# Отчёт по лабораторной работе №8

**Дисциплина: Архитектура компьютеров**

Шабалина Елизавета Андреевна

# Содержание

# Список иллюстраций

# Список таблиц

# 1 Цель работы

Целью работы является приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки в NASM.

# 2 Задание

1. Напишите программу, которая находит сумму значений функции f(x) для x = x1, x2, ..., xn, т.е. программа должна выводить значение f(x1) + f(x2) + ... + f(xn). Значения xi 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 = x1, x2 ..., xn.

# 3 Теоретическое введение

# 4 Выполнение лабораторной работы

## 4.1 4.1 Реализация циклов в NASM

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

```
eashabalina@dk3n55 ~ $ mkdir ~/work/arch-pc/lab08
eashabalina@dk3n55 ~ $ cd ~/work/arch-pc/lab08
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ touch lab8-1.asm
```

Рис. 4.1: Переход в каталог и создание файла

2. Ввожу в файл lab8-1.asm текст программы из листинга 8.1. Запускаю исполняемый файл.

```
lab8-1.asm          [----]  0 L:[  1+ 0   1/ 35]
%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
```

```
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ nasm -
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ld -m
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ./lab8
Введите N: 5
5
4
3
2
1
```

3. Изменим текст программы, добавив изменение значение регистра ecx в цикле. Запустим исправленную программу. Число проходов цикла не соответствует значению, введенному с клавиатуры.

```
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
loop label ; `ecx=ecx-1` и если `ecx` не '0'
; переход на `label`
call quit
```

```
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ nasm -
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ld -m
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ./lab8
Введите N: 4
3
1
```

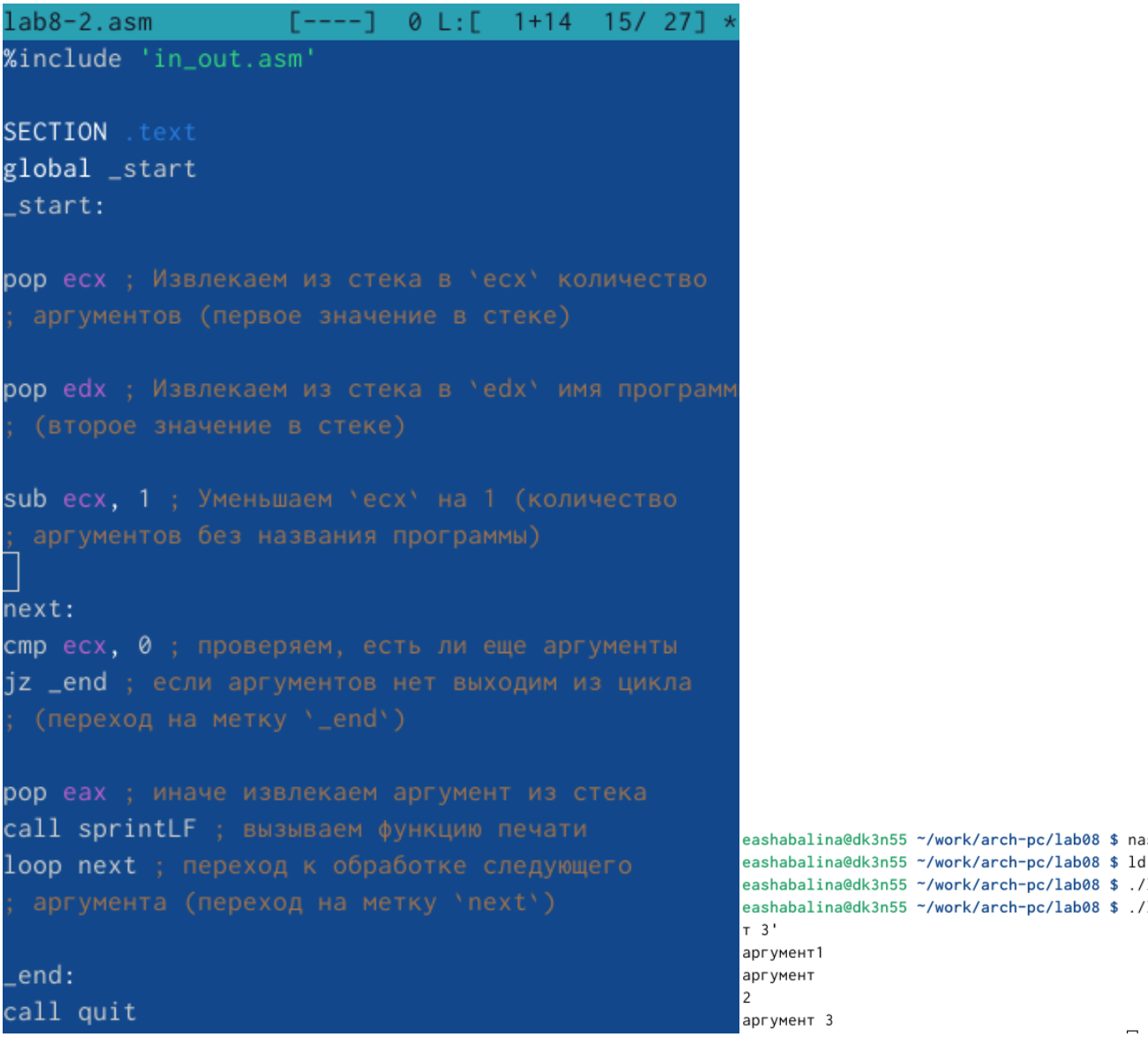
4. Внесем изменения в текст программы добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop. Запустим программу и проверим ее работу. Теперь число проходов цикла соответствует числу, введенному с клавиатуры.

```
; ------ Организация цикла
mov ecx,[N] ; Счетчик цикла, `ecx=N`
label:
push ecx ; добавление значения ecx в стек
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
pop ecx
loop label ; `ecx=ecx-1` и если `ecx` не '0'
; переход на `label`
call quit
```

```
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ nasm -
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ld -m
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ./lab8
Введите N: 5
4
3
2
1
0
```

