

# **Отчет по выполнению лабораторной работы №8**

**Дисциплина: архитектура компьютеров**

Намруев Максим Саналович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
3.1	Реализация циклов в NASM . . . . .	7
3.2	Обработка аргументов командной строки . . . . .	12
<b>4</b>	<b>Выводы</b>	<b>22</b>

## Список иллюстраций

3.1	Создание каталога и файла . . . . .	7
3.2	Ввод программы из листинга 8.1 . . . . .	8
3.3	Проверка работы файла . . . . .	8
3.4	Изменение файла . . . . .	9
3.5	Проверка работы файла . . . . .	10
3.6	Внесение изменений в текст программы . . . . .	11
3.7	Запуск программы . . . . .	12
3.8	Создание файла lab8-2.asm . . . . .	12
3.9	Ввод программы из листинга 8.2 . . . . .	13
3.10	Запуск программы . . . . .	13
3.11	Ввод программы из листинга 8.3 . . . . .	14
3.12	Проверка работы файла . . . . .	14
3.13	Изменение текста листинга 8.3 . . . . .	16
3.14	Проверка работы программы . . . . .	17
3.15	Написание программы для самостоятельной работы . . . . .	18
3.16	Написание программы для самостоятельной работы . . . . .	19
3.17	Запуск программы . . . . .	20

## Список таблиц

# 1 Цель работы

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

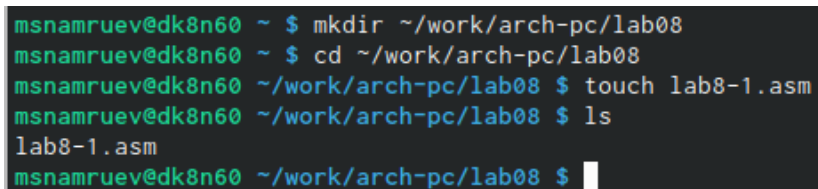
## 2 Задание

1. Реализация циклов в NASM
2. Обработка аргументов командной строки

## 3 Выполнение лабораторной работы

### 3.1 Реализация циклов в NASM

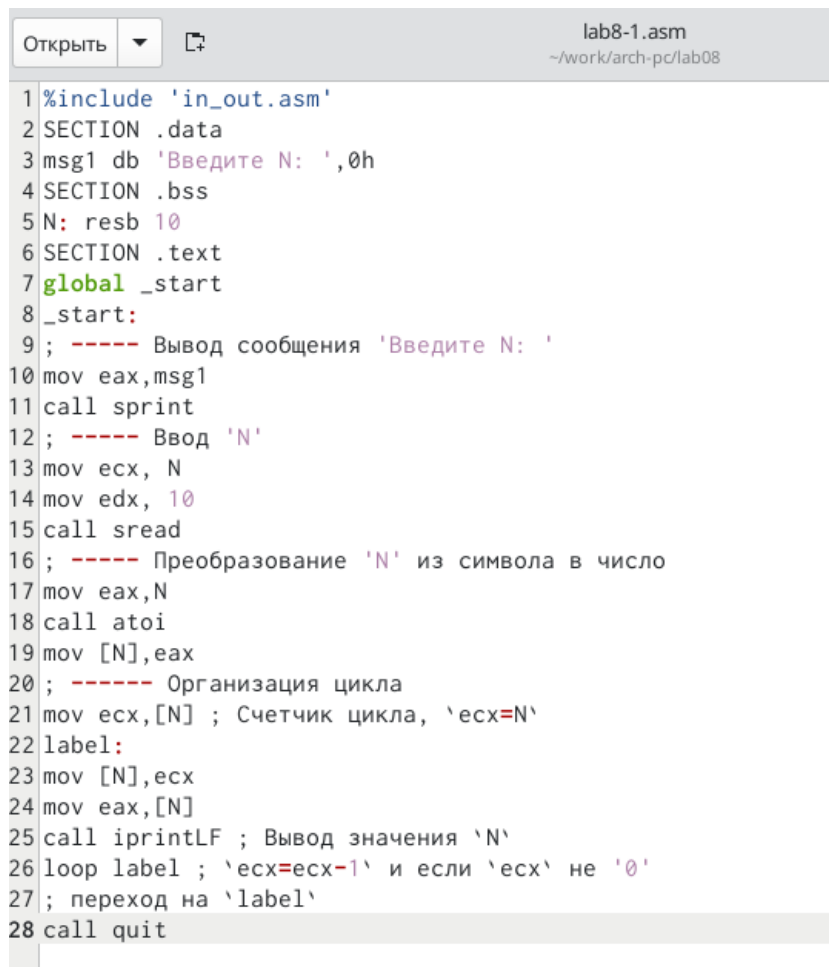
Создаю каталог lab08, перехожу в него и создаю файл lab8-1.asm (рис. [3.1]).



```
msnamruev@dk8n60 ~ $ mkdir ~/work/arch-pc/lab08
msnamruev@dk8n60 ~ $ cd ~/work/arch-pc/lab08
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ touch lab8-1.asm
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ ls
lab8-1.asm
msnamruev@dk8n60 ~/work/arch-pc/lab08 $
```

