

Отчёт по лабораторной работе №8

Дисциплина: Архитектура компьютера

Студент: Верниковская Екатерина Андреевна

Содержание

Цель работы	5
Задание	6
Выполнение лабораторной работы	7
Реализация циклов в NASM	7
Обработка аргументов командной строки	15
Задание для самостоятельной работы	20
Выводы	23

Список таблиц

Список иллюстраций

1	Создание первого файла	7
2	Копирование файла «in_out.asm»	8
3	Ввод текста программы с инструкцией loop	9
4	Создание исполняемого файла и его запуск	10
5	Изменение программы	11
6	Исполняемый файл + запуск 1	11
7	Исполняемый файл + запуск 2	12
8	Исполняемый файл + запуск 3	12
9	Изменение программы	14
10	Исполняемый файл + запуск	14
11	Создание файла «lab8-2.asm»	15
12	Ввод текста программы	16
13	Создание исполняемого файла и его запуск	16
14	Создание файла «lab8-3.asm»	16
15	Ввод текста программы	18
16	Создание исполняемого файла и его запуск	18
17	Изменение программы	19
18	Создание исполняемого файла и его запуск	20
19	Создание файла «lab8-4.asm»	20
20	Ввод текста программы	22
21	Создание исполняемого файла и его запуск	22

Цель работы

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

Задание

1. Создать каталог для программ лабораторной работы №8 и в нём создать файл «lab8-1.asm».
2. Ввести в файл «lab8-1.asm» определённый текст программы с использованием инструкции `loop`. Создать исполняемый файл и запустить его.
3. Изменить текст программы. Снова создать исполняемый файл и запустить его.
4. Опять изменить текст программы, добавив команды `push` и `pop`, создать исполняемый файл и запустить его.
5. Создать файл «lab8-2.asm» и ввести в него определённый текст программы, которая выводит на экран аргументы командной строки. Создать исполняемый файл и запустить его.
6. Создать файл «lab8-3.asm» и ввести в него определённый текст программы, которая вычисляет сумму аргументов командной строки. Создать исполняемый файл и запустить его.
7. Изменить программу для вычисления произведения аргументов командной строки. Снова создать исполняемый файл и запустить его.
8. Написать программу, которая находит сумму значений функции $f(x)$ для $x = x_1, x_2, \dots, x_n$. Программа должна выводить значение $f(x_1) + f(x_2) + \dots + f(x_n)$. Вид функции брать из определённой таблицы, в соответствии с полученным вариантом (В нашем случае это 17 вариант).
9. Создать исполняемый файл и проверить его работу на нескольких наборах $x = x_1, x_2, \dots, x_n$.

