

Отчет по лабораторной работе №8

Дисциплина: архитектура компьютера

Ермакова Анастасия Алексеевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Реализация циклов в NASM	8
4.2	Обработка аргументов командной строки	12
4.3	Выполнение заданий для самостоятельной работы	15
5	Выводы	19
6	Список литературы	20

Список иллюстраций

4.1	Создание каталога и файла	8
4.2	Текст программы файла lsb8-1.asm	9
4.3	Результат работы программы	9
4.4	Измененный текст файла	10
4.5	Результат работы программы	10
4.6	Измененный текст программы	11
4.7	Результат работы программы	11
4.8	Создание файла lab8-2.asm	12
4.9	Текст программы файла lab8-2.asm	12
4.10	Результат работы программы	12
4.11	Текст программы файла lab8-3.asm	13
4.12	Результат работы программы	13
4.13	Измененный текст программы	14
4.14	Результат работы программы	14
4.15	Создание файла lab8-4.asm	15
4.16	Текст программы файла	16
4.17	Результат работы файла	18

Список таблиц

1 Цель работы

Цель данной лабораторной работы - приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

2 Задание

1. Реализация циклов в NASM
2. Обработка аргументов командной строки
3. Выполнение заданий для самостоятельной работы

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. Стек имеет вершину, адрес последнего добавленного элемента, который хранится в регистре esp (указатель стека). Противоположный конец стека называется дном. Значение, помещённое в стек последним, извлекается первым. При помещении значения в стек указатель стека уменьшается, а при извлечении — увеличивается. Для стека существует две основные операции:

- добавление элемента в вершину стека (push);
- извлечение элемента из вершины стека (pop).

4 Выполнение лабораторной работы

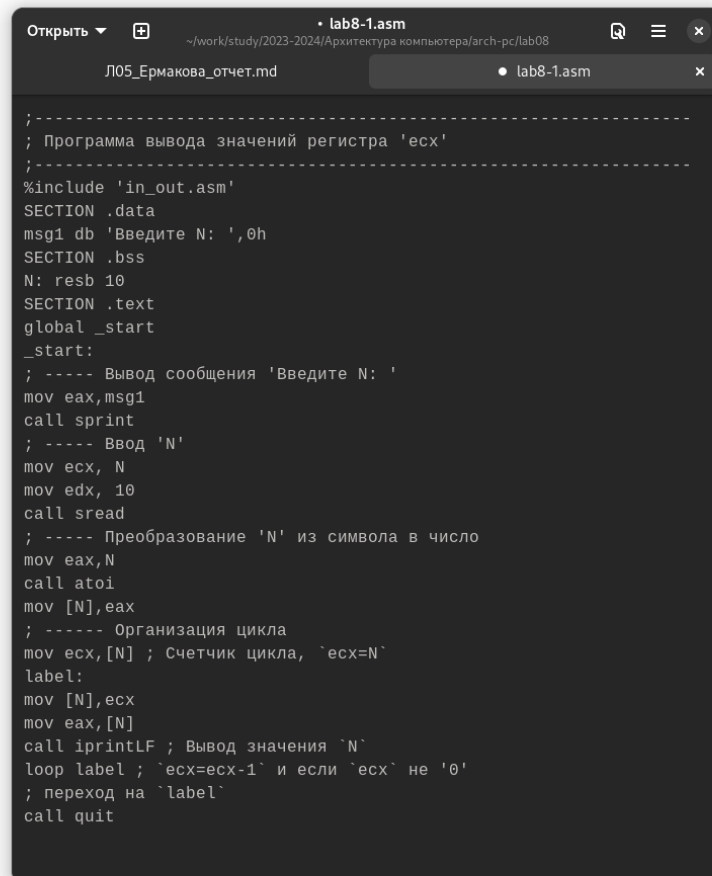
4.1 Реализация циклов в NASM

Создаю каталог для программ лабораторной работы №8, перехожу в него и создаю файл lab8-1.asm. (рис. 4.1).

```
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc$ mkdir lab08
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc$ cd lab08
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ touch lab8-1.asm
```

Рис. 4.1: Создание каталога и файла

Ввожу в файл текст программы из листинга 8.1. (рис. 4.2).

