

# Отчёт по лабораторной работе №8

Дисциплина: архитектура компьютеров и операционные системы

Сахно Никита НКАбд-05-23

## Содержание

Цель работы .....	1
Задание .....	1
Выполнение лабораторной работы .....	1
Реализация циклов в NASM.....	1
Обработка аргументов командной строки .....	5
Задание для самостоятельной работы .....	8
Выводы .....	9
Список литературы .....	9

## Цель работы

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

## Задание

1. Реализация циклов в NASM.
2. Обработка аргументов командной строки.
3. Задание для самостоятельной работы.

## Выполнение лабораторной работы

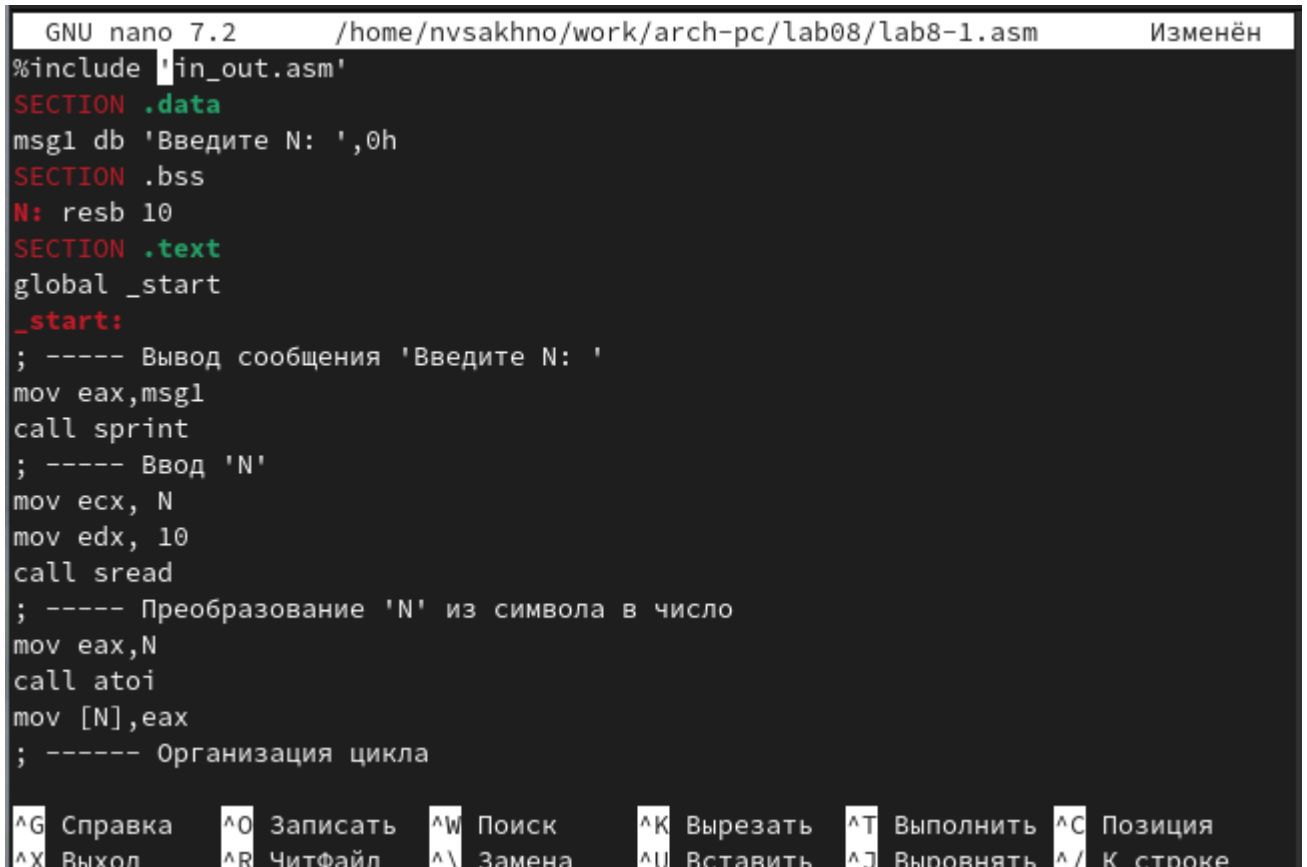
### Реализация циклов в NASM

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

```
[nvsakhno@fedora ~]$ mkdir ~/work/arch-pc/lab08
[nvsakhno@fedora ~]$ cd ~/work/arch-pc/lab08
[nvsakhno@fedora lab08]$ touch lab8-1.asm
[nvsakhno@fedora lab08]$
```