Выполнение лабораторной работы

Реализация циклов в NASM

В созданном каталоге «~/work/arch-pc/lab08» создаём файл «lab8-1.asm» (рис. [-@fig:001])

```
eavernikovskaya@ubuntu-katerok:~$ mkdir ~/work/arch-pc/lab08
eavernikovskaya@ubuntu-katerok:~$ cd ~/work/arch-pc/lab08
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ touch lab8-1.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ls
lab8-1.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

Рис. 1: Создание первого файла

Копируем из каталога «~/work/arch-pc/lab07» файл «in_out.asm» (рис. [-@fig:002])

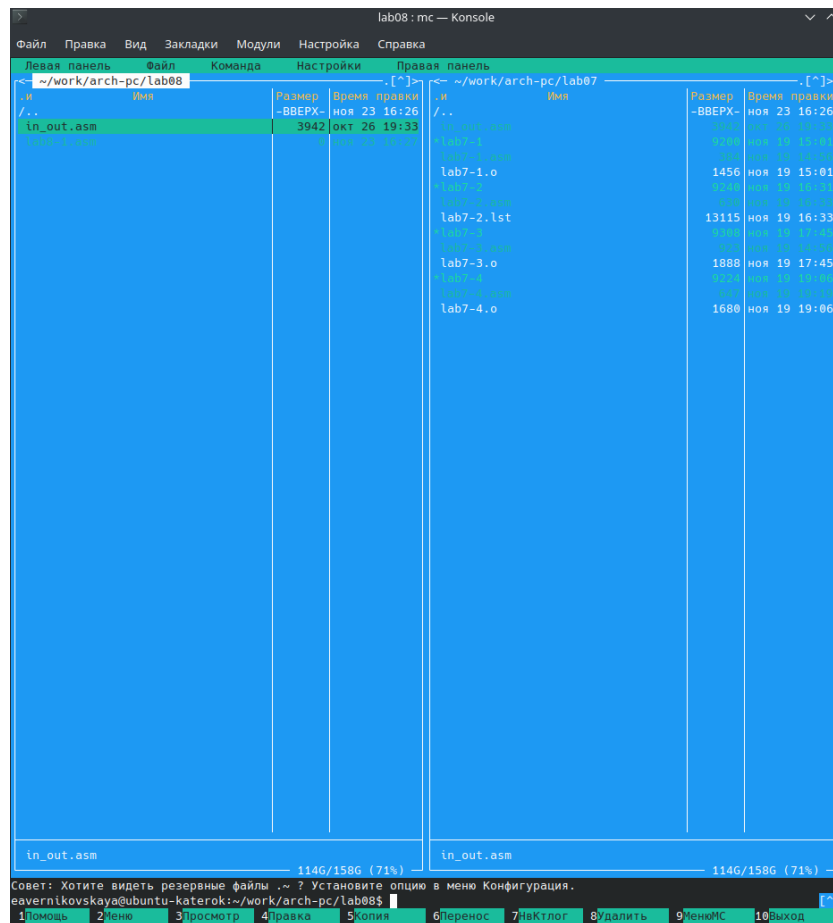


Рис. 2: Копирование файла «in_out.asm»

Вводим нужный текст программы с использованием инструкции loop (рис. [-@fig:003])

Текст программы:

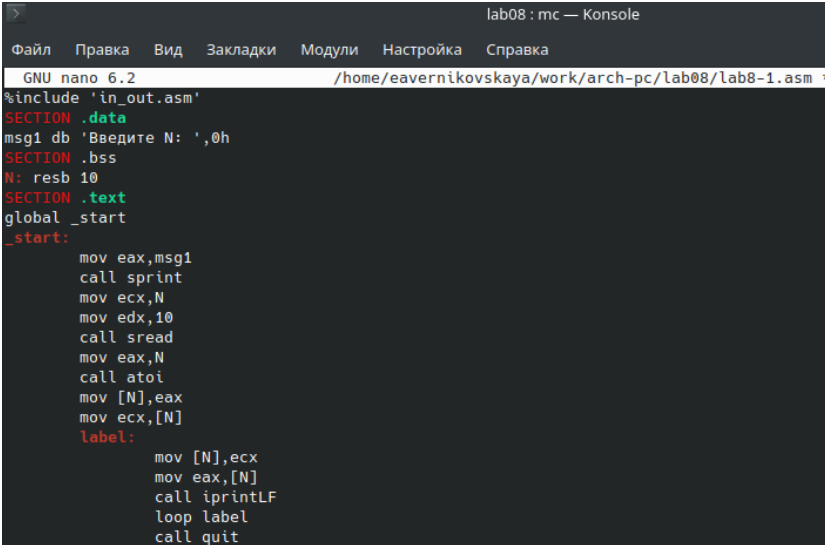
```
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
```



```

_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

```



```

> lab08: mc — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
GNU nano 6.2 /home/eavernikovskaya/work/arch-pc/lab08/lab8-1.asm *
#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: Ввод текста программы с инструкцией loop

Создаём исполняемый файл и запускаем его (рис. [-@fig:004])

```

eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 4
4
3
2
1
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 6
6
5
4
3
2
1
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ █

```

Рис. 4: Создание исполняемого файла и его запуск

Изменяем текст программы, добавив изменение значение регистра `ecx` в цикле `label` (рис. [-@fig:005])

Изменённый текст программы:

```

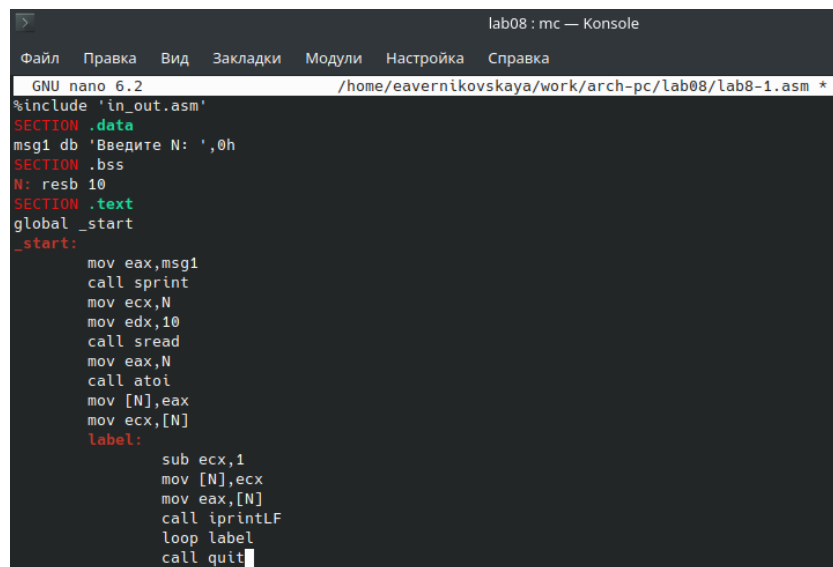
#include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
    mov eax,msg1
    call sprint
    mov ecx,N
    mov edx,10
    call sread
    mov eax,N
    call atoi
    mov [N],eax
    mov ecx,[N]

```

```

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

```



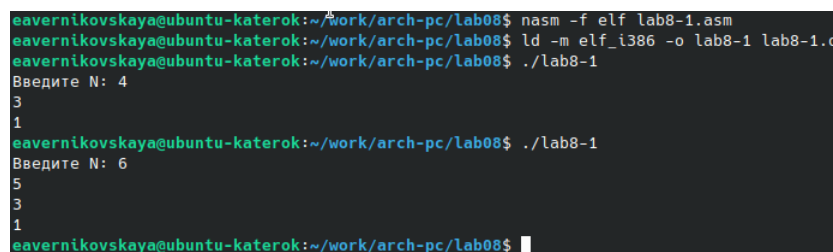
```

lab08 : mc — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
GNU nano 6.2 /home/eavernikovskaya/work/arch-pc/lab08/lab8-1.asm *
#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
    call quit

```

Рис. 5: Изменение программы

Снова создаём исполняемый файл и запускаем его (рис. [-@fig:006]), (рис. [-@fig:007]) и (рис. [-@fig:008])



```

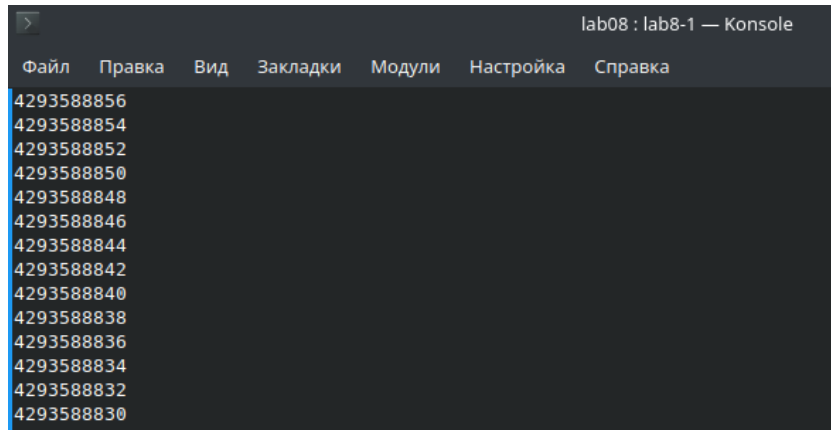
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 4
3
1
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 6
5
3
1
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$

