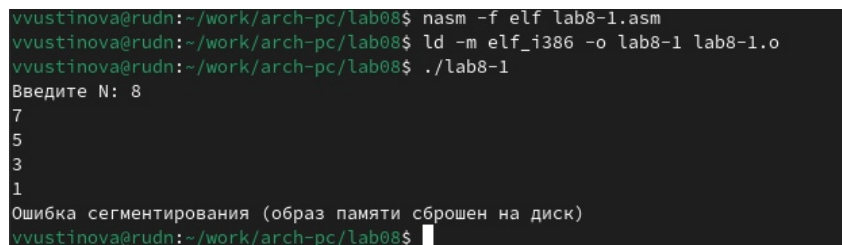
Снова открываем файл для редактирования и изменяем его значения регистра в цикле(рис. 3.4).



```
lab8-1.asm [----] 0 L: [ 1+10 11/ 24] *(153 / 300b) 0109 0x060
#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:
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
loop label
```

Рис. 3.4: Редактируем файл

Запускаем файл(рис. 3.5).



```
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 8
7
5
3
1
Ошибка сегментирования (образ памяти сброшен на диск)
vvustinova@rudn:~/work/arch-pc/lab08$
```

Рис. 3.5: Смотрим как работает файл

Регистр есх принимает значения 7,5,3,1. Число проходов цикла не соответствует числу N, так как уменьшается на 2

Требуется снова отредактировать файл(рис. 3.6).

```
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
```

Рис. 3.6: Редактируем данный файл

Запускаем файл(рис. 3.7).

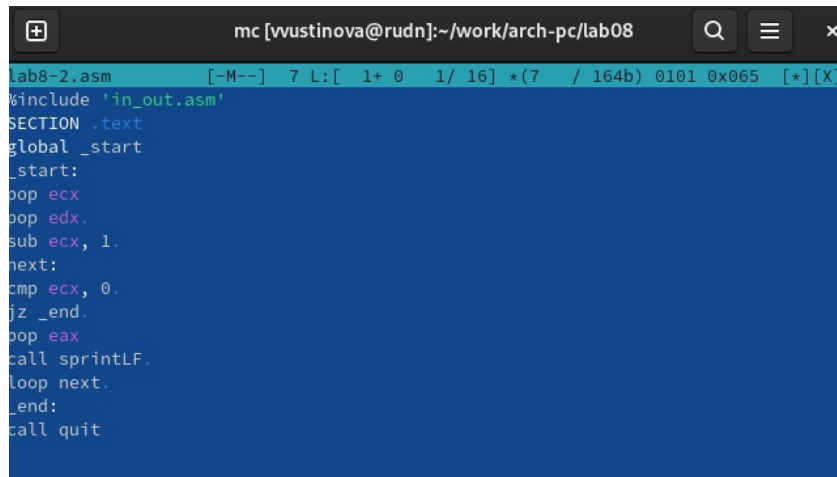
```
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 8
7
6
5
4
3
2
1
0
Ошибка сегментирования (образ памяти сброшен на диск)
vvustinova@rudn:~/work/arch-pc/lab08$
```

Рис. 3.7: Сверяемся с нужным выводом, все верно

В данном случае число проходов цикла равно числу N.

Обработка аргументов командной строки

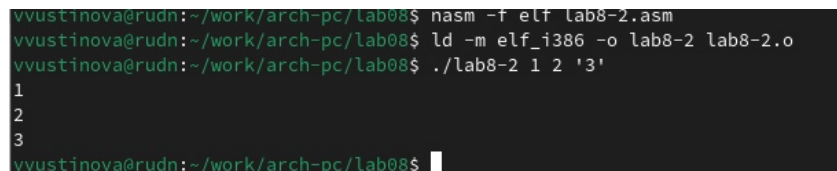
Создаем файл lab8-2.asm и открываем файл в Midnight Commander и заполняем его в соответствии с листингом 8.2(рис. 3.8).

