

# **Лабораторная работа номер 8**

**Отчёт**

Виноградова Мария Андреевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
3.1	Реализация циклов в NASM . . . . .	7
3.2	Обработка аргументов командной строки. . . . .	10
3.3	Задание для самостоятельной работы . . . . .	13
<b>4</b>	<b>Выводы</b>	<b>15</b>

# Список иллюстраций

3.1	Создаем каталог с помощью команды mkdir и файл с помощью команды touch . . . . .	7
3.2	Заполняем файл . . . . .	8
3.3	Запускаем файл и проверяем его работу . . . . .	8
3.4	Изменяем файл . . . . .	9
3.5	Запускаем файл и смотрим на его работу . . . . .	9
3.6	Редактируем файл . . . . .	10
3.7	Проверяем, сошелся ли наш вывод с данным в условии выводом .	10
3.8	Создаем файл командой touch . . . . .	11
3.9	Заполняем файл . . . . .	11
3.10	Смотрим на работу программ . . . . .	11
3.11	Создаем файл командой touch . . . . .	12
3.12	Заполняем файл . . . . .	12
3.13	Смотрим на работу программы . . . . .	12
3.14	Изменяем файл . . . . .	13
3.15	Проверяем работу файла(работает правильно) . . . . .	13
3.16	Создаем файл командой touch . . . . .	14
3.17	Пишем программу . . . . .	14
3.18	Смотрим на работу программы при x1=1 x2=2 x3=3 x4=4 (всё верно)	14

## **Список таблиц**

# 1 Цель работы

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

## 2 Задание

Написать программы с использованием циклов и обработкой аргументов командной строки.

## 3 Выполнение лабораторной работы

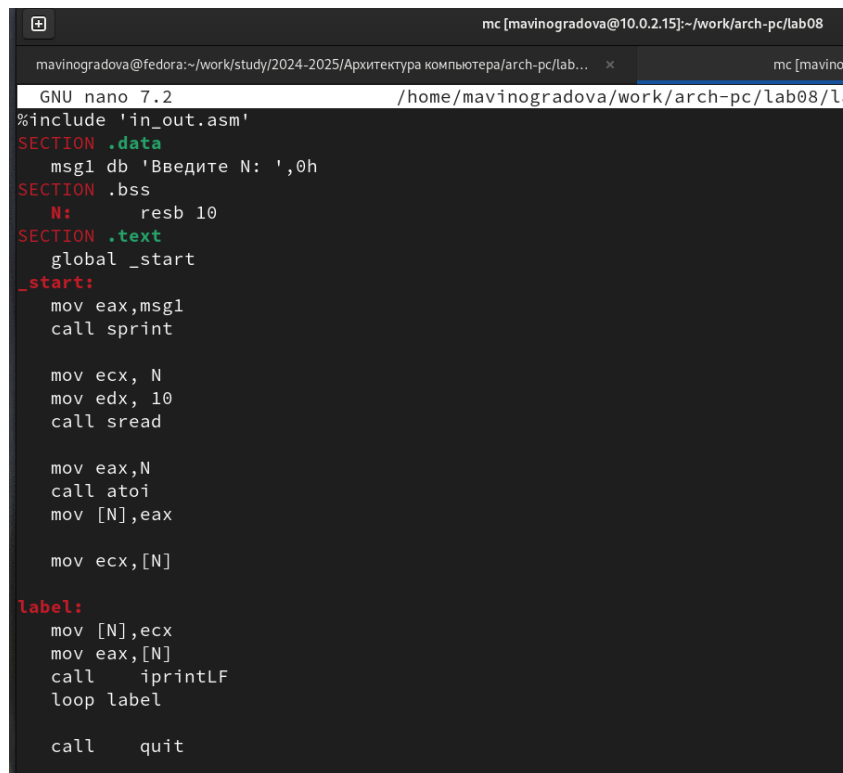
### 3.1 Реализация циклов в NASM

Создаем каталог для программ ЛБ8, и в нем создаем файл (рис. fig. 3.1).

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

Рис. 3.1: Создаем каталог с помощью команды `mkdir` и файл с помощью команды `touch`

