

# **Лабораторная работа №9**

Щетинин Даниил Николаевич

# Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
4	Задание для самостоятельной работы	10
5	Выводы	12

## Список иллюстраций

3.1	работа lab9-1.asm . . . . .	6
3.2	работа нового lab9-1 . . . . .	7
3.3	работа lab9-1 . . . . .	7
3.4	проверка lab9-2 . . . . .	8
3.5	Проверка лаб9-3 . . . . .	8
3.6	код лаб9-3 (2) . . . . .	9
3.7	проверка работы . . . . .	9
4.1	проверка 9.asm . . . . .	10
4.2	код 9.asm . . . . .	11

# 1 Цель работы

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

## 2 Задание

Изучить циклы в NASM и написать программу вычисления функций

## 3 Выполнение лабораторной работы

### Шаг 1

Создадим каталог для лабораторной работы 9, а также файл lab9-1.asm:

Введём в него текст из Листинга 9.1, для того чтобы понять принцип работы инструкции loop

Создадим исполняемый файл и запустим его:

```
[dnthetinin@fedora lab09]$ nasm -f elf lab9-1.asm
[dnthetinin@fedora lab09]$ ld -m elf_i386 -o lab9-1 lab9-1.o
[dnthetinin@fedora lab09]$ ./lab9-1
Введите N: 3
3
2
1
```

Рис. 3.1: работа lab9-1.asm

Как мы видим, программа выдала

Изменим программу таким образом, чтобы она была ошибкой, из-за которой два раза подряд уменьшался параметр N и проверим работу файла:

```
[dnthetinin@fedora lab09]$ ./lab9-1
Введите N: 4
3
1
[dnthetinin@fedora lab09]$ ./lab9-1
Введите N: 6
5
3
1
```

Рис. 3.2: работа нового lab9-1

Как мы видим, файл пропускает некоторые значения N из-за некорректного кода

#### **Шаг 2**

еще раз изменим код, чтобы N корректно уменьшалось каждый цикл, при этом N = числу циклов

```
[dnthetinin@fedora lab09]$ ./lab9-1
Введите N: 4
3
2
1
0
```

Рис. 3.3: работа lab9-1

**Шаг 3** Создадим файл lab9-2.asm, и введем в него текст из Листинга 9.2, для того чтобы поочерёдно выводить аргументы на экран

Проверим его работу для различных аргументов, в том числе с кавычками:

```
[dnthetinin@fedora lab09]$ ./lab9-2 2 4 '5'
2
4
5
[dnthetinin@fedora lab09]$ ./lab9-2 1 2 '3'
1
2
3
[dnthetinin@fedora lab09]$ ./lab9-2 1 2 '1'
1
2
1
```

Рис. 3.4: проверка lab9-2

#### Шаг 4

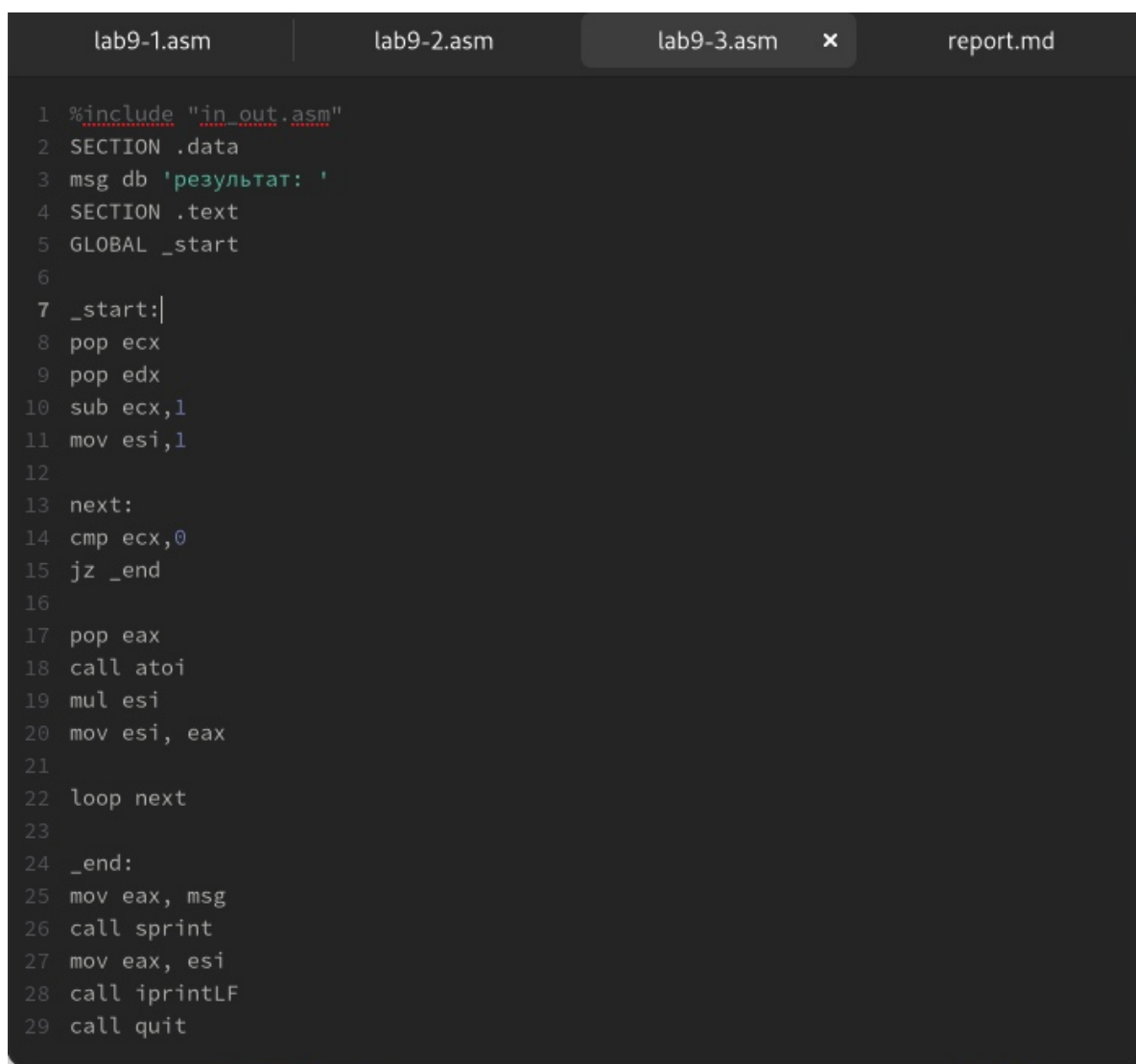
создадим файл lab9-3.asm, и введем в него текст из Листинга 9.3, для нахождения суммы аргументов, проверим работу

```
[dnthetinin@fedora lab09]$ touch lab9-3.asm
[dnthetinin@fedora lab09]$ nasm -f elf lab9-3.asm
[dnthetinin@fedora lab09]$ ld -m elf_i386 -o lab9-3 lab9-3.o
[dnthetinin@fedora lab09]$ ./lab9-3 2 4 5 5
Результат: 16
```