A screenshot of a code editor window titled 'mc [vvustinova@rudn]:~/work/arch-pc/lab08'. The editor shows the contents of 'lab8-2.asm'. The code includes a header line with metadata, followed by an include directive for 'in_out.asm'. It then defines a text section, declares a global symbol '_start', and contains assembly instructions for a loop that reads input, decrements a counter, and prints a line feed. The code ends with a 'quit' instruction.

```
lab8-2.asm      [-M--]  7 L:[  1+ 0   1/ 16] *(7   / 164b) 0101 0x065  [*][X]
%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 sprintf.
    loop next.
_end:
    call quit
```

Рис. 3.8: Используем команду touch, заполняем файл как указано в листинге

Запускаем файл и вводим различные значения(рис. 3.9).

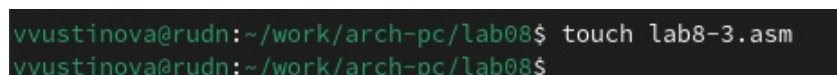
A screenshot of a terminal window showing the compilation and execution of the assembly program. The user runs 'nasm -f elf lab8-2.asm' to create an object file, then 'ld -m elf_i386 -o lab8-2 lab8-2.o' to create an executable. Finally, they run './lab8-2 1 2 '3'', which results in the program printing the numbers 1, 2, and 3 on separate lines.

```
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-2 1 2 '3'
1
2
3
vvustinova@rudn:~/work/arch-pc/lab08$
```

Рис. 3.9: Смотрим как работает наш файл

Программа обработала 3 аргумента.

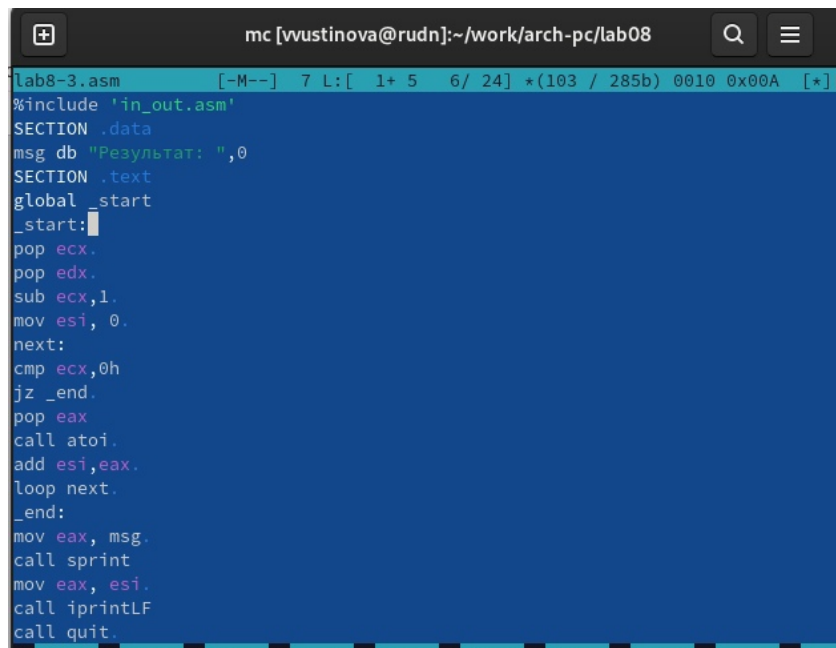
Создаем новый файл lab8-3.asm(рис. 3.10).

A screenshot of a terminal window showing the user running the 'touch' command to create a new file named 'lab8-3.asm' in the current directory.