## 4.2 4.2 Обработка аргументов командной строки

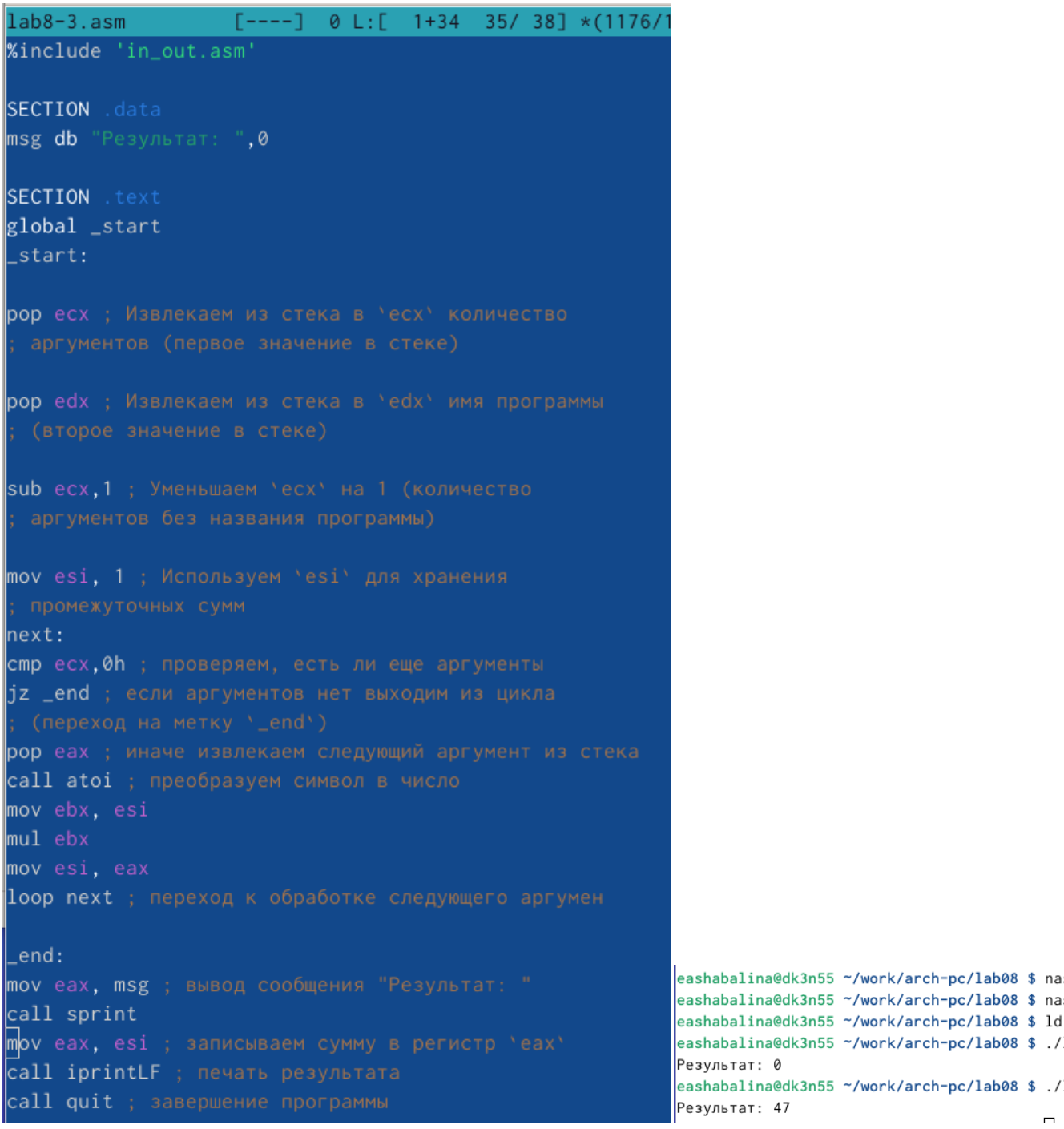
5. Создаем файл lab8-2.asm. Вводим в него программу из листинга 8.2. Программа обработала 4 аргумента.



```
lab8-2.asm          [----]  0 L:[  1+14  15/ 27] *
%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 программ
; (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 sprintLF ; вызываем функцию печати
loop next ; переход к обработке следующего
; аргумента (переход на метку `next`)

_end:
call quit
```

```
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ nasm -f
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ld -m el
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ./lab8-2
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ./lab8-2
т 3'
аргумент1
аргумент
2
аргумент 3
```

6. Создадим файл lab8-3.asm и введем в него текст программы из листинга 8.3.




7. Изменяю текст программы для вычисления произведения аргументов командной строки.

```
lab8-3.asm          [-M--]  0 L:[  1+28  29/ 38] *(991
%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, 1 ; 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
mov ebx, esi
mul ebx
mov esi, eax
loop next ; переход к обработке следующего аргумен

_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
```

```
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ nasm -