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



```
mc [mavinogradova@10.0.2.15]:~/work/arch-pc/lab08
mavinogradova@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab... x mc [mavinogradova@10.0.2.15]:~/work/arch-pc/lab08/l
GNU nano 7.2 /home/mavinogradova/work/arch-pc/lab08/l
%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

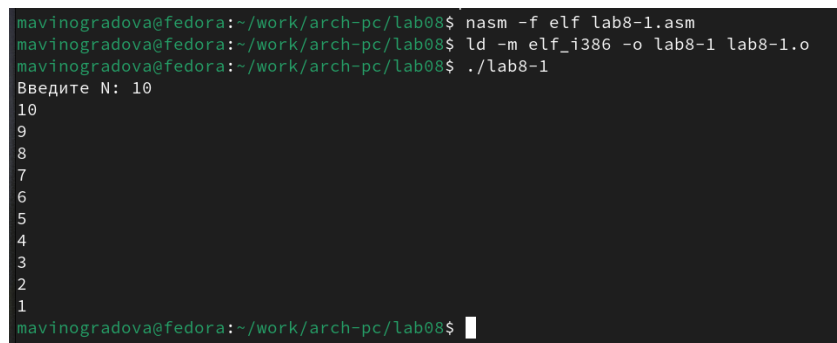
    mov eax,N
    call atoi
    mov [N],eax

    mov ecx,[N]
label:
    mov [N],ecx
    mov eax,[N]
    call iprintLF
    loop label

    call quit
```

Рис. 3.2: Заполняем файл

Создаем исполняемый файл и запускаем его (рис. fig. 3.3).



```
mavinogradova@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
mavinogradova@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
mavinogradova@fedora:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
mavinogradova@fedora:~/work/arch-pc/lab08$
```

Рис. 3.3: Запускаем файл и проверяем его работу

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



```

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

    call    quit

```

Рис. 3.4: Изменяем файл

Создаем исполняемый файл и запускаем его (рис. fig. 3.5).

```

mavinogradova@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
mavinogradova@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
mavinogradova@fedora:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
7
5
3
1
mavinogradova@fedora:~/work/arch-pc/lab08$

```

Рис. 3.5: Запускаем файл и смотрим на его работу

Регистр ecx принимает значения 9,7,5,3,1(на вход подается число 10, в цикле label данный регистр уменьшается на 2 командой sub и loop).

Число проходов цикла не соответствует числу N, так как уменьшается на 2.

Снова открываем файл для редактирования и изменяем его, чтобы все корректно работало (рис. fig. 3.6).

```

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

    call    quit

```

Рис. 3.6: Редактируем файл

Создаем исполняемый файл и запускаем его (рис. fig. 3.7).

```

mavinogradova@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
mavinogradova@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
mavinogradova@fedora:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
mavinogradova@fedora:~/work/arch-pc/lab08$

```

Рис. 3.7: Проверяем, сошелся ли наш вывод с данным в условии выводом

В данном случае число проходов цикла равна числу N.

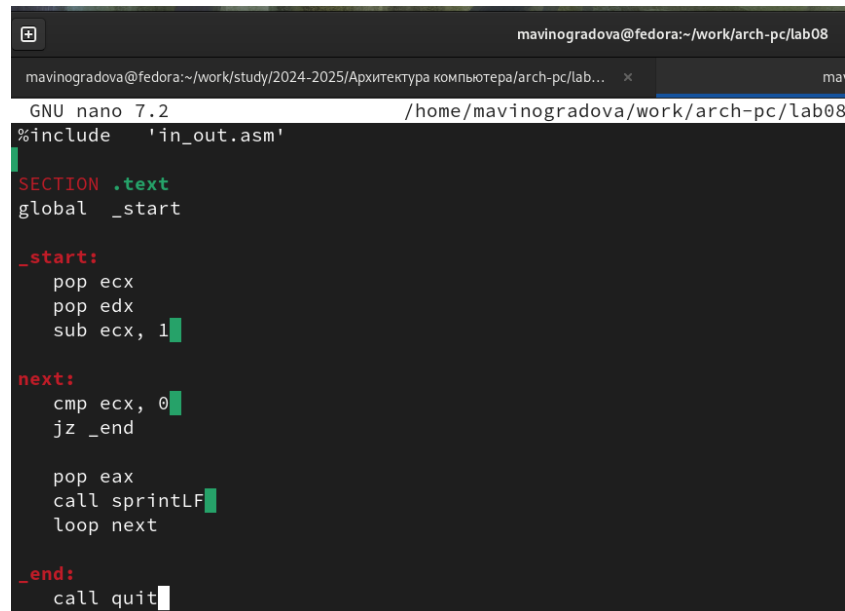
## 3.2 Обработка аргументов командной строки.