```

Рис. 6: Исполняемый файл + запуск 1

```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 5
```

Рис. 7: Исполняемый файл + запуск 2



The screenshot shows a debugger window titled 'lab08 : lab8-1 — Konsole'. The menu bar includes 'Файл', 'Правка', 'Вид', 'Закладки', 'Модули', 'Настройка', and 'Справка'. The main pane displays a list of memory addresses in hexadecimal, starting from 4293588856 and ending at 4293588830, with a step of 16 bytes. The addresses are: 4293588856, 4293588854, 4293588852, 4293588850, 4293588848, 4293588846, 4293588844, 4293588842, 4293588840, 4293588838, 4293588836, 4293588834, 4293588832, and 4293588830.

Рис. 8: Исполняемый файл + запуск 3

Ответы на вопросы:

1) Какие значения принимает регистр `ecx` в цикле?

- Регистр `ecx` принимает некорректные значения в цикле.

2) Соответствует ли число проходов цикла значению `N` введённому с клавиатуры?

- Нет. Число проходов не соответствует значению `N` введённому с клавиатуры.

Снова изменяем текст программы так, добавив команды `push` и `pop` (рис. [fig:009])

Изменённый текст программы:

```
%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

```

```
lab08 : mc — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
GNU nano 6.2 /home/eavernikovskaya/work/arch-pc/lab08/lab8-1.asm *
#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. 9: Изменение программы

Опять создаём исполняемый файл и запускаем его (рис. [-@fig:010])

```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 4
3
2
1
0
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 6
5
4
3
2
1
0
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

Рис. 10: Исполняемый файл + запуск

Ответы на вопросы:

- 1) Соответствует ли в данном случае число проходов цикла значению N введённому с клавиатуры?
 - Да, теперь соответствует.

Обработка аргументов командной строки

Создаём файл «lab8-2.asm» (рис. [-@fig:011])

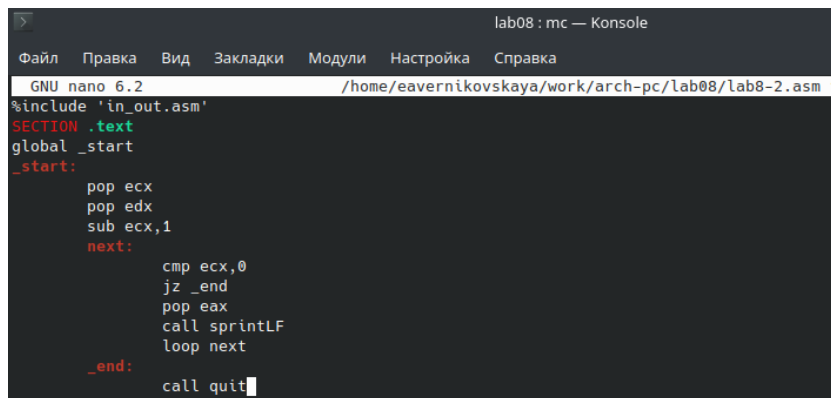
```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ touch lab8-2.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ls
in_out.asm  lab8-1  lab8-1.asm  lab8-1.o  lab8-2.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

Рис. 11: Создание файла «lab8-2.asm»

Вводим текст программы, которая выводит на экран аргументы командной строки (рис. [-@fig:012])

Текст программы:

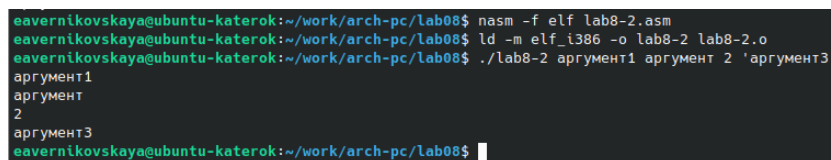
```
%include 'in_out.asm'
SECTION .text
global _start
_start:
    pop ecx
    pop edx
    sub ecx,1
next:
    cmp ecx,0
    jz _end
    pop eax
    call sprintLF
    loop next
_end:
    call quit
```