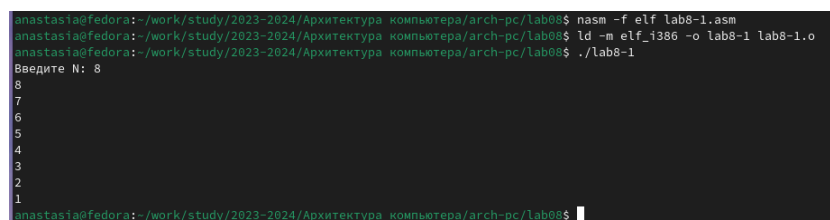


```
lab8-1.asm
~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08
Л05_Ермакова_отчет.md lab8-1.asm

;-----
; Программа вывода значений регистра 'ecx'
;-----
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ---- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ---- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ---- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, 'ecx=N'
label:
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit
```

Рис. 4.2: Текст программы файла lab8-1.asm

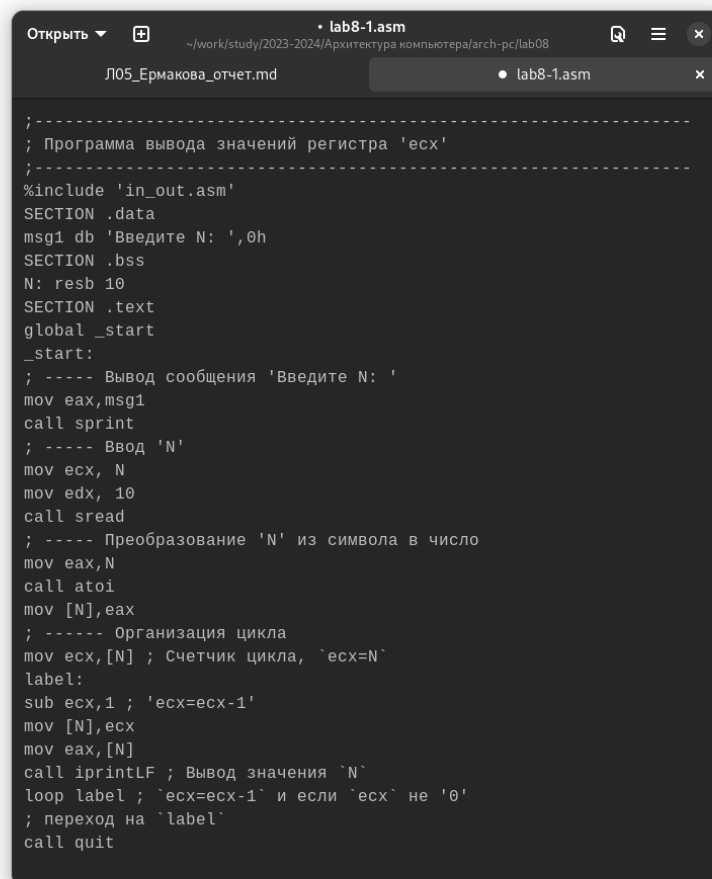
Создаю исполняемый файл и запускаю его. (рис. 4.3). Результат программы - работа циклов в NASM.



```
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ nasm -f elf lab8-1.asm
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ./lab8-1
Введите N: 8
8
7
6
5
4
3
2
1
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$
```

Рис. 4.3: Результат работы программы

Изменяю текст программы, добавив изменение в значении регистра ecx. (рис. 4.4).

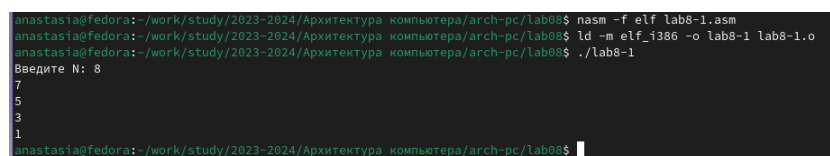


```
lab8-1.asm
~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08
Л05_Ермакова_отчет.md lab8-1.asm

;-----
; Программа вывода значений регистра 'ecx'
;-----
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ---- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ---- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ---- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, `ecx`=N
label:
sub ecx,1 ; `ecx`=ecx-1
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения `N`
loop label ; `ecx`=ecx-1` и если `ecx` не `0`
; переход на `label`
call quit
```