### *Создание файлов для лабораторной работы*

Ввожу в файл lab8-1.asm текст программы из листинга 8.1. (рис. 2).



```
GNU nano 7.2 /home/nvsakhno/work/arch-pc/lab08/lab8-1.asm Изменён
%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

^G Справка  ^O Записать  ^W Поиск    ^K Вырезать  ^T Выполнить ^C Позиция
^X Выход    ^R ЧитФайл  ^\ Замена   ^U Вставить  ^J Выровнять ^/_ К строке
```

### *Ввод текста из листинга 8.1*

Создаю исполняемый файл и проверяю его работу. (рис. 3).

```
[nvsakhno@fedora lab08]$ nasm -f elf lab8-1.asm
[nvsakhno@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[nvsakhno@fedora lab08]$ ./lab8-1
Введите N: 4
4
3
2
1
[nvsakhno@fedora lab08]$
```

### *Запуск исполняемого файла*

Данная программа выводит числа от N (4) до 1 включительно.

Изменяю текст программы, добавив изменение значения регистра ecx в цикле. (рис. 4).

```
GNU nano 7.2      /home/nvsakhno/work/arch-pc/lab08/lab8-1.asm      Изменён
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:
sub ecx,1 ; `ecx=ecx-1`
mov [N],ecx
mov eax,[N]
call iprintLF
loop labe
call quit
```

*Изменение текста программы*

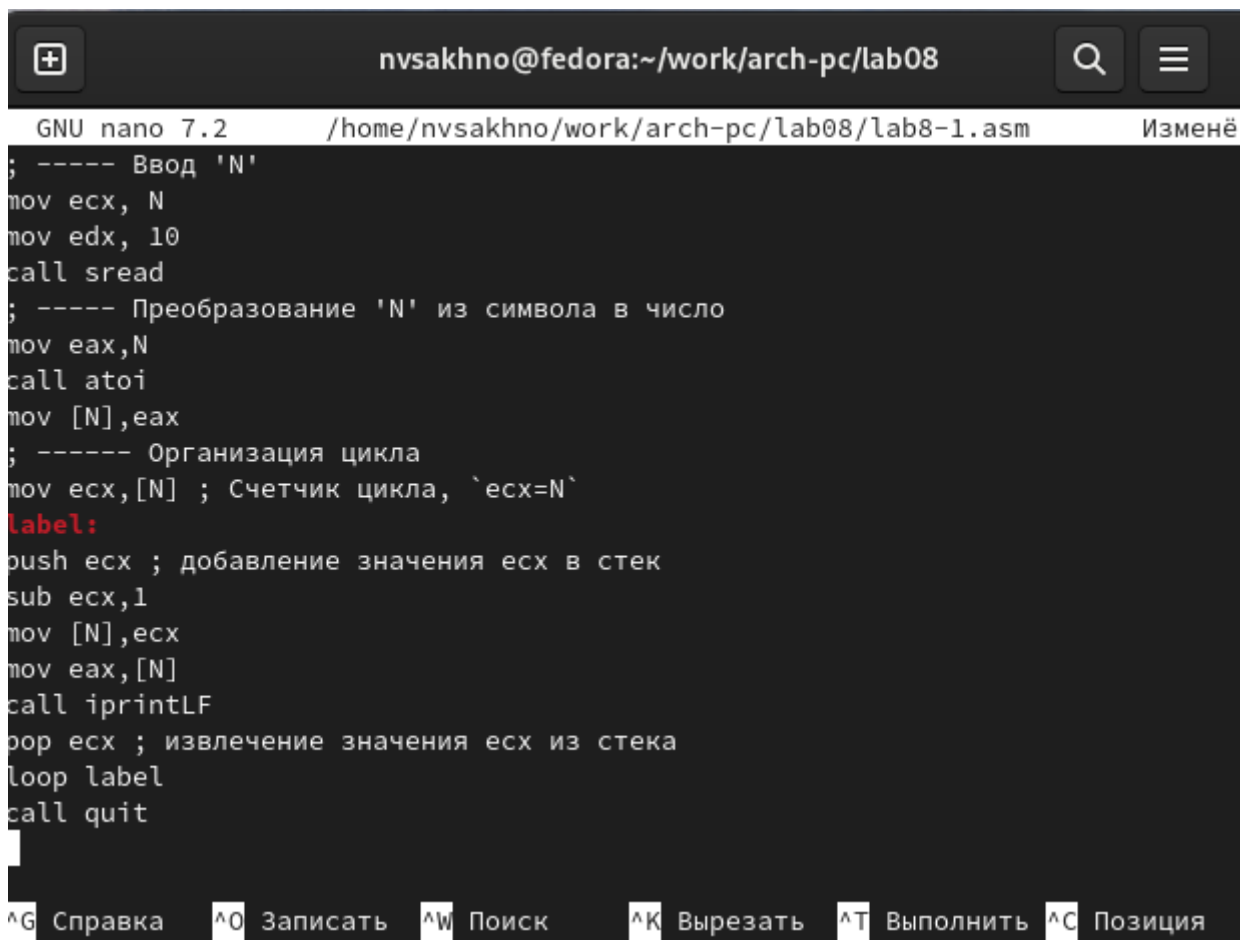
Создаю исполняемый файл и проверяю его работу. (рис. 5).

```
[nvsakhno@fedora lab08]$ nasm -f elf lab8-1.asm
[nvsakhno@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[nvsakhno@fedora lab08]$ ./lab8-1
Введите N: 12
11
9
7
5
3
1
```