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

Рис. 3.10: Используем команду touch

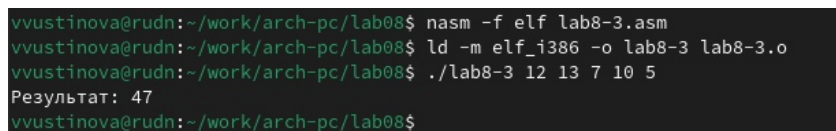
Открываем файл в соответствии с листингом 8.3(рис. 3.11).



```
lab8-3.asm [-M--] 7 L: [ 1+ 5 6/ 24] *(103 / 285b) 0010 0x00A [*]  
%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. 3.11: Заполняем файл

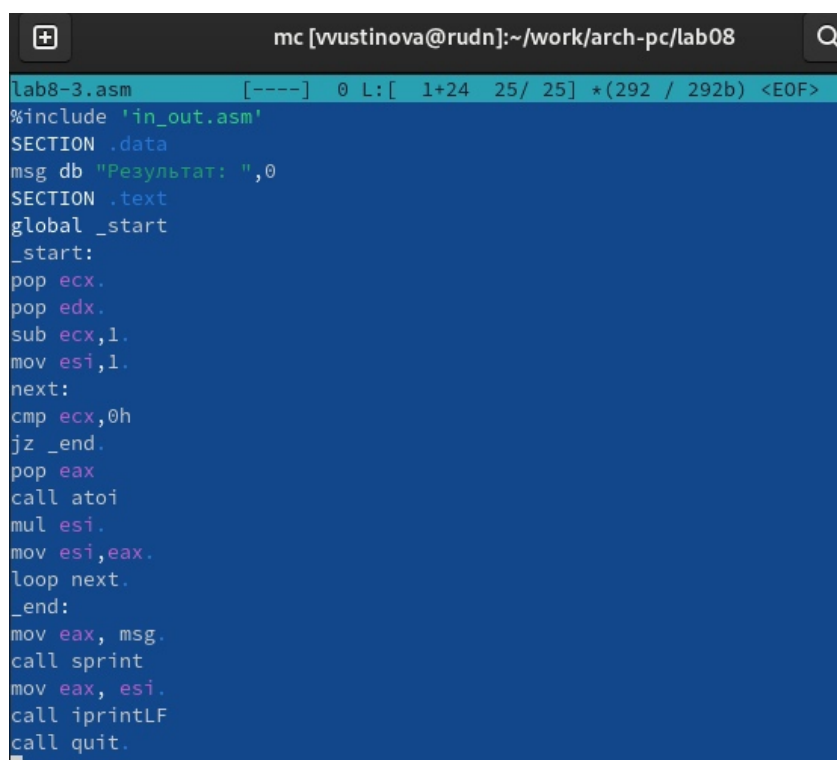
Запускаем файл и вводим различные значения(рис. 3.12).



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

Рис. 3.12: Смотрим как работает наша программа

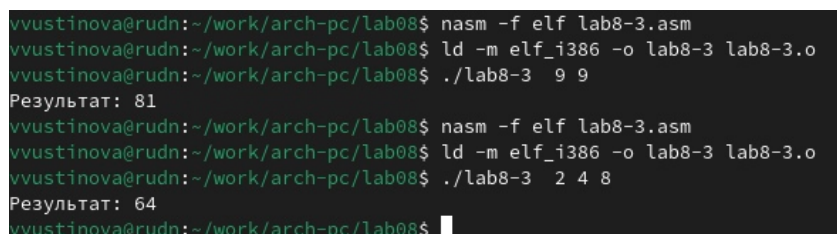
Снова открываем файл для редактирования(рис. 3.13).



```
mc [vvustinova@rudn]:~/work/arch-pc/lab08
lab8-3.asm [----] 0 L: [ 1+24 25/ 25] *(292 / 292b) <EOF>
#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.
next:
cmp ecx,0h
jz _end.
pop eax
call atoi
mul esi.
mov esi,eax.
loop next.
_end:
mov eax, msg.
call sprint
mov eax, esi.
call iprintLF
call quit.
```

Рис. 3.13: Изменяем программу

Запускаем файл, указываем аргументы (рис. 3.14).