Рис. 4.4: Измененный текст файла

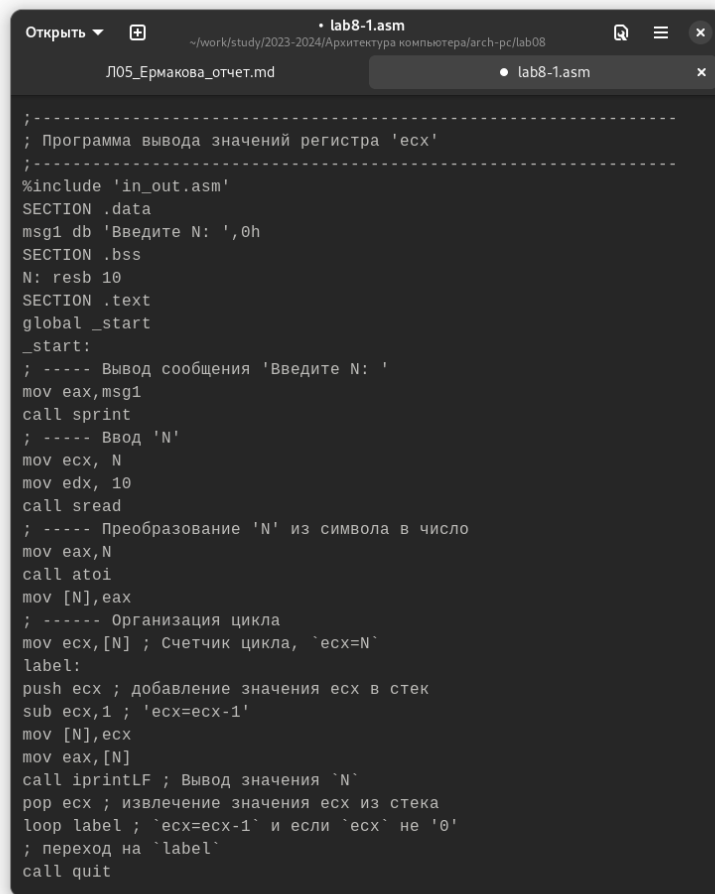
Создаю исполняемый файл и запускаю его. (рис. 4.5). Регистр `ecx` каждый раз уменьшается на 2, количество итераций уменьшается вдвое.



```
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ nasm -f elf lab8-1.asm
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ./lab8-1
Введите N: 8
7
5
3
1
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$
```

Рис. 4.5: Результат работы программы

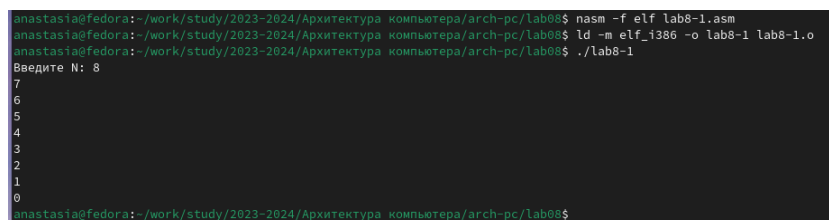
Вношу изменения в текст программы, добавив команды `push` и `pop` для сохранения значения счетчика цикла `loop`. (рис. 4.6).



```
-----  
; Программа вывода значений регистра 'ecx'  
-----  
%include 'in_out.asm'  
SECTION .data  
msg1 db 'Введите N: ',0h  
SECTION .bss  
N: resb 10  
SECTION .text  
global _start  
_start:  
; ----- Вывод сообщения 'Введите N: '  
mov eax,msg1  
call sprint  
; ----- Ввод 'N'  
mov ecx, N  
mov edx, 10  
call sread  
; ----- Преобразование 'N' из символа в число  
mov eax,N  
call atoi  
mov [N],eax  
; ----- Организация цикла  
mov ecx,[N] ; Счетчик цикла, 'ecx=N'  
label:  
push ecx ; добавление значения ecx в стек  
sub ecx,1 ; 'ecx=ecx-1'  
mov [N],ecx  
mov eax,[N]  
call iprintLF ; Вывод значения 'N'  
pop ecx ; извлечение значения ecx из стека  
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'  
; переход на 'label'  
call quit
```