Рис. 3.1: Создание каталога и файла

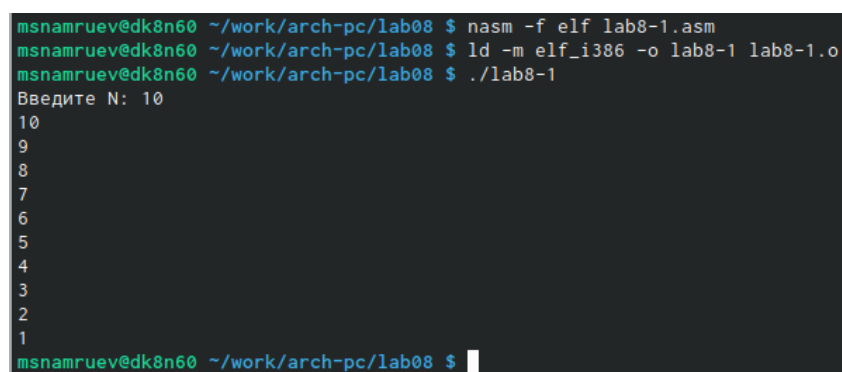
Ввожу в файл текст программы из листинга 8.1.(рис. [3.2]).



```
1 %include 'in_out.asm'
2 SECTION .data
3 msg1 db 'Введите N: ',0h
4 SECTION .bss
5 N: resb 10
6 SECTION .text
7 global _start
8 _start:
9 ; ----- Вывод сообщения 'Введите N: '
10 mov eax,msg1
11 call sprint
12 ; ----- Ввод 'N'
13 mov ecx, N
14 mov edx, 10
15 call sread
16 ; ----- Преобразование 'N' из символа в число
17 mov eax,N
18 call atoi
19 mov [N],eax
20 ; ----- Организация цикла
21 mov ecx,[N] ; Счетчик цикла, 'ecx=N'
22 label:
23 mov [N],ecx
24 mov eax,[N]
25 call iprintLF ; Вывод значения 'N'
26 loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
27 ; переход на 'label'
28 call quit
```

Рис. 3.2: Ввод программы из листинга 8.1

Создаю исполняемый файл и проверяю его работу.(рис. [3.3]).



```
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
msnamruev@dk8n60 ~/work/arch-pc/lab08 $
```

Рис. 3.3: Проверка работы файла



Далее изменяю текст программы, изменив значение регистра ecx в цикле.(рис. [3.4]).

```
1 %include 'in_out.asm'
2 SECTION .data
3 msg1 db 'Введите N: ',0h
4 SECTION .bss
5 N: resb 10
6 SECTION .text
7 global _start
8 _start:
9 ; ----- Вывод сообщения 'Введите N: '
10 mov eax,msg1
11 call sprint
12 ; ----- Ввод 'N'
13 mov ecx, N
14 mov edx, 10
15 call sread
16 ; ----- Преобразование 'N' из символа в число
17 mov eax,N
18 call atoi
19 mov [N],eax
20 ; ----- Организация цикла
21 mov ecx,[N] ; Счетчик цикла, 'ecx=N'
22 label:
23 sub ecx,1 ; 'ecx=ecx-1'
24 mov [N],ecx
25 mov eax,[N]
26 call iprintLF
27 loop label
28 call quit
```

Рис. 3.4: Изменение файла

Создаю исполняемый файл и проверяю его работу.(рис. [3.5]).

```
4294429866
4294429864
4294429862
4294429860
4294429858
4294429856
4294429854
4294429852
4294429850
4294429848
4294429846
4294429844
4294429842
4294429840
4294429838
4294429836
4294429834
4294429832
4294429830
4294429828
4294429826
4294429824
4294429822
4294429820
4294429818
4294429816
4294429814
4294429812
4
```

Рис. 3.5: Проверка работы файла

В данном случае число проходов цикла не соответствует значению N.