```
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-3 9 9
Результат: 81
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-3 2 4 8
Результат: 64
vvustinova@rudn:~/work/arch-pc/lab08$
```

Рис. 3.14: Проверяем работу файла, все корректно

Задания для самостоятельной работы

ВАРИАНТ 12

1. Напишите 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. Вид функции $f(x)$ выбрать из таблицы 8.1 вариантов заданий в соответствии с вариантом, полученным

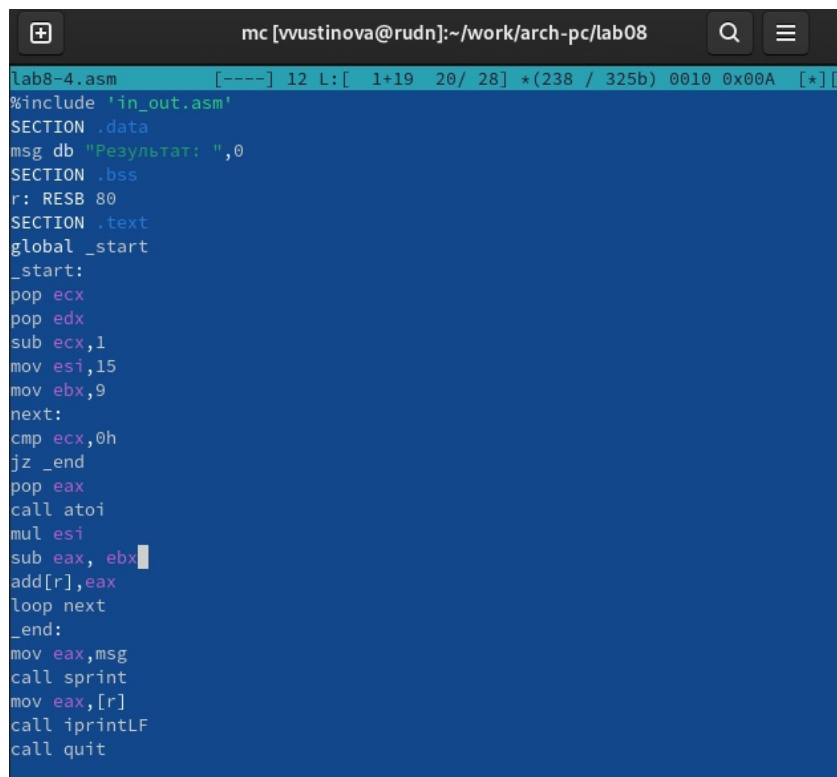
при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу на нескольких наборах $x = x_1, x_2 \dots x_n$.

Создаем новый файл lab8-4.asm(рис. 3.15).

```
vvustinova@rudn:~/work/arch-pc/lab08$ touch lab8-4.asm
vvustinova@rudn:~/work/arch-pc/lab08$
```

Рис. 3.15: Используем команду touch

Открываем его и пишем программу, которая выведет сумму функции: $f(x)=15x-9$ (рис. 3.16).



```
lab8-4.asm [----] 12 L: [ 1+19 20/ 28] *(238 / 325b) 0010 0x00A [*]
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .bss
r: RESB 80
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx,1
mov esi,15
mov ebx,9
next:
cmp ecx,0h
jz _end
pop eax
call atoi
mul esi
sub eax, ebx
add[r],eax
loop next
_end:
mov eax,msg
call sprint
mov eax,[r]
call iprintLF
call quit
```

Рис. 3.16: Сама программа

Запускаем файл и смотрим на его выполнение при разных x (рис. 3.17).

```
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-4 1 2
Результат: 27
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-4 1 2 3
Результат: 63
vvustinova@rudn:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
vvustinova@rudn:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
vvustinova@rudn:~/work/arch-pc/lab08$ ./lab8-4 5 2 3
Результат: 123
```

Рис. 3.17: Все работает корректно

4 Выводы

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