Рис. 4.6: Измененный текст программы

Создаю исполняемый файл и запускаю его. (рис. 4.7). Теперь количество подходов цикла совпадает с введенным значением, но из-за смещения на 1 вывод начался со значения N-1 и заканчивается нулем.



```
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ nasm -f elf lab8-1.asm  
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ ld -m elf_i386 -o lab8-1 lab8-1.o  
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ ./lab8-1  
Введите N: 8  
7  
6  
5  
4  
3  
2  
1  
0  
anastasia@fedora:~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$
```

Рис. 4.7: Результат работы программы

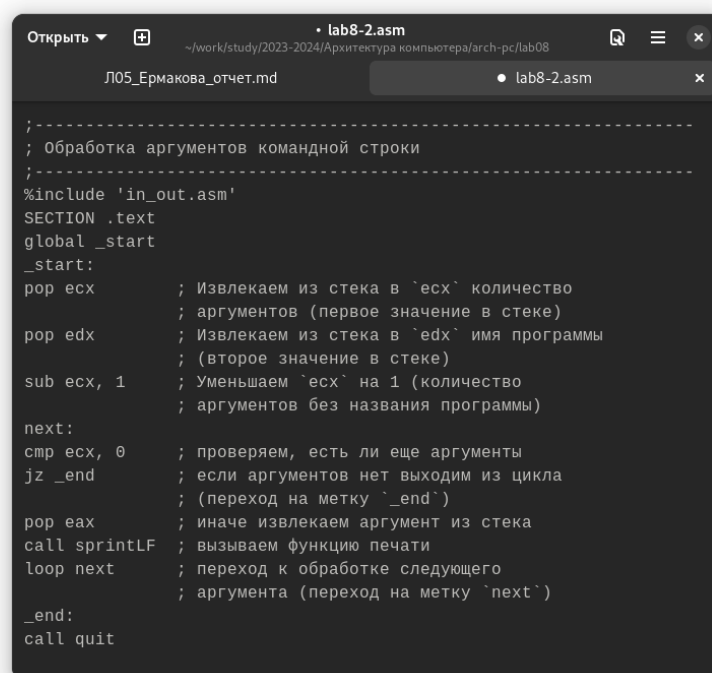
4.2 Обработка аргументов командной строки

Создаю файл lab8-2.asm в текущем каталоге. (рис. 4.8).

```
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ touch lab8-2.asm
```

Рис. 4.8: Создание файла lab8-2.asm

Ввожу текст программы из листинга 8.2. (рис. 4.9).



```
Открыть ▾ + lab8-2.asm
~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08
Л05_Ермакова_отчет.md lab8-2.asm x

;-----
; Обработка аргументов командной строки
;-----
%include 'in_out.asm'
SECTION .text
global _start
_start:
pop ecx      ; Извлекаем из стека в `ecx` количество
              ; аргументов (первое значение в стеке)
pop edx      ; Извлекаем из стека в `edx` имя программы
              ; (второе значение в стеке)
sub ecx, 1    ; Уменьшаем `ecx` на 1 (количество
              ; аргументов без названия программы)

next:
cmp ecx, 0    ; проверяем, есть ли еще аргументы
jz _end       ; если аргументов нет выходим из цикла
              ; (переход на метку `_end`)
pop eax       ; иначе извлекаем аргумент из стека
call printf   ; вызываем функцию печати
loop next     ; переход к обработке следующего
              ; аргумента (переход на метку `next`)
_end:
call quit
```

