

# **Отчёт по лабораторной работе №8**

**дисциплина: Архитектура вычислительных систем**

Мосолов Александр Денисович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
2.1	Реализация циклов в NASM . . . . .	6
2.2	Обработка аргументов командной строки . . . . .	8
2.3	Задание для самостоятельной работы . . . . .	11
<b>3</b>	<b>Выводы</b>	<b>13</b>

## Список иллюстраций

2.1	Создаем каталог для работы . . . . .	6
2.2	Текст программы в соответствии с листингом 8.1 . . . . .	7
2.3	Вывод программы из листинга 8.1 . . . . .	7
2.4	Изменённый текст программы в соответствии с листингом 8.1 . .	8
2.5	Вывод изменённой программы из листинга 8.1 . . . . .	8
2.6	Текст программы lab8-2 . . . . .	9
2.7	Вывод программы lab8-2 . . . . .	9
2.8	Текст программы из листинга 8.3 . . . . .	10
2.9	Проверяем работу программы из листинга 8.3 . . . . .	10
2.10	Изменённый текст программы lab8-3 . . . . .	11
2.11	Вывод изменённой программы lab8-3 . . . . .	11
2.12	Текст программы из задания для самостоятельного выполнения (lab8-4) . . . . .	12
2.13	Вывод программы из задания для самостоятельного выполнения	12

## **Список таблиц**

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

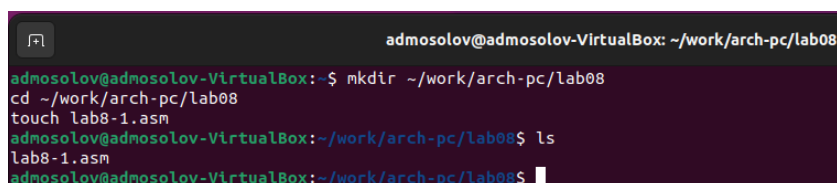
### 2.1 Реализация циклов в NASM

Создаём каталог для программ лабораторной работы № 8, переходим в него и создаём файл *lab8-1.asm*. С помощью команд:

```
mkdir ~/work/arch-pc/lab08
```

```
cd ~/work/arch-pc/lab08
```

```
touch lab8-1.asm
```

A screenshot of a terminal window with a dark background. The title bar at the top reads 'admosolov@admosolov-VirtualBox: ~/work/arch-pc/lab08'. The terminal shows the following commands and their outputs: 'mkdir ~/work/arch-pc/lab08' is executed, followed by 'cd ~/work/arch-pc/lab08', then 'touch lab8-1.asm'. Finally, the 'ls' command is executed, showing 'lab8-1.asm' as the only file in the directory. The prompt 'admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08\$' is visible at the bottom.

```
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$ mkdir ~/work/arch-pc/lab08
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$ cd ~/work/arch-pc/lab08
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$ touch lab8-1.asm
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$ ls
lab8-1.asm
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$
```

Рис. 2.1: Создаем каталог для работы

При реализации циклов в *NASM* с использованием инструкции *loop* необходимо помнить о том, что эта инструкция использует регистр *ecx* в качестве счетчика и на каждом шаге уменьшает его значение на единицу. В качестве примера рассмотрим программу, которая выводит значение регистра *ecx*. Внимательно изучим текст программы (*Листинг 8.1*).

```
mc [admosolov@admosolov-VirtualBox]:~/work/arch-pc/lab08
GNU nano 6.2 /home/admosolov/work/arch-pc/lab08/lab8-1.asm
_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] ; Счетчик цикла, `ecx=N`
label:
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения `N`
loop label
call quit
```

Рис. 2.2: Текст программы в соответствии с листингом 8.1

Для того, чтобы программа транслировалась без ошибок перенесем файл *in\_out.asm* в *~/work/arch-pc/lab08*.

Создадим исполняемый файл и запустим его. Результат работы данной программы будет следующим:

```
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
ld -m elf_i386 -o lab8-1 lab8-1.o
./lab8-1
Введите N: 3
3
2
1
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$
```

Рис. 2.3: Вывод программы из листинга 8.1

Данный пример показывает, что использование регистра *ecx* в теле цикла *loop* может привести к некорректной работе программы. Изменим текст программы добавив изменение значение регистра *ecx* в цикле:

```
mc [admosolov@admosolov-VirtualBox]:~/work/arch-pc/lab08
GNU nano 6.2 /home/admosolov/work/arch-pc/lab08/lab8-1.asm
mov eax,msg1
call sprint

mov ecx, N
mov edx, 10
call sread

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

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 ; извлечение значения ecx из стека
loop label
```

Рис. 2.4: Изменённый текст программы в соответствии с листингом 8.1

Транслируем текст, создаём объектный файл, компилируем его и запускаем программу *lab8-1*:

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

Рис. 2.5: Вывод изменённой программы из листинга 8.1

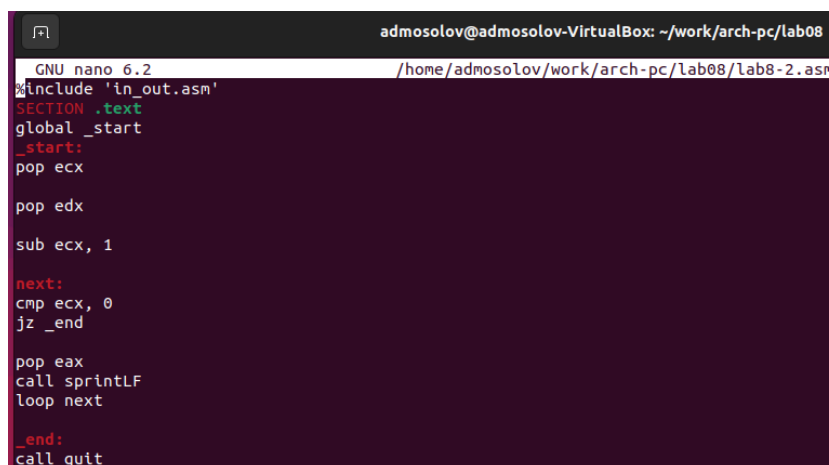
Число проходов цикла соответствует значению  $N$  введенному с клавиатуры, регистр *ecx* принимает значения все нечётные значения от 0 до  $N$ .

## 2.2 Обработка аргументов командной строки

Для того чтобы использовать аргументы в программе, их просто нужно извлечь из стека. Обработку аргументов нужно проводить в цикле. Т.е. сначала нужно



извлечь из стека количество аргументов, а затем циклично для каждого аргумента выполнить логику программы. В качестве примера рассмотрим программу, которая выводит на экран аргументы командной строки. Внимательно изучим текст программы (Листинг 8.2).



```
admosolov@admosolov-VirtualBox: ~/work/arch-pc/lab08
GNU nano 6.2 /home/admosolov/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
```

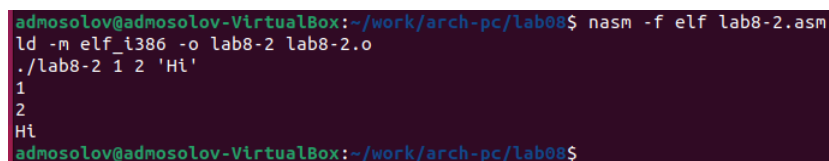
Рис. 2.6: Текст программы lab8-2

Транслируем текст, создаём объектный файл, компилируем его и запускаем программу *lab8-2*:

```
nasm -f elf lab8-2.asm
```

```
ld -m elf_i386 -o lab8-2 lab8-2.o
```

```
./lab8-2 1 2 'Hi'
```



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

Рис. 2.7: Вывод программы lab8-2

Программа обрабатывает 3 аргумента.

Рассмотрим еще один пример программы которая выводит сумму чисел, которые передаются в программу как аргументы. Создадим файл *lab8-3.asm* в каталоге *~/work/arch-pc/lab08* и введём в него текст программы из листинга 8.3.

```

admosolov@admosolov-VirtualBox: ~/work/arch-pc/lab08
GNU nano 6.2 /home/admosolov/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

```

Рис. 2.8: Текст программы из листинга 8.3

Создадим исполняемый файл и запустим его, указав аргументы.

```

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

```

Рис. 2.9: Проверяем работу программы из листинга 8.3

Измените текст программы из листинга 8.3 для вычисления произведения аргументов командной строки.

```

admosolov@admosolov-VirtualBox: ~/work/arch-pc/lab08
GNU nano 6.2 /home/admosolov/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
mul esi
mov esi, eax

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

```

Рис. 2.10: Изменённый текст программы lab8-3

Проверим работу программы:

```

admosolov@admosolov-VirtualBox: ~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
ld -m elf_i386 -o lab8-3 lab8-3.o
./lab8-3 12 5 2
Результат: 120

```

Рис. 2.11: Вывод изменённой программы lab8-3

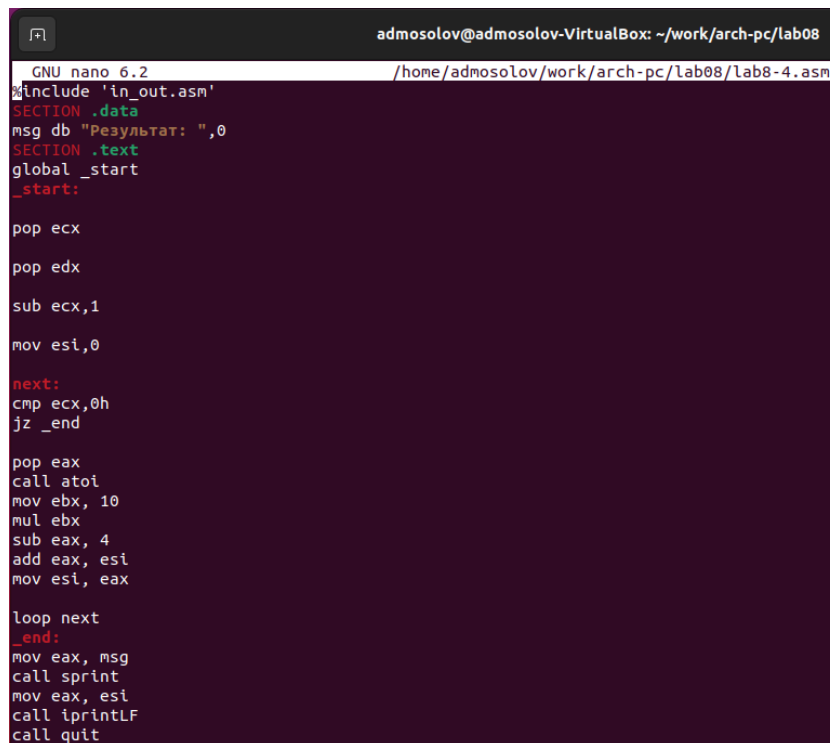
А теперь на вывод программы с использованием *iprint*.

## 2.3 Задание для самостоятельной работы

Напишите программу, которая находит сумму значений функции  $f(x)$  для  $x = x_1, x_2, \dots, x_n$ , т.е. программа должна выводить значение  $f(x_1) + f(x_2) + \dots + f(x_n)$ . Значения  $x_i$  передаются как аргументы. Вид функции  $f(x)$  выбрать из таблицы 8.1 вариантов заданий в соответствии с вариантом (в моём случае вариант 9), полученным при выполнении лабораторной работы № 7. Создайте исполняемый

файл и проверьте его работу на нескольких наборах  $x = x_1, x_2, \dots, x_n$ .

Напишем текст программы для  $f(x) = 10x - 4$  (вариант 9):



```
admosolov@admosolov-VirtualBox: ~/work/arch-pc/lab08
GNU nano 6.2 /home/admosolov/work/arch-pc/lab08/lab8-4.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

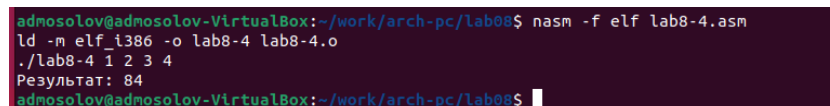
next:
    cmp ecx,0h
    jz _end

    pop eax
    call atoi
    mov ebx, 10
    mul ebx
    sub eax, 4
    add eax, esi
    mov esi, eax

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

Рис. 2.12: Текст программы из задания для самостоятельного выполнения (*lab8-4*)

Убедимся в правильности написанной программы.



```
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
ld -m elf_i386 -o lab8-4 lab8-4.o
./lab8-4 1 2 3 4
Результат: 84
admosolov@admosolov-VirtualBox:~/work/arch-pc/lab08$
```

Рис. 2.13: Вывод программы из задания для самостоятельного выполнения

## **3 Выводы**

В ходе лабораторной работы были приобретены навыки написания программ с использованием циклов и обработкой аргументов командной строки.