Рис. 3.5: Проверка лаб9-3

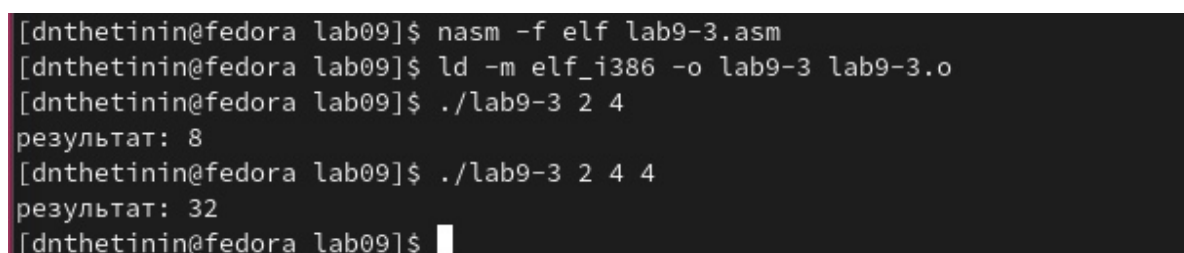
Изменим файл таким образом, чтобы вместо суммы мы искали произведение: просто в каждом цикле умножим eax на esi и перенесём значение eax в esi





```
1 %include "in_out.asm"
2 SECTION .data
3 msg db 'результат: '
4 SECTION .text
5 GLOBAL _start
6
7 _start:|
8 pop ecx
9 pop edx
10 sub ecx,1
11 mov esi,1
12
13 next:
14 cmp ecx,0
15 jz _end
16
17 pop eax
18 call atoi
19 mul esi
20 mov esi, eax
21
22 loop next
23
24 _end:
25 mov eax, msg
26 call sprint
27 mov eax, esi
28 call iprintLF
29 call quit
```

Рис. 3.6: код лаб9-3 (2)



```
[dnthetinin@fedora lab09]$ nasm -f elf lab9-3.asm
[dnthetinin@fedora lab09]$ ld -m elf_i386 -o lab9-3 lab9-3.o
[dnthetinin@fedora lab09]$ ./lab9-3 2 4
результат: 8
[dnthetinin@fedora lab09]$ ./lab9-3 2 4 4
результат: 32
[dnthetinin@fedora lab09]$
```

Рис. 3.7: проверка работы

## 4 Задание для самостоятельной работы

### Шаг 1

Создадим файл 9.asm для создания программы для нахождения функции за основу взяв код lab9-3.asm

как и в лаб9-3 мы каждый цикл просто умножаем x на 30 и вычитаем 11, записываем все в esi

проверим работу

```
[dnthetinin@fedora lab09]$ nasm -f elf 9.asm
[dnthetinin@fedora lab09]$ ld -m elf_i386 -o 9 9.o
[dnthetinin@fedora lab09]$ ./9 2 3 4
Результат: 237
```

Рис. 4.1: проверка 9.asm

```
1 %include 'in_out.asm'
2 SECTION .data
3 msg db "Результат: ",0
4 SECTION .text
5 global _start
6 _start:
7 pop ecx ; Извлекаем из стека в `ecx` количество
8 ; аргументов (первое значение в стеке)
9 pop edx ; Извлекаем из стека в `edx` имя программы
10 ; (второе значение в стеке)
11 sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
12 ; аргументов без названия программы)
13 mov esi, 0 ; Используем `esi` для хранения
14 ; промежуточных сумм
15 next:
16
17 cmp ecx,0h ; проверяем, есть ли еще аргументы
18 mov edi,30
19 jz _end ; если аргументов нет выходим из цикла
20 ; (переход на метку `_end`)
21 pop eax ; иначе извлекаем следующий аргумент из стека
22 call atoi ; преобразуем символ в число
23 mul edi
24 sub eax,11
25
26 add esi,eax ; добавляем к промежуточной сумме
27 ; след. аргумент `esi=esi+eax`
28 loop next ; переход к обработке следующего аргумента
29 _end:
30 mov eax, msg ; вывод сообщения "Результат: "
31 call sprint
32 mov eax, esi ; записываем сумму в регистр `eax`
33 call iprintfLF ; печать результата
34 call quit ; завершение программы
```

Рис. 4.2: код 9.asm

## 5 Выводы

Я смог успешно написать код для вычисления функции  $f(x)$