```
GNU nano 6.2 /home/eavernikovskaya/work/arch-pc/lab08/lab8-2.asm *
#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. 12: Ввод текста программы

Создаём исполняемый файл и запускаем его, указав аргументы (рис. [-@fig:013])



```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент3'
аргумент1
аргумент
2
аргумент3
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

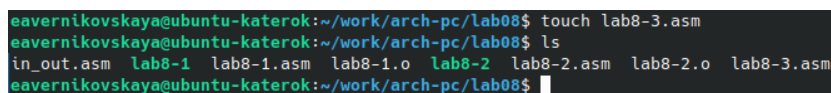
Рис. 13: Создание исполняемого файла и его запуск

Ответы на вопросы:

1) Сколько аргументов было обработано программой?

- Программой было обработано 4 аргумента.

Создаём файл «lab8-3.asm» (рис. [-@fig:014])



```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ touch lab8-3.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1 lab8-1.asm lab8-1.o lab8-2 lab8-2.asm lab8-2.o lab8-3.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

Рис. 14: Создание файла «lab8-3.asm»

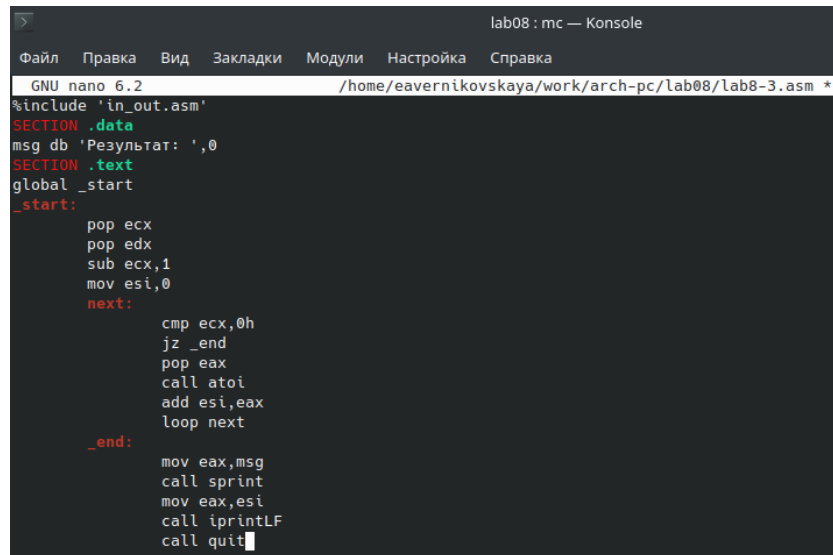
Вводим текст программы, которая вычисляет сумму аргументов командной строки (рис. [-@fig:015])

Текст программы:


```

#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

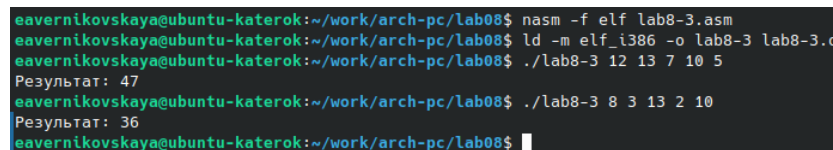
```



```
lab08: mc — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
GNU nano 6.2 /home/eavernikovskaya/work/arch-pc/lab08/lab8-3.asm *
#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. 15: Ввод текста программы

Создаём исполняемый файл и запускаем его, указав аргументы (рис. [-@fig:016])