Создаем новый файл (рис. fig. 3.8).

```
mavinogradova@fedora:~/work/arch-pc/lab08$ touch lab8-2.asm
mavinogradova@fedora:~/work/arch-pc/lab08$
```

Рис. 3.8: Создаем файл командой touch

Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 8.2 (рис. fig. 3.9).



```
GNU nano 7.2 /home/mavinogradova/work/arch-pc/lab08
%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. 3.9: Заполняем файл

Создаем исполняемый файл и проверяем его работу, указав аргументы (рис. fig. 3.10).

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

Рис. 3.10: Смотрим на работу программ

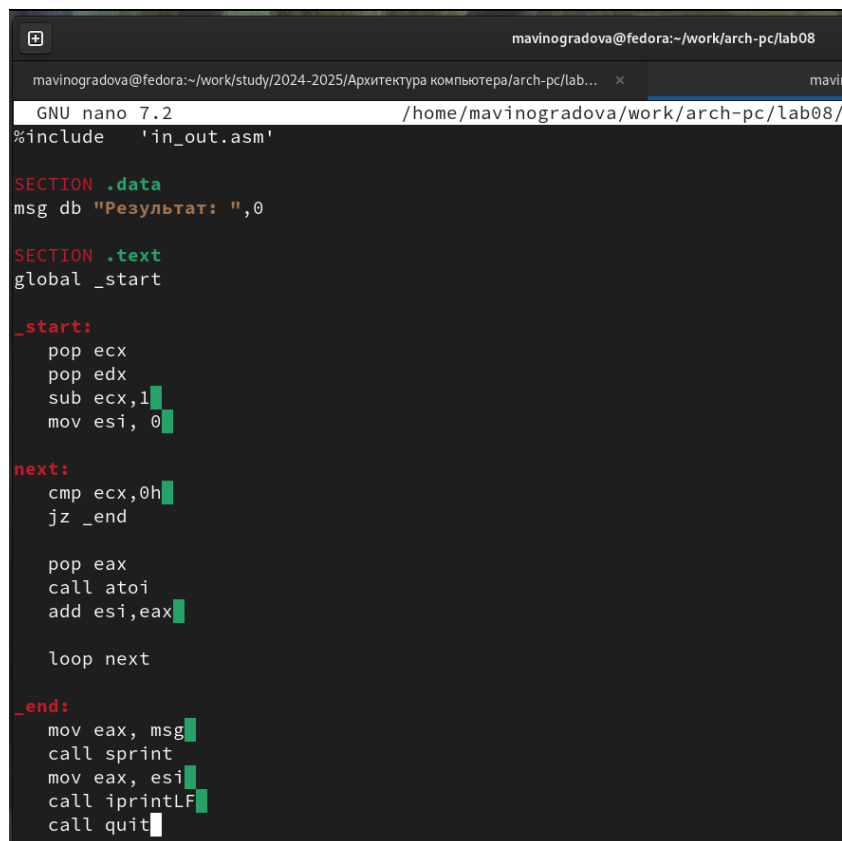
Программой было обработано 3 аргумента.

Создаем новый файл lab8-3.asm (рис. fig. 3.11).

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

Рис. 3.11: Создаем файл командой touch

Открываем файл и заполняем его в соответствии с листингом 8.3 (рис. fig. 3.12).



```
GNU nano 7.2 /home/mavinogradova/work/arch-pc/lab08/
%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. 3.12: Заполняем файл

Создаём исполняемый файл и запускаем его, указав аргументы (рис. fig. 3.13).

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

Рис. 3.13: Смотрим на работу программы

Снова открываем файл для редактирования и изменяем его, чтобы вычислялось произведение вводимых значений (рис. fig. 3.14).

```

next:
    cmp ecx, 0h
    jz _end

    pop eax
    call atoi
    mul esi
    mov esi, eax

    loop next

```

Рис. 3.14: Изменяем файл

Создаём исполняемый файл и запускаем его, указав аргументы (рис. fig. 3.15).

```

mavinogradova@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
mavinogradova@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
mavinogradova@fedora:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 54600
mavinogradova@fedora:~/work/arch-pc/lab08$

```

Рис. 3.15: Проверяем работу файла(работает правильно)

### 3.3 Задание для самостоятельной работы

ВАРИАНТ-12

Создаем новый файл (рис. fig. 3.16).

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

Рис. 3.16: Создаем файл командой touch

Открываем его и пишем программу, которая выведет сумму значений, получившихся после решения выражения  $(15x-9)$  (рис. fig. 3.17).

```
GNU nano 7.2 /home/mavinogradova/work/arch-pc/lab0
%include 'in_out.asm'

SECTION .data
msg db "Результат: ", 0

SECTION .bss
prn resb 80

SECTION .text
global _start

_start:
    pop ecx
    sub ecx, 1
    mov esi, 0

next:
    cmp ecx, 0
    jz _end
```

Рис. 3.17: Пишем программу

Транслируем файл и смотрим на работу программы (рис. fig. ??).

```
mavinogradova@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
mavinogradova@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
mavinogradova@fedora:~/work/arch-pc/lab08$ ./lab8-4 1 2 3 4
Результат: 114
mavinogradova@fedora:~/work/arch-pc/lab08$
```

Рис. 3.18: Смотрим на работу программы при  $x_1=1$   $x_2=2$   $x_3=3$   $x_4=4$  (всё верно)

## 4 Выводы

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