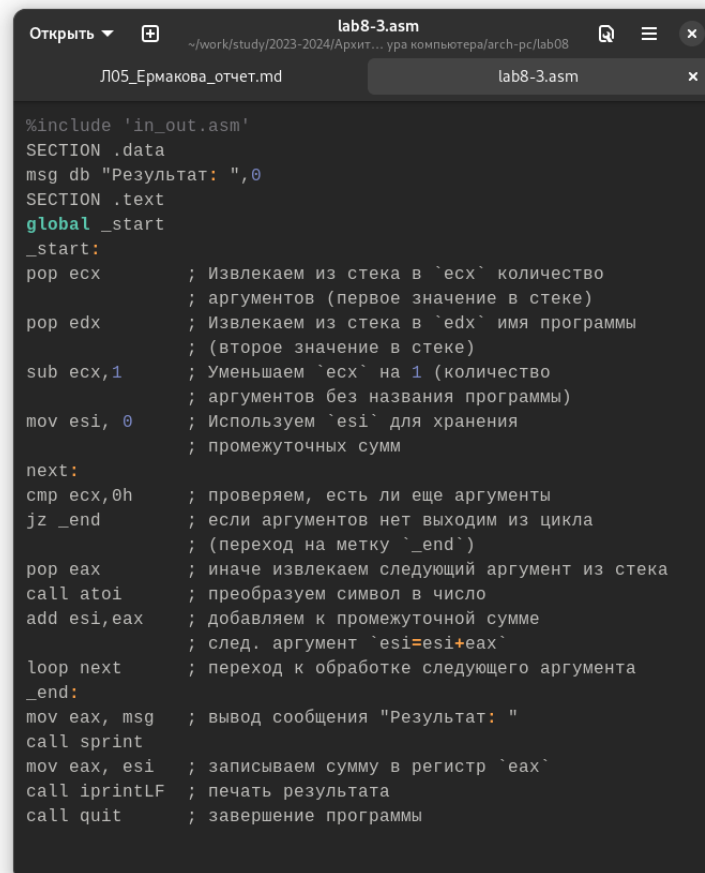
Рис. 4.9: Текст программы файла lab8-2.asm

Создаю исполняемый файл и запускаю его, указав аргументы. (рис. 4.10). Все три аргумента были обработаны программой

```
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ nasm -f elf lab8-2.asm
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$
```

Рис. 4.10: Результат работы программы

Создаю файл lab8-3.asm и ввожу текст программы из листинга 8.3. (рис. 4.11).



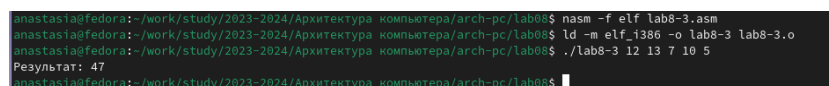
```
lab8-3.asm
~/work/study/2023-2024/Архит... ура компьютера/arch-pc/lab08
Л05_Ермакова_отчет.md lab8-3.asm

%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx      ; Извлекаем из стека в `ecx` количество
              ; аргументов (первое значение в стеке)
pop edx      ; Извлекаем из стека в `edx` имя программы
              ; (второе значение в стеке)
sub ecx,1    ; Уменьшаем `ecx` на 1 (количество
              ; аргументов без названия программы)
mov esi, 0    ; Используем `esi` для хранения
              ; промежуточных сумм

next:
cmp ecx,0h   ; проверяем, есть ли еще аргументы
jz _end      ; если аргументов нет выходим из цикла
              ; (переход на метку `_end`)
pop eax      ; иначе извлекаем следующий аргумент из стека
call atoi    ; преобразуем символ в число
add esi,eax  ; добавляем к промежуточной сумме
              ; след. аргумент `esi=esi+eax`
loop next    ; переход к обработке следующего аргумента
_end:
mov eax,msg  ; вывод сообщения "Результат: "
call sprint
mov eax,esi  ; записываем сумму в регистр `eax`
call iprintLF ; печать результата
call quit    ; завершение программы
```

