

Отчёт по лабораторной работе №8

дисциплина: Архитектура компьютера

Люкшина Влада Алексеевна

Содержание

1)Цель работы	5
2)Задание	6
3)Выполнение лабораторной работы	7
3.1) Создаем каталог для программ лабораторной работы № 8, переходим в него и создаем файл lab8-1.asm:	7
3.2) Вводим в файл lab8-1.asm текст программы из листинга 8.1.	8
3.3) Создаем исполняемый файл и проверяем его работу.	9
3.4) Изменяем текст программы, добавив изменение значение регистра еsx в цикле.	10
3.5) Создаем исполняемый файл и проверяем его работу.	11
3.6) Вносим изменения в текст программы добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop.	12
3.7) Создаем исполняемый файл и проверяем его работу.	13
3.8) Создаем файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и вводим в него текст программы из листинга 8.2.	14
3.9) Создаем исполняемый файл и проверяем его работу, указав аргументы.	15
3.10) Изменяем текст программы из листинга 8.3 для вычисления произведения аргументов командной строки.	16
3.11) Создаем исполняемый файл и проверяем его работу, указав аргументы.	17
4) Самостоятельная работа	18
4.2) Пишем программу, которая находит сумму значений функции(вариант 3, $10x-5$)	19
4.3) Запускаем программу и вводим аргументы.	20
5)Выводы	21

Список иллюстраций

1	Создаем каталог, переходим в него и создаем файл	7
2	Вводим текст в файл	8
3	Создаем и запускаем файл	9
4	Изменяем текст программы	10
5	Создаем и запускаем файл	11
6	Изменяем текст программы	12
7	Создаем файл и вводим текст	14
8	Создаем и запускаем файл	15
9	Изменяем текст программы	16
10	Создаем и запускаем файл	17
1	Пишем программу	19
2	Запускаем программу	20

Список таблиц

1)Цель работы

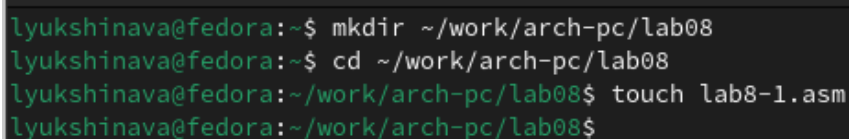
Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

2)Задание

Написать программы с использованием циклов и обработкой аргументов командной строки.

3)Выполнение лабораторной работы

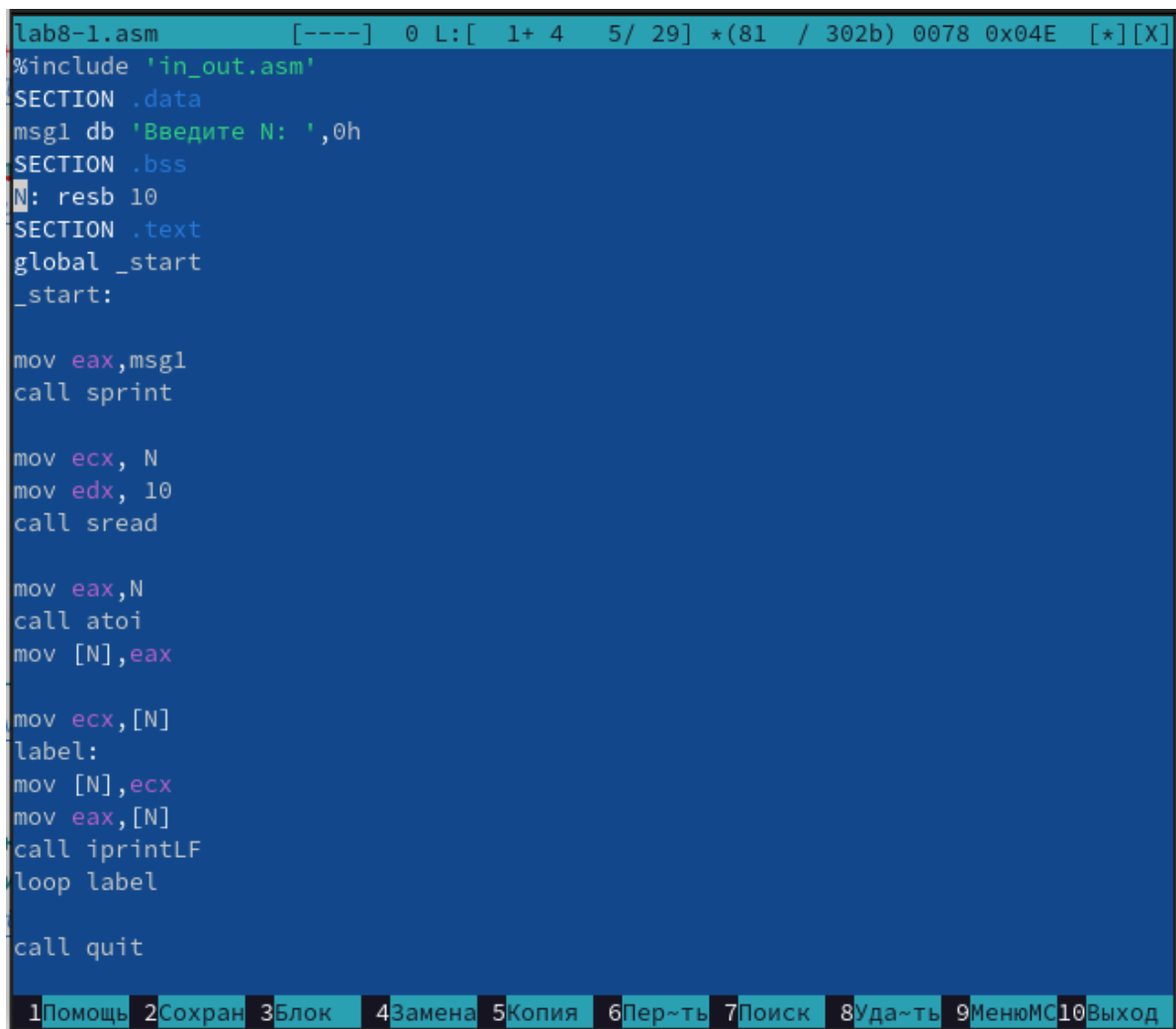
3.1) Создаем каталог для программ лабораторной работы № 8, переходим в него и создаем файл lab8-1.asm:



```
lyukshinava@fedora:~$ mkdir ~/work/arch-pc/lab08
lyukshinava@fedora:~$ cd ~/work/arch-pc/lab08
lyukshinava@fedora:~/work/arch-pc/lab08$ touch lab8-1.asm
lyukshinava@fedora:~/work/arch-pc/lab08$
```

Рис. 1: Создаем каталог, переходим в него и создаем файл

3.2) Вводим в файл lab8-1.asm текст программы из листинга 8.1.



```
lab8-1.asm [----] 0 L: [ 1+ 4 5/ 29] *(81 / 302b) 0078 0x04E [*] [X]
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_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
```

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюMC10Выход

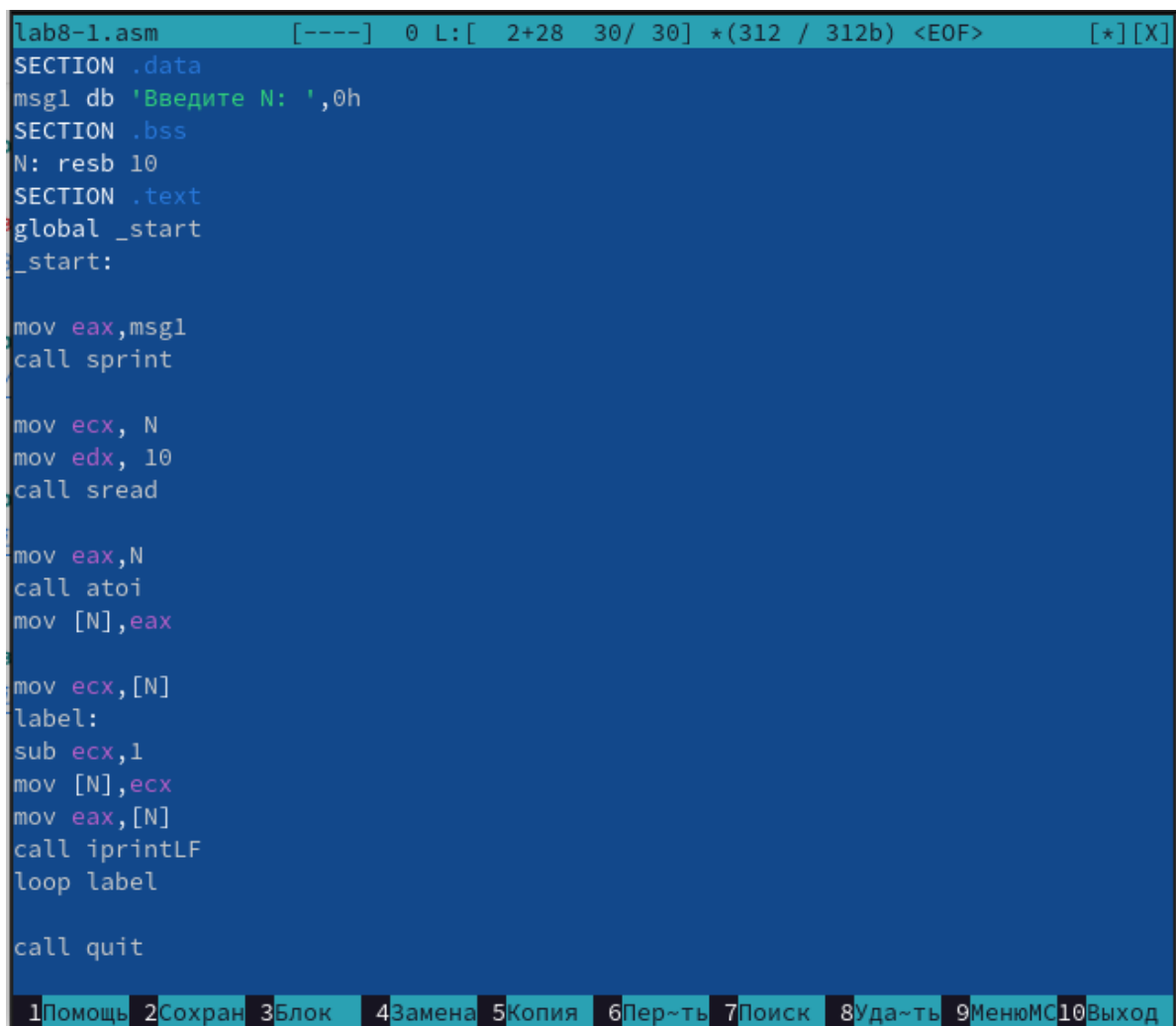
Рис. 2: Вводим текст в файл

3.3) Создаем исполняемый файл и проверяем его работу.

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

Рис. 3: Создаем и запускаем файл

3.4) Изменяем текст программы, добавив изменение значение регистра ecx в цикле.



```
lab8-1.asm [----] 0 L:[ 2+28 30/ 30] *(312 / 312b) <EOF> [*] [X]
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_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
```

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюMC10Выход

Рис. 4: Изменяем текст программы