*Запуск обновленной программы*

В данном случае число проходов цикла не соответствует введенному с клавиатуры значению.

Вношу изменения в текст программы, добавив команды push и pop для сохранения значения счетчика цикла loop. (рис. 6).

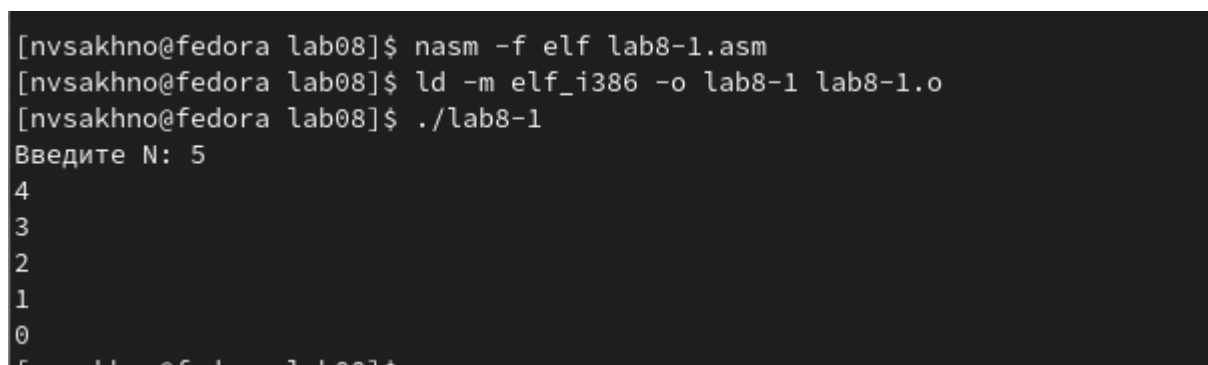


```
GNU nano 7.2 /home/nvsakhno/work/arch-pc/lab08/lab8-1.asm Изменё
; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ----- Преобразование 'N' из символа в число
mov eax, N
call atoi
mov [N], eax
; ----- Организация цикла
mov ecx, [N] ; Счетчик цикла, `ecx=N`
label:
push ecx ; добавление значения ecx в стек
sub ecx, 1
mov [N], ecx
mov eax, [N]
call iprintLF
pop ecx ; извлечение значения ecx из стека
loop label
call quit

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Позиция
```

*Изменение текста программы*

Создаю исполняемый файл и проверяю его работу. (рис. 7).



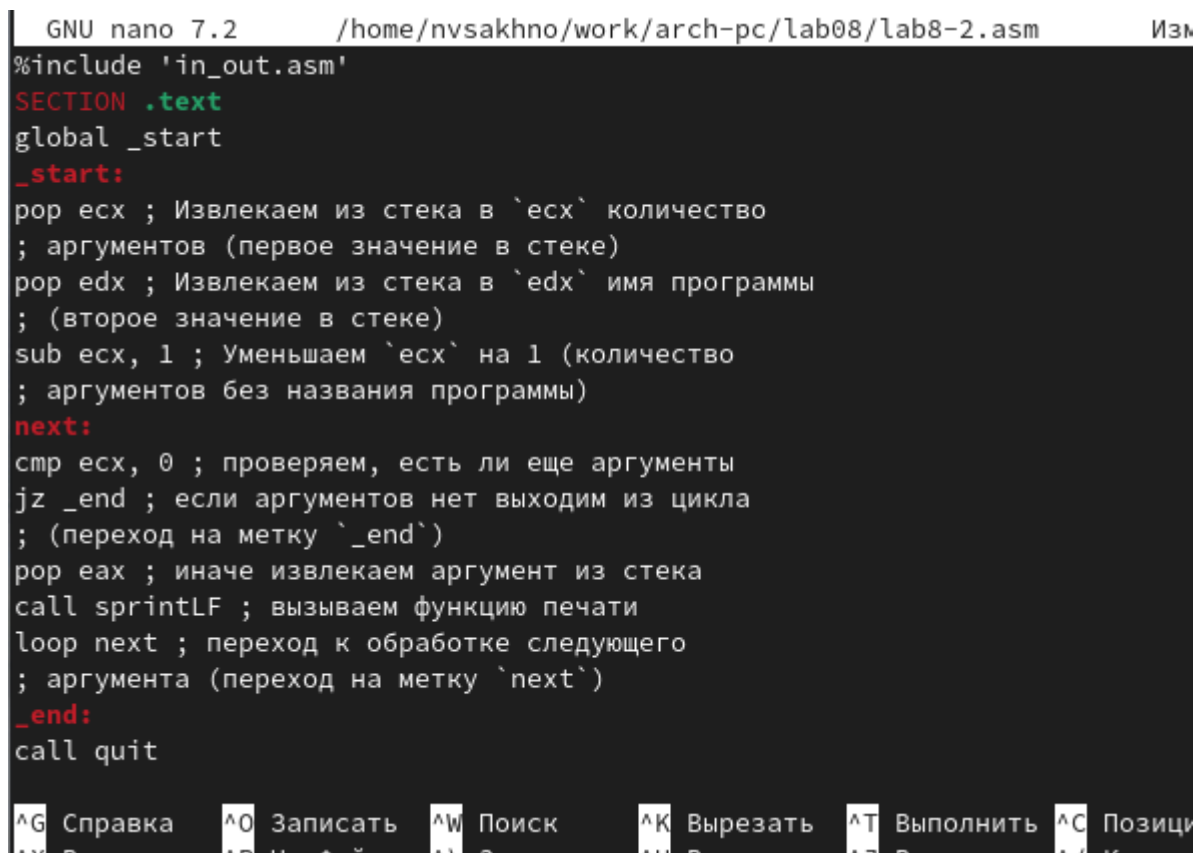
```
[nvsakhno@fedora lab08]$ nasm -f elf lab8-1.asm
[nvsakhno@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[nvsakhno@fedora lab08]$ ./lab8-1
Введите N: 5
4
3
2
1
0
```

*Запуск исполняемого файла*

В данном случае число проходов цикла соответствует введенному с клавиатуры значению и выводит числа от N-1 до 0 включительно.

## Обработка аргументов командной строки

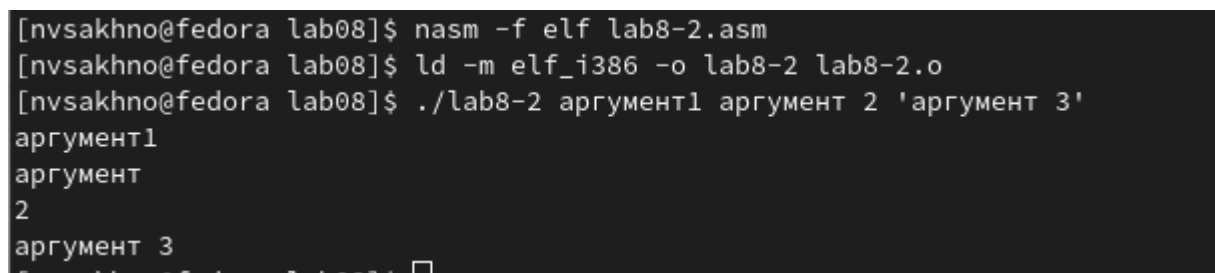
Создаю файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и ввожу в него текст программы из листинга 8.2. (рис. 8).



```
GNU nano 7.2 /home/nvsakhno/work/arch-pc/lab08/lab8-2.asm Изм
%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 программы
             ; (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 sprintf ; вызываем функцию печати
    loop next ; переход к обработке следующего
             ; аргумента (переход на метку `next`)
    _end:
    call quit
```

Ввод текста программы из листинга 8.2

Создаю исполняемый файл и запускаю его, указав нужные аргументы. (рис. 9).



```
[nvsakhno@fedora lab08]$ nasm -f elf lab8-2.asm
[nvsakhno@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[nvsakhno@fedora lab08]$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
```

Запуск исполняемого файла

Программа вывела 4 аргумента, так как аргумент 2 не взят в кавычки, в отличие от аргумента 3, поэтому из-за пробела программа считывает “2” как отдельный аргумент.

Рассмотрим пример программы, которая выводит сумму чисел, которые передаются в программу как аргументы. Создаю файл lab8-3.asm в каталоге ~/work/archpc/lab08 и ввожу в него текст программы из листинга 8.3. (рис. 10).

```
GNU nano 1.2 /home/nvsakhno/work/at-sb-pc/lab08/lab8-3.asm изменен
#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, 0 ; 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
; ...
```

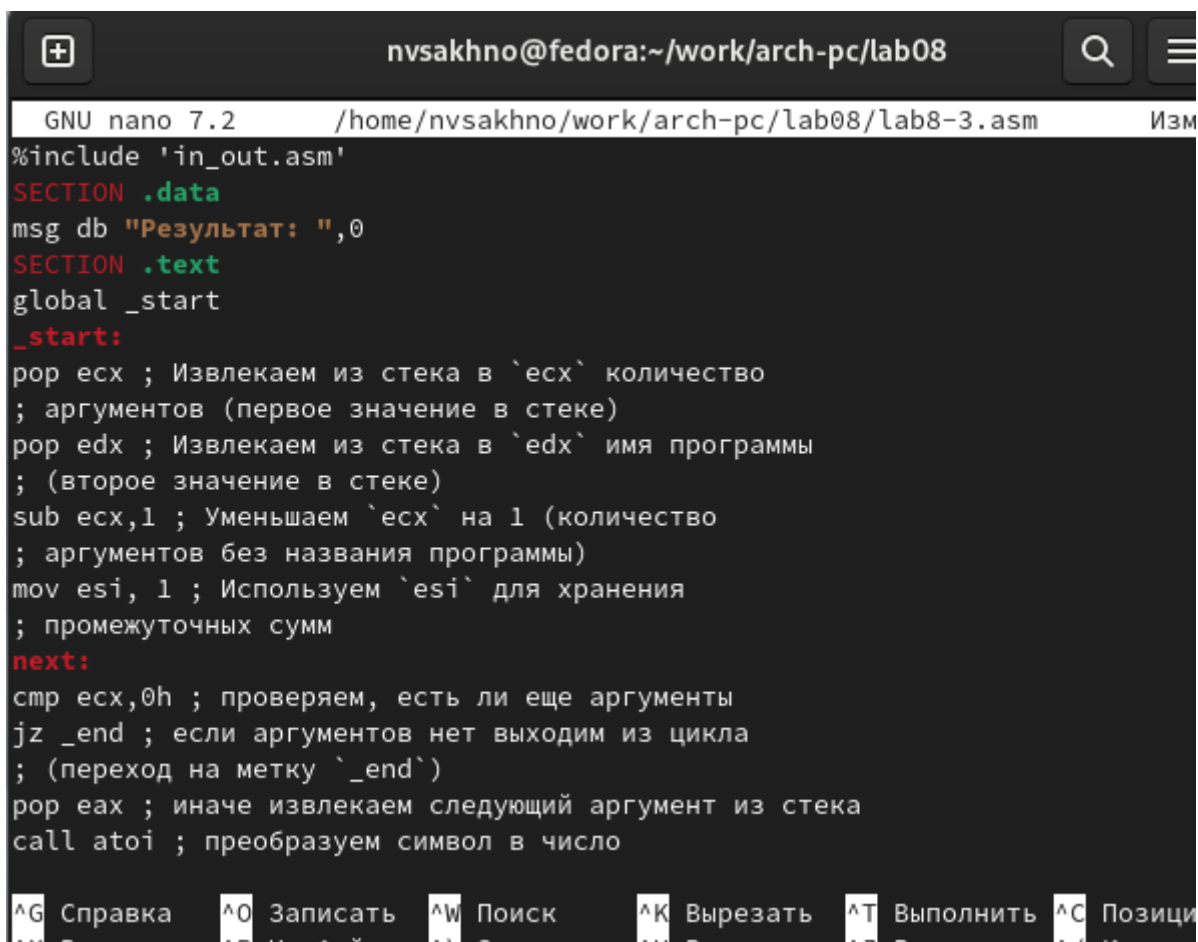
*Ввод текста программы из листинга 8.3*

Создаю исполняемый файл и запускаю его, указав аргументы. (рис. 11).

```
[nvsakhno@fedora lab08]$ ./lab8-3 12 13 7 10 5
Результат: 47
```

*Запуск исполняемого файла*

Изменяю текст программы из листинга 8.3 для вычисления произведения аргументов командной строки. (рис. 12).

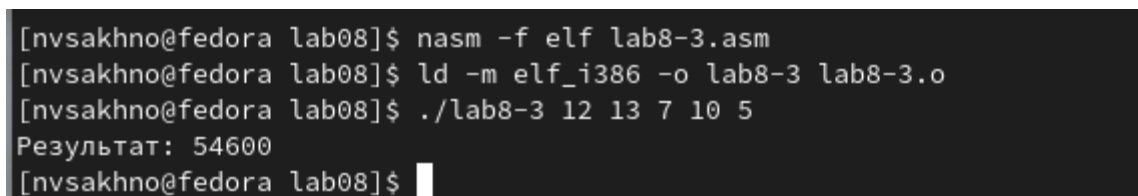


```
GNU nano 7.2 /home/nvsakhno/work/arch-pc/lab08/lab8-3.asm Изм
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
    pop ecx ; Извлекаем из стека в `ecx` количество
    ; аргументов (первое значение в стеке)
    pop edx ; Извлекаем из стека в `edx` имя 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

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Позиции
^X Выход ^D Подсказка ^U Заменить ^N Вставить ^I Выделить ^J Копировать
```

*Изменение текста программы*

Создаю исполняемый файл и запускаю его, указав аргументы. (рис. 13).



```
[nvsakhno@fedora lab08]$ nasm -f elf lab8-3.asm
[nvsakhno@fedora lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o
[nvsakhno@fedora lab08]$ ./lab8-3 12 13 7 10 5
Результат: 54600
[nvsakhno@fedora lab08]$
```

*Запуск исполняемого файла*

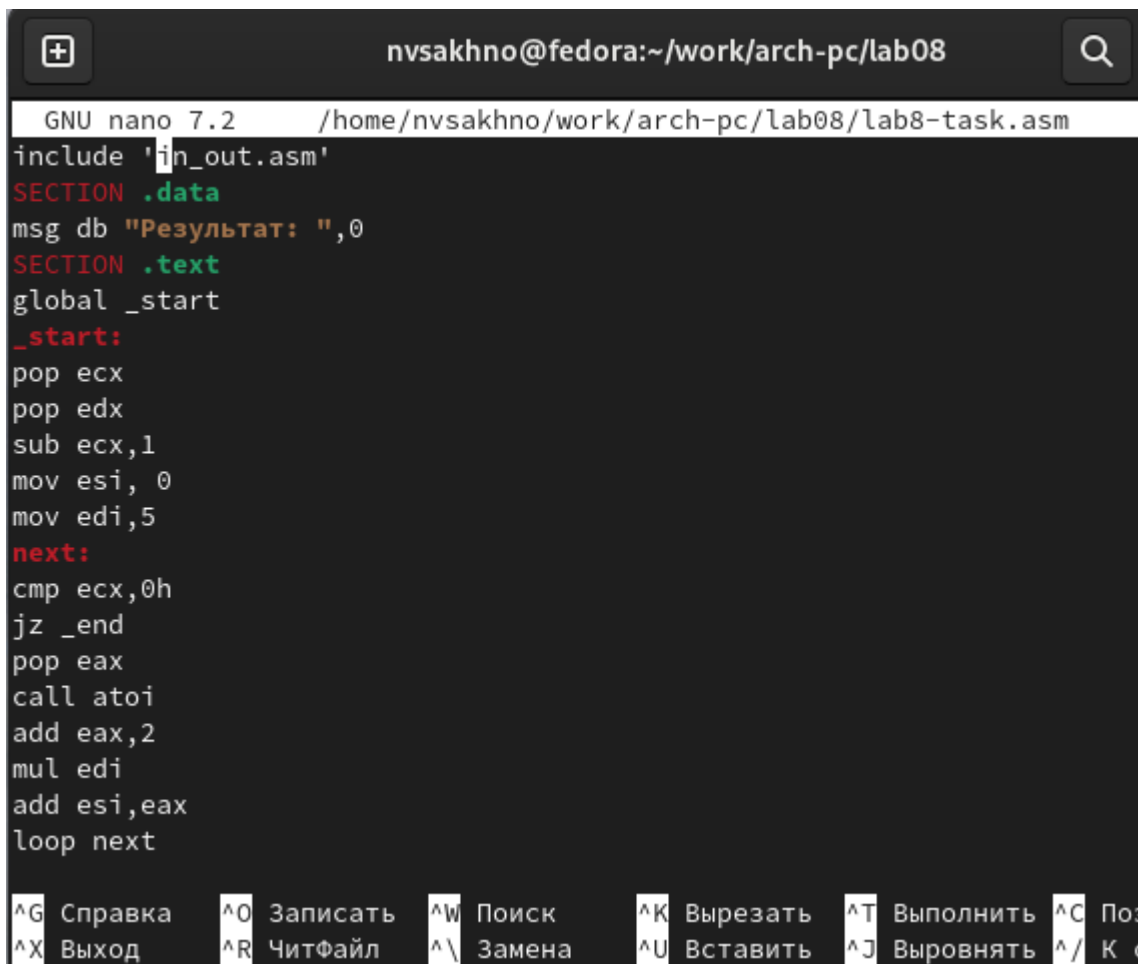
## Задание для самостоятельной работы

У меня вариант 10 (рис. 0)

```
[nvsakhno@fedora lab06]$ ./variant
Введите No студенческого билета:
1132230298
Ваш вариант: 10
[nvsakhno@fedora lab06]$
```

*Вариант из лабораторной работы 6*

Пишу текст программы, которая находит сумму значений функции  $f(x) = 5 \cdot (2 + x)$  в соответствии для  $x = x_1, x_2, \dots, x_n$ . Значения  $x_i$  передаются как аргументы. (рис. 14).



```
GNU nano 7.2 /home/nvsakhno/work/arch-pc/lab08/lab8-task.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
mov edi,5
next:
cmp ecx,0h
jz _end
pop eax
call atoi
add eax,2
mul edi
add esi,eax
loop next

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Поз
^X Выход ^R Читфайл ^\ Замена ^U Вставить ^J Выровнять ^/ К с
```

*Текст программы*



Создаю исполняемый файл и проверьте его работу на нескольких наборах  $x = x_1, x_2, \dots, x_n$ . (рис. 15).

```
[nvsakhno@fedora lab08]$ nasm -f elf task.asm
[nvsakhno@fedora lab08]$ ld -m elf_i386 -o task task.o
[nvsakhno@fedora lab08]$ ./task 1 2 3
Результат: 60
[nvsakhno@fedora lab08]$ ./ task 4 6 8 9
bash: ./: Это каталог
[nvsakhno@fedora lab08]$ ./task 4 6 8 9
Результат: 175
[nvsakhno@fedora lab08]$ ./task 10 15 33 21
Результат: 435
[nvsakhno@fedora lab08]$
```

*Запуск исполняемого файла и проверка его работы*

Программа работает корректно.

## Выводы

Благодаря данной лабораторной работе я приобрела навыки написания программ использованием циклов и обработкой аргументов командной строки, что поможет мне при выполнении последующих лабораторных работ.

## Список литературы

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.