

# **Отчёт по лабораторной работе №8**

**Дисциплина: Архитектура Компьютеров**

Мургия Марк Максимович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>14</b>
	<b>Список литературы</b>	<b>15</b>

## Список иллюстраций

4.1	lab8-1.asm №1 . . . . .	8
4.2	Результат исполнения . . . . .	9
4.3	lab8-1.asm №2 . . . . .	9
4.4	С моим вводом появились только нечетные числа 1-9 . . . . .	10
4.5	lab8-1.asm №3 . . . . .	10
4.6	Числа $0 < i < N$ , где $N = 10$ . . . . .	11
4.7	lab8-2.asm . . . . .	11
4.8	Аргумент1 единственный; Аргумент 2 поделен на две части из-за пробела; 'Аргумент 3' - стринг . . . . .	11
4.9	lab8-3.asm . . . . .	12
4.10	Вся сумма . . . . .	12
4.11	Последний ассемблерский файл . . . . .	13
4.12	Последний результат . . . . .	13

## **Список таблиц**

# 1 Цель работы

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

## **2 Задание**

1. Понять работу циклов
2. Использовать обработку аргументов командной строки

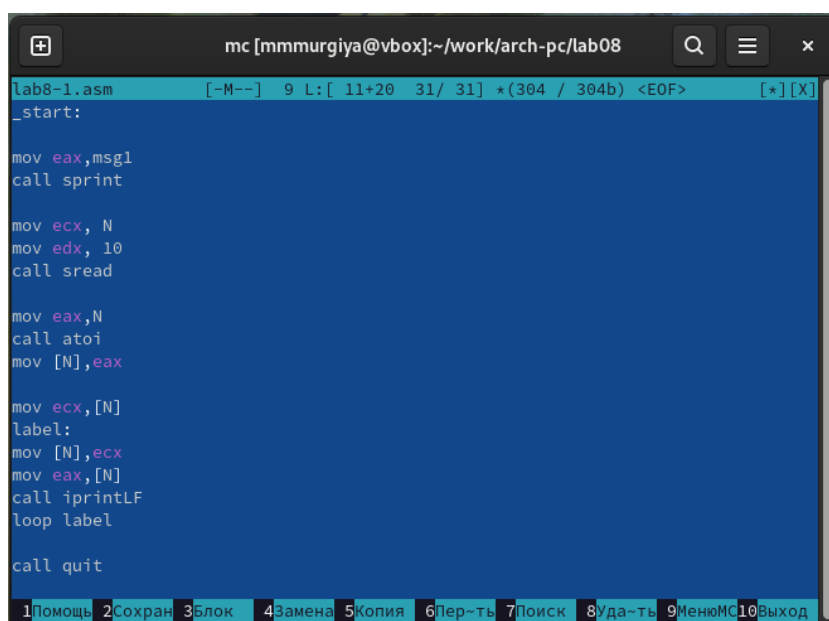
## 3 Теоретическое введение

Идей для таблиц больше нет.

push мы используем для вставки в стек значение переменной или регистра, а pop используется для извлечения элемента из стека. loop заставляет команды в скрипте повторяться несколько раз или при каком-либо условии.

## 4 Выполнение лабораторной работы

Первая версия lab8-1.asm выводит числа от 1 до N. Цикл используется для перехода к следующему числу.



```
lab8-1.asm [-M--] 9 L: [ 11+20 31/ 31] *(304 / 304b) <EOF> [*][X]
_start:

mov eax,msg1
call sprint

mov ecx, N
mov edx, 10
call sread

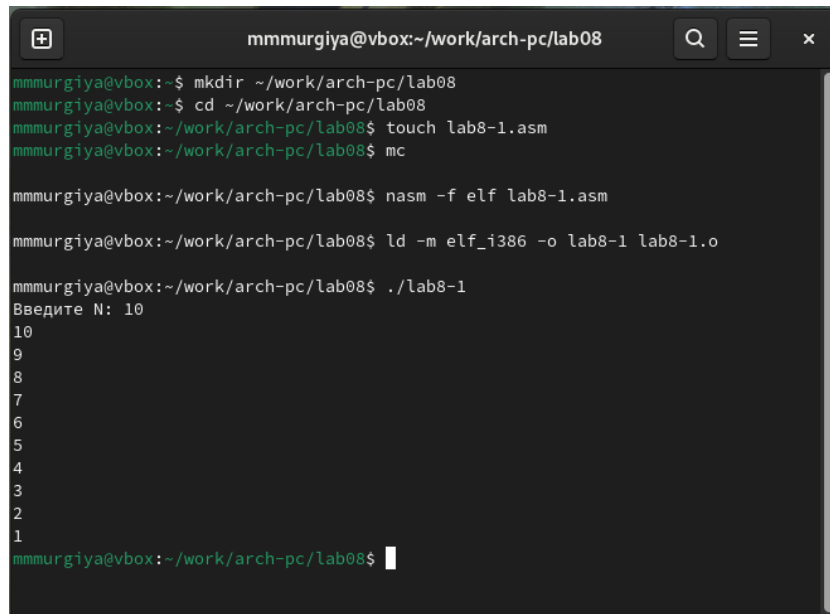
mov eax,N
call atoi
mov [N],eax

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

call quit
```