Вношу изменения в текст программы, добавив команды push и pop.(рис. [3.6]).

```

1 %include 'in_out.asm'
2 SECTION .data
3 msg1 db 'Введите N: ',0h
4 SECTION .bss
5 N: resb 10
6 SECTION .text
7 global _start
8 _start:
9 ; ----- Вывод сообщения 'Введите N: '
10 mov eax,msg1
11 call sprint
12 ; ----- Ввод 'N'
13 mov ecx, N
14 mov edx, 10
15 call sread
16 ; ----- Преобразование 'N' из символа в число
17 mov eax,N
18 call atoi
19 mov [N],eax
20 ; ----- Организация цикла
21 mov ecx,[N] ; Счетчик цикла, 'ecx=N'
22 label:
23 push ecx ; добавление значения ecx в стек
24 sub ecx,1
25 mov [N],ecx
26 mov eax,[N]
27 call iprintLF
28 pop ecx ; извлечение значения ecx из стека
29 loop label
30 call quit

```

Рис. 3.6: Внесение изменений в текст программы

Создаю исполняемый файл и запускаю его.(рис. [3.7]).

```

msnamruev@dk8n60 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
msnamruev@dk8n60 ~/work/arch-pc/lab08 $

```

Рис. 3.7: Запуск программы

В данном случае число проходов цикла соответствует значению N.

## 3.2 Обработка аргументов командной строки

Создаю файл lab8-2.asm и проверяю его создание.(рис. [3.8]).

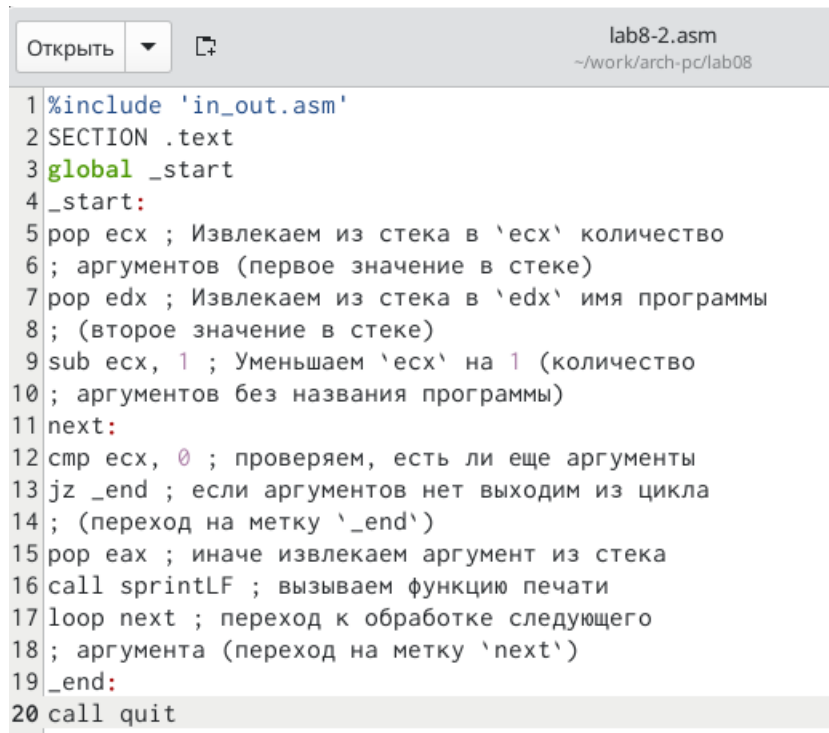
```

msnamruev@dk8n60 ~/work/arch-pc/lab08 $ touch lab8-2.asm
msnamruev@dk8n60 ~/work/arch-pc/lab08 $ ls
in_out.asm lab8-1 lab8-1.asm lab8-1.o lab8-2.asm
msnamruev@dk8n60 ~/work/arch-pc/lab08 $