Рис. 4.11: Текст программы файла lab8-3.asm

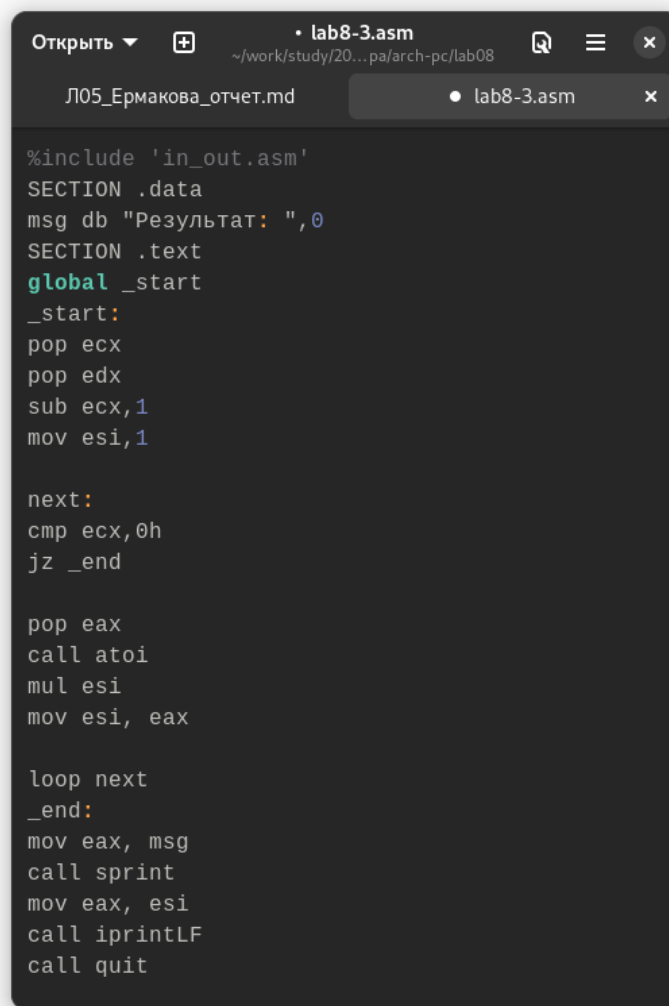
Создаю исполняемый файл и запускаю его, указав аргументы. (рис. 4.12). Программа выводит результат - сумма введенных аргументов.



```
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ nasm -f elf lab8-3.asm
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$
```

Рис. 4.12: Результат работы программы

Изменяю текст программы так, чтобы она считала произведение введенных аргументов. (рис. 4.13).



```
Открыть ▾ + lab8-3.asm
~/work/study/20...pa/arch-pc/lab08
Л05_Ермакова_отчет.md lab8-3.asm x

%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

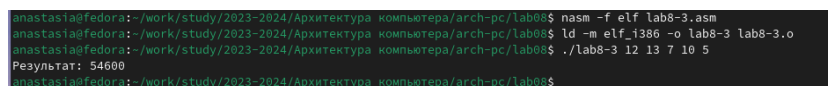
next:
cmp ecx,0h
jz _end

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. 4.13: Измененный текст программы

Создаю исполняемый файл и запускаю его. (рис. 4.14). Программа работает исправно.




```
anastasiadefedora: /work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ nasm -f elf lab8-3.asm
anastasiadefedora: /work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
anastasiadefedora: /work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 54600
anastasiadefedora: /work/study/2023-2024/Архитектура компьютера/arch-pc/lab08$
```

Рис. 4.14: Результат работы программы

4.3 Выполнение заданий для самостоятельной работы

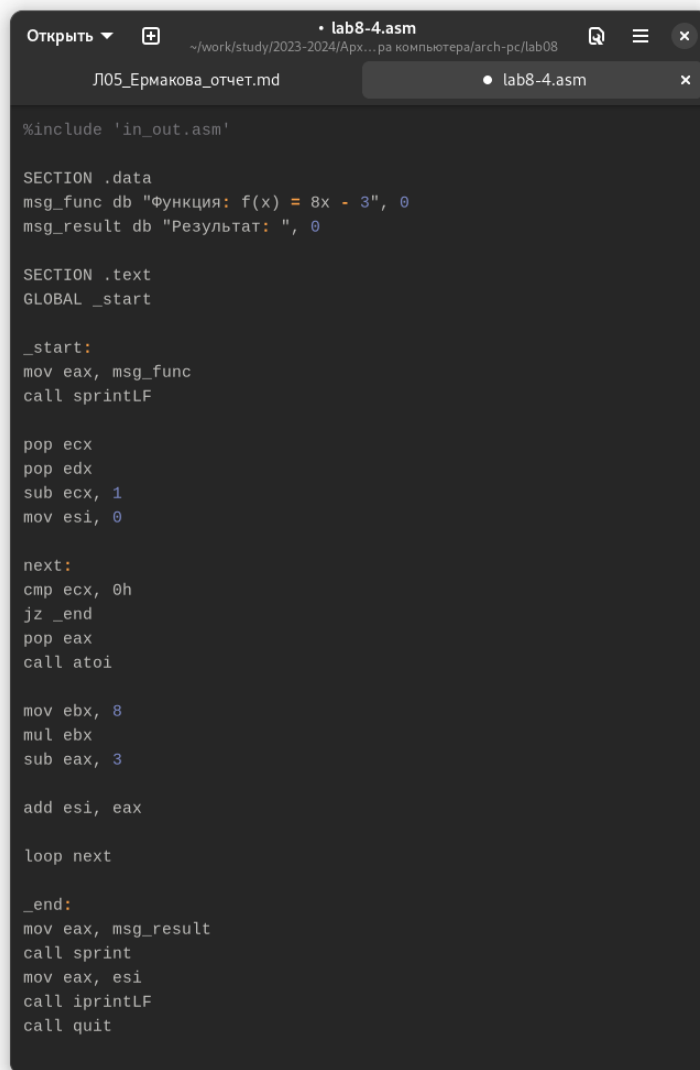
Напишу программу, которая находит сумму значений функции $f(x)$ для введенных значений x . Из таблицы 8.1 выбираю свой вариант (19), полученный в ходе выполнения одной из предыдущих лабораторных работ. Создаю файл lab8-4.asm для написания кода. (рис. 4.15).



```
anastasiiafedora: /work/study/2023-2024/Архитектура компьютера/arch-pr/lab8$ touch lab8-4.asm
```

Рис. 4.15: Создание файла lab8-4.asm

Ввожу текст кода. (рис. 4.16).



```
Открыть ▾ + • lab8-4.asm
~/work/study/2023-2024/Арх...ра компьютера/arch-pc/lab08
Л05_Ермакова_отчет.md • lab8-4.asm x

%include 'in_out.asm'

SECTION .data
msg_func db "Функция: f(x) = 8x - 3", 0
msg_result db "Результат: ", 0

SECTION .text
GLOBAL _start

_start:
mov eax, msg_func
call sprintf

pop ecx
pop edx
sub ecx, 1
mov esi, 0

next:
cmp ecx, 0h
jz _end
pop eax
call atoi

mov ebx, 8
mul ebx
sub eax, 3

add esi, eax

loop next

_end:
mov eax, msg_result
call sprintf
mov eax, esi
call iprintf
call quit
```

Рис. 4.16: Текст программы файла

Код программы:

```
%include 'in_out.asm'

SECTION .data
msg_func db "Функция: f(x) = 8x - 3", 0
msg_result db "Результат: ", 0
```



```

SECTION .text
GLOBAL _start

_start:
mov eax, msg_func
call sprintf

pop ecx
pop edx
sub ecx, 1
mov esi, 0

next:
cmp ecx, 0h
jz _end
pop eax
call atoi

mov ebx, 8
mul ebx
sub eax, 3

add esi, eax

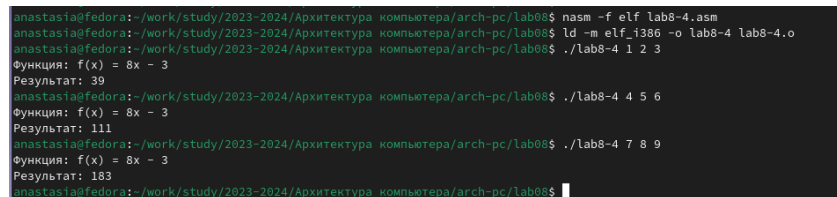
loop next

_end:
mov eax, msg_result

```

```
call sprint
mov eax, esi
call iprintLF
call quit
```

Создаю исполняемый файл и запускаю его. (рис. 4.17).



```
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ nasm -f elf lab8-4.asm
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ ld -m elf_i386 -o lab8-4 lab8-4.o
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ ./lab8-4 1 2 3
Функция: f(x) = 8x - 3
Результат: 39
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ ./lab8-4 4 5 6
Функция: f(x) = 8x - 3
Результат: 111
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$ ./lab8-4 7 8 9
Функция: f(x) = 8x - 3
Результат: 183
anastasia@fedora: ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab8$
```

Рис. 4.17: Результат работы файла

Проверяю работу программы трижды, введя разные аргументы. Программа работает исправно, это легко проверить, посчитав самому.

5 Выводы

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

6 Список литературы

1. Архитектура ЭВМ