```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-3 8 3 13 2 10
Результат: 36
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

Рис. 16: Создание исполняемого файла и его запуск

Изменяем текст программы так, чтобы она вычисляла произведение аргументов командной строки (рис. [-@fig:017])

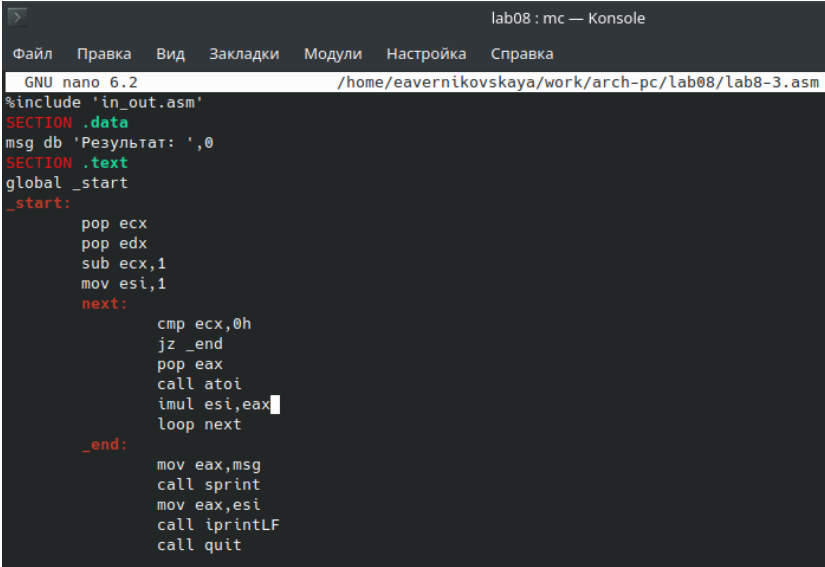
Изменённый текст программы:

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

```



```

lab08 : mc — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
GNU nano 6.2 /home/eavernikovskaya/work/arch-pc/lab08/lab8-3.asm
#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
    imul esi,eax
    loop next
_end:
    mov eax,msg
    call sprint
    mov eax,esi
    call iprintLF
    call quit

```

Рис. 17: Изменение программы

Создаём исполняемый файл и запускаем его, указав аргументы (рис. [-@fig:018])

```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-3 8 3 13 2 10
Результат: 6240
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 54600
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

Рис. 18: Создание исполняемого файла и его запуск

Задание для самостоятельной работы

Создаём файл «lab8-4.asm» (рис. [-@fig:019])

```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ touch lab8-4.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1 lab8-1.asm lab8-1.o lab8-2 lab8-2.asm lab8-2.o lab8-3 lab8-3.asm lab8-3.o lab8-4.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

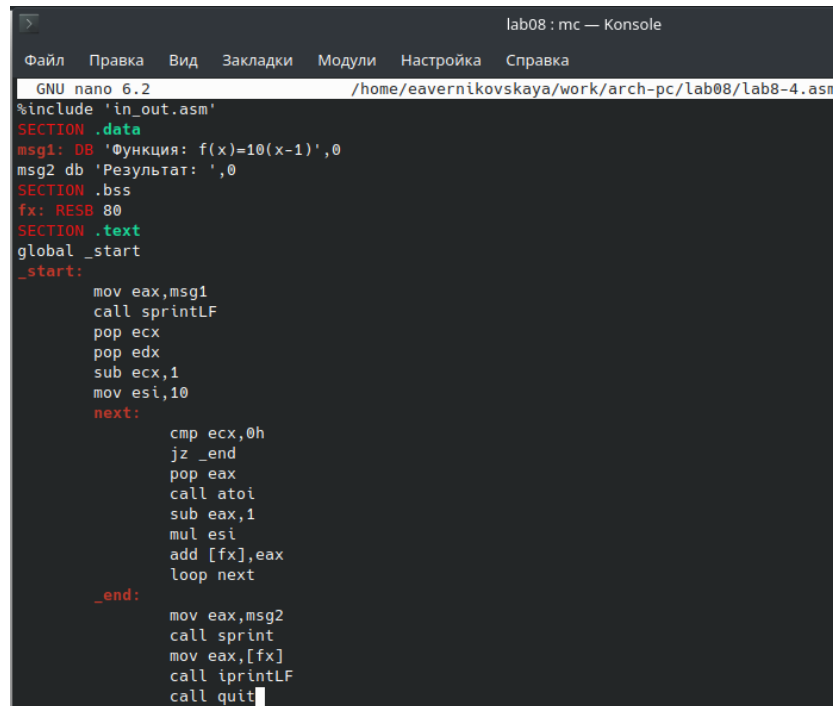
Рис. 19: Создание файла «lab8-4.asm»

Вводим текст программы. Программа находит сумму значений функции $f(x)=10(x-1)$ для разных x . Программа выводит значение $f(x_1) + f(x_2) + \dots + f(x_n)$ (рис. [-@fig:020])

Текст программы:

```
%include 'in_out.asm'
SECTION .data
msg1: DB 'Функция: f(x)=10(x-1)',0
msg2 db 'Результат: ',0
SECTION .bss
fx: RESB 80
SECTION .text
global _start
_start:
    mov eax,msg1
    call sprintLF
```

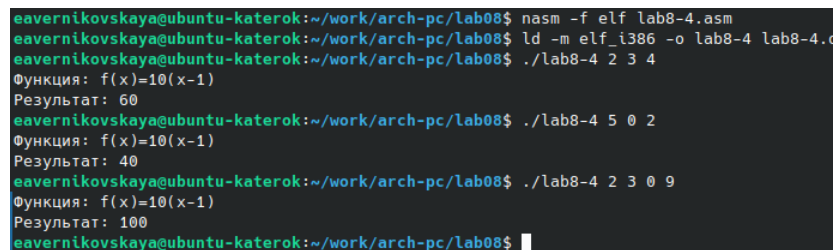
```
pop ecx
pop edx
sub ecx,1
mov esi,10
next:
    cmp ecx,0h
    jz _end
    pop eax
    call atoi
    sub eax,1
    mul esi
    add [fx],eax
    loop next
_end:
    mov eax,msg2
    call sprint
    mov eax,[fx]
    call iprintLF
    call quit
```



```
lab08: mc — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
GNU nano 6.2 /home/eavernikovskaya/work/arch-pc/lab08/lab8-4.asm
#include 'in_out.asm'
SECTION .data
msg1: DB 'Функция: f(x)=10(x-1)',0
msg2 db 'Результат: ',0
SECTION .bss
fx: RESB 80
SECTION .text
global _start
_start:
    mov eax,msg1
    call sprintLF
    pop ecx
    pop edx
    sub ecx,1
    mov esi,10
next:
    cmp ecx,0h
    jz _end
    pop eax
    call atoi
    sub eax,1
    mul esi
    add [fx],eax
    loop next
_end:
    mov eax,msg2
    call sprint
    mov eax,[fx]
    call iprintLF
    call quit
```

Рис. 20: Ввод текста программы

Создаём исполняемый файл и запускаем его, указав аргументы - разные значения x (рис. [-@fig:021])



```
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-4 2 3 4
Функция: f(x)=10(x-1)
Результат: 60
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-4 5 0 2
Функция: f(x)=10(x-1)
Результат: 40
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$ ./lab8-4 2 3 0 9
Функция: f(x)=10(x-1)
Результат: 100
eavernikovskaya@ubuntu-katerok:~/work/arch-pc/lab08$
```

Рис. 21: Создание исполняемого файла и его запуск

Выводы

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