```

Рис. 3.8: Создание файла lab8-2.asm

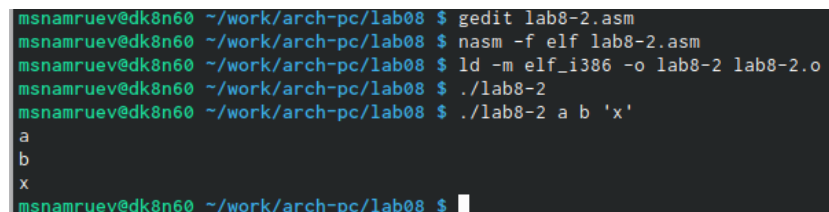
Ввожу в этот файл программу из листинга 8.2.(рис. [3.9]).



```
1 %include 'in_out.asm'
2 SECTION .text
3 global _start
4 _start:
5     pop ecx ; Извлекаем из стека в 'ecx' количество
6     ; аргументов (первое значение в стеке)
7     pop edx ; Извлекаем из стека в 'edx' имя программы
8     ; (второе значение в стеке)
9     sub ecx, 1 ; Уменьшаем 'ecx' на 1 (количество
10    ; аргументов без названия программы)
11 next:
12     cmp ecx, 0 ; проверяем, есть ли еще аргументы
13     jz _end ; если аргументов нет выходим из цикла
14     ; (переход на метку '_end')
15     pop eax ; иначе извлекаем аргумент из стека
16     call sprintLF ; вызываем функцию печати
17     loop next ; переход к обработке следующего
18     ; аргумента (переход на метку 'next')
19 _end:
20 call quit
```

Рис. 3.9: Ввод программы из листинга 8.2

Создаю исполняемый файл и запускаю его, указав аргументы.(рис. [3.10]).

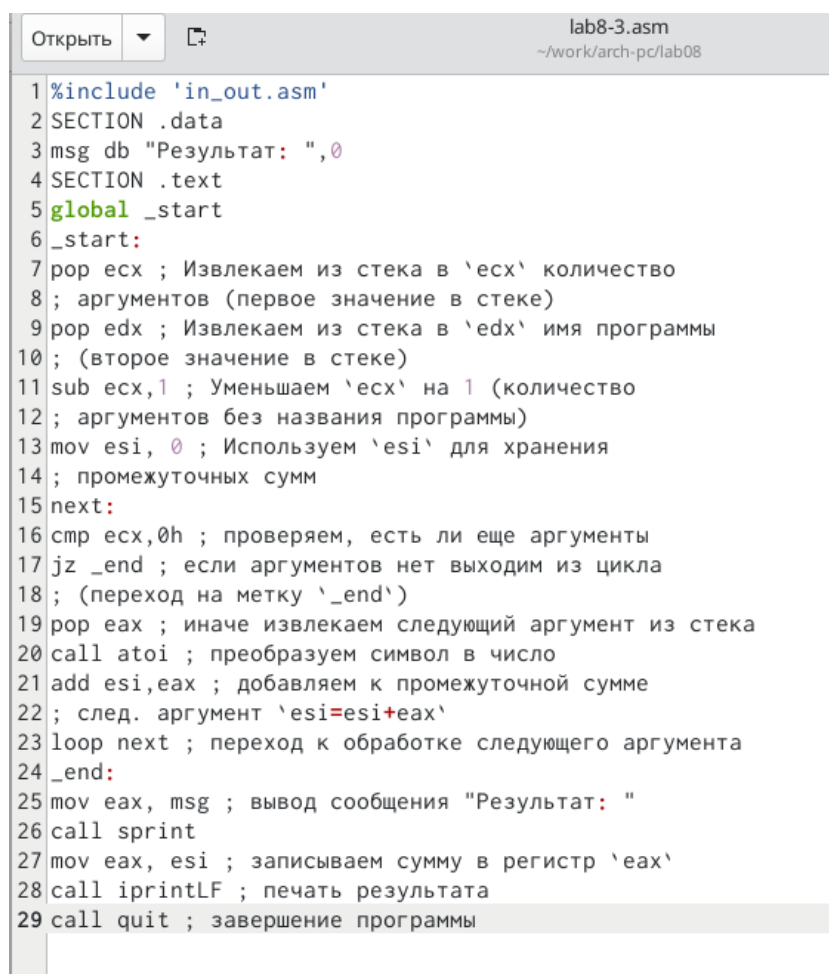


```
msnamruev@dk8n60 ~/.work/arch-pc/lab08 $ gedit lab8-2.asm
msnamruev@dk8n60 ~/.work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
msnamruev@dk8n60 ~/.work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
msnamruev@dk8n60 ~/.work/arch-pc/lab08 $ ./lab8-2
msnamruev@dk8n60 ~/.work/arch-pc/lab08 $ ./lab8-2 a b 'x'
a
b
x
msnamruev@dk8n60 ~/.work/arch-pc/lab08 $
```

Рис. 3.10: Запуск программы

Программа обработала 3 аргумента.