Рис. 4.1: lab8-1.asm №1





```
mmmurgiya@vbox:~/work/arch-pc/lab08
mmmurgiya@vbox:~$ mkdir ~/work/arch-pc/lab08
mmmurgiya@vbox:~$ cd ~/work/arch-pc/lab08
mmmurgiya@vbox:~/work/arch-pc/lab08$ touch lab8-1.asm
mmmurgiya@vbox:~/work/arch-pc/lab08$ mc

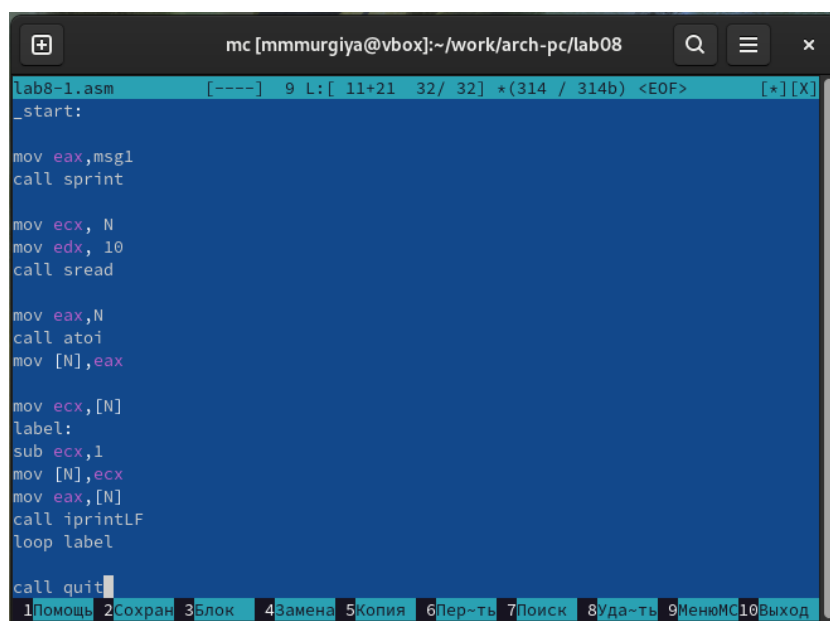
mmmurgiya@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm

mmmurgiya@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o

mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
mmmurgiya@vbox:~/work/arch-pc/lab08$
```

Рис. 4.2: Результат исполнения

Добавив `sub ecx,1`, мы получаем четные или не четные числа меньше N.



```
lab8-1.asm  [----]  9  L:[ 11+21  32/ 32] *(314 / 314b) <EOF>  [*][X]
_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. 4.3: lab8-1.asm №2

```
mmmurgiya@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o

mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1

mmmurgiya@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
mmmurgiya@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
7
5
3
1

mmmurgiya@vbox:~/work/arch-pc/lab08$
```

Рис. 4.4: С моим вводом появились только нечетные числа 1-9

Если добавить функции push и pop, то получится программа, выводящая числа меньше N, начиная с нуля.

```
lab8-1.asm  [----]  0 L: [ 14+21  35/ 35] *(323 / 332b) 0099 0x063 [*][X]
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
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюМС10Выход
```

Рис. 4.5: lab8-1.asm №3

```
mmmurgiya@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
mmmurgiya@vbox:~/work/arch-pc/lab08$ 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
mmmurgiya@vbox:~/work/arch-pc/lab08$
```

Рис. 4.6: Числа  $0 < i < N$ , где  $N = 10$

Следующий скрипт не только показывает использование аргументов в командной строке, но также как терминал понимает аргументы разных типов.