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ld -m
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ./lab8
Результат: 120
```

# 5 5 Задания для самостоятельной работы

1. Напишем программу, которая находит сумму значений функции f(x) для x=x1, x2, ..., xn, т.е. программа должна выводить значение f(x1) + f(x2) + ... + f(xn). Мой вариант - 10. Создадим исполняемый файл и проверим его работу на нескольких наборах x=x1, x2, ..., xn. Программа работает корректно.

```
lab8-4.asm          [----]  0 L:[  1+23   24/ 3
%include 'in_out.asm'

SECTION .data
msg_func db "Функция: f(x) = 10 * (x - 1)", 0
msg_result db "Результат: ", 0

SECTION .text
GLOBAL _start

_start:
mov eax, msg_func
call sprintLF

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

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

add eax, -1
mov ebx, 10
mul ebx

add esi, eax

loop next

_end:
mov eax, msg_result
call sprint
mov eax, esi
call iprintLF
call quit
```

```
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ nasm -f
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ld -m el
eashabalina@dk3n55 ~/work/arch-pc/lab08 $ ./lab8-4
Функция: f(x) = 10 * (x - 1)
Результат: 30
```

# 6 Выводы

В результате выполнения лабораторной работы я приобрёл навыки написания программ с использованием циклов и обработкой аргументов командной строки в NASM.

# Список литературы