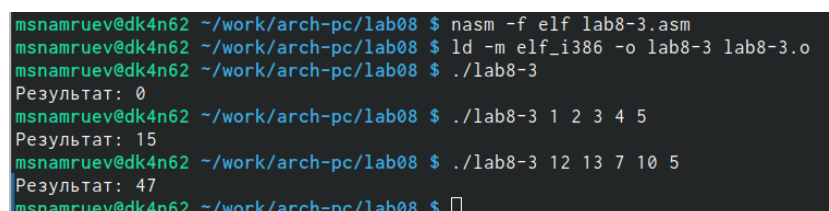
Создаю файл lab8-3.asm и ввожу в него программу из листинга 8.3.(рис. [3.11]).



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

Рис. 3.11: Ввод программы из листинга 8.3

Созаю исполняемый файл и запускаю его, указав аргументы.(рис. [3.12]).



```
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ./lab8-3
Результат: 0
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ./lab8-3 1 2 3 4 5
Результат: 15
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 10 5
Результат: 47
msnamruev@dk4n62 ~/work/arch-pc/lab08 $
```

Рис. 3.12: Проверка работы файла

Программа работает.

Теперь изменяю текст программы из листинга 8.3 так, чтобы она вычисляла

произведение аргументов каждой строки.(рис. [3.13]).

```

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

```

Рис. 3.13: Изменение текста листинга 8.3



Создаю исполняемый файл и запускаю его, указав аргументы.(рис. [3.14]).

```
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ./lab8-3 1 2 3 4 5
Результат: 120
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ./lab8-3 1 2 3 4 5 0
Результат: 0
msnamruev@dk4n62 ~/work/arch-pc/lab08 $
```

Рис. 3.14: Проверка работы программы

Текст программы: %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

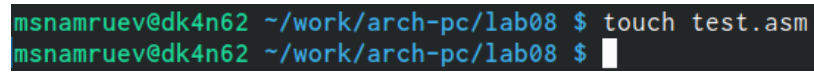
mov eax, esi

call iprintLF

call quit

#Задания для самостоятельной работы

Создаю файл test.asm.(рис. [3.15]).



```
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ touch test.asm
msnamruev@dk4n62 ~/work/arch-pc/lab08 $
```

Рис. 3.15: Написание программы для самостоятельной работы

Начинаю написание программы, которая будет вычислять сумму значений  $f(x)=30x-11$ .(вариант 16).(рис. [3.16]).

```

1 %include 'in_out.asm'
2 SECTION .data
3 msg db "Результат: ",0
4 SECTION .text
5 global _start
6 _start:
7 pop ecx
8 pop edx
9 sub ecx,1
10 mov esi, 0
11 mov edi, 30
12 next:
13 cmp ecx,0h
14 jz _end
15 pop eax
16 call atoi
17 mul edi
18 sub eax,11
19 add esi,eax
20 loop next
21 _end:
22 mov eax, msg
23 call sprint
24 mov eax, esi
25 call iprintLF
26 call quit

```

Рис. 3.16: Написание программы для самостоятельной работы

Создаю исполняемый файл и запускаю его, указав аргументы.(рис. [3.17]).

```
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ nasm -f elf test.asm
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o test test.o
msnamruev@dk4n62 ~/work/arch-pc/lab08 $ ./test 10 2 3 6 9
Результат: 845
msnamruev@dk4n62 ~/work/arch-pc/lab08 $
```

Рис. 3.17: Запуск программы

Программы работает.

Текст прогораммы: %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

mov edi,30

next:

cmp ecx,0h

jz \_end

pop eax

call atoi

mul edi

sub eax,11

add esi,eax

loop next

\_end:

mov eax,msg

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

## 4 Выводы

После выполнения данной лабораторной работы я приобрел навыки написания программ с использованием циклов и обработкой аргументов командной строки