```
mc [mmmurgiya@vbox]:~/work/arch-pc/lab08
lab8-2.asm  [----]  9 L: [ 1+21 22/ 22] *(164 / 164b) <EOF>  [*][X]
#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

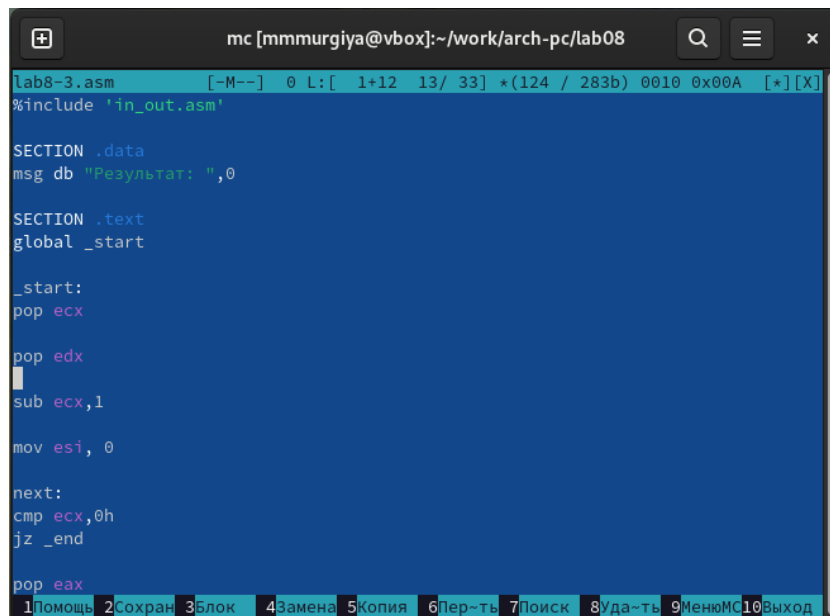
_end:
call quit
```

Рис. 4.7: lab8-2.asm

```
mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
mmmurgiya@vbox:~/work/arch-pc/lab08$
```

Рис. 4.8: Аргумент1 единственный; Аргумент 2 поделен на две части из-за пробела; 'Аргумент 3' - строинг

Третий скрипт суммирует все аргументы в командной строке.



```
lab8-3.asm [-M--] 0 L: [ 1+12 13/ 33] *(124 / 283b) 0010 0x00A [*][X]
#include 'in_out.asm'

SECTION .data
msg db "Результат: ",0

SECTION .text
global _start

_start:
pop ecx

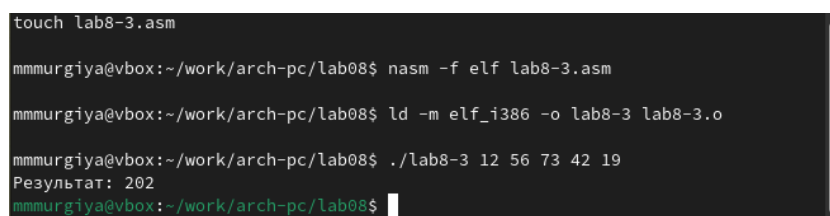
pop edx
sub ecx,1

mov esi, 0

next:
cmp ecx,0h
jz _end

pop eax
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюМС10Выход
```

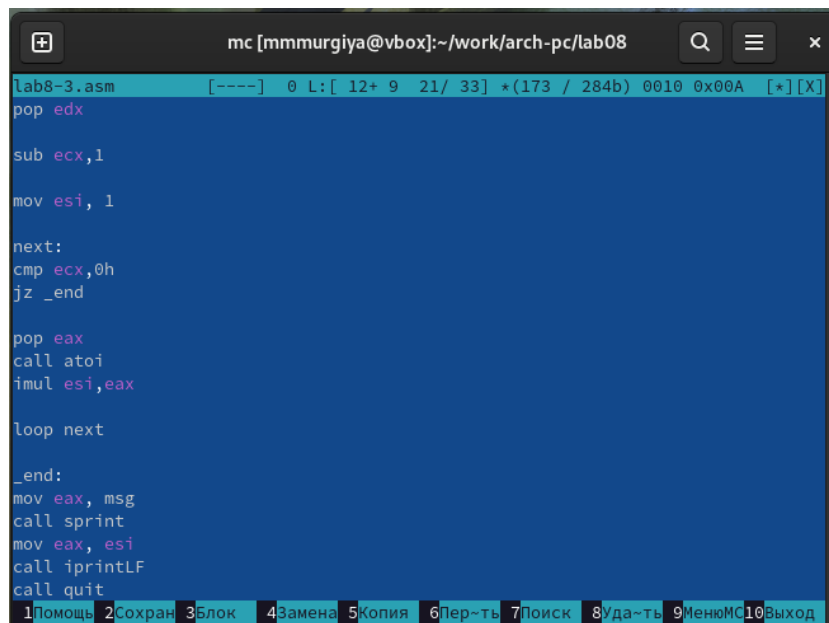
Рис. 4.9: lab8-3.asm



```
touch lab8-3.asm
mmmurgiya@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
mmmurgiya@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-3 12 56 73 42 19
Результат: 202
mmmurgiya@vbox:~/work/arch-pc/lab08$
```

Рис. 4.10: Вся сумма

Для Лабораторной работы нужно изменить программу, что умножила все аргументы друг на друга.



```
lab8-3.asm [----] 0 L: [ 12+ 9 21/ 33] *(173 / 284b) 0010 0x00A [*] [X]
pop edx

sub ecx,1

mov esi,1

next:
cmp ecx,0h
jz _end

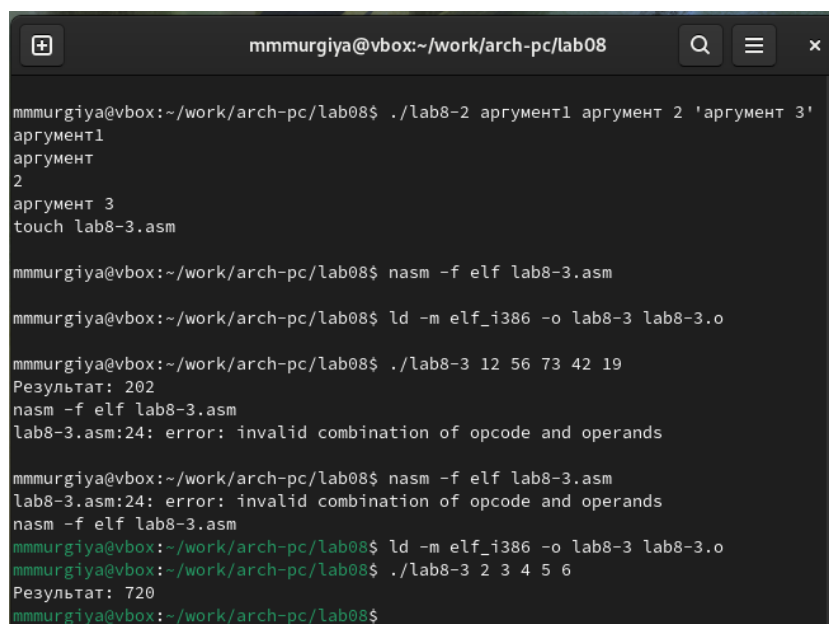
pop eax
call atoi
imul esi,eax

loop next

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

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9Меню10Выход

Рис. 4.11: Последний ассемблерский файл



```
mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
touch lab8-3.asm

mmmurgiya@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm

mmmurgiya@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o

mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-3 12 56 73 42 19
Результат: 202
nasm -f elf lab8-3.asm
lab8-3.asm:24: error: invalid combination of opcode and operands

mmmurgiya@vbox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
lab8-3.asm:24: error: invalid combination of opcode and operands
nasm -f elf lab8-3.asm
mmmurgiya@vbox:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
mmmurgiya@vbox:~/work/arch-pc/lab08$ ./lab8-3 2 3 4 5 6
Результат: 720
mmmurgiya@vbox:~/work/arch-pc/lab08$
```

Рис. 4.12: Последний результат

## **5 Выводы**

Мы приобрели навыки написания программ с использованием циклов и обработки аргументов командной строки.

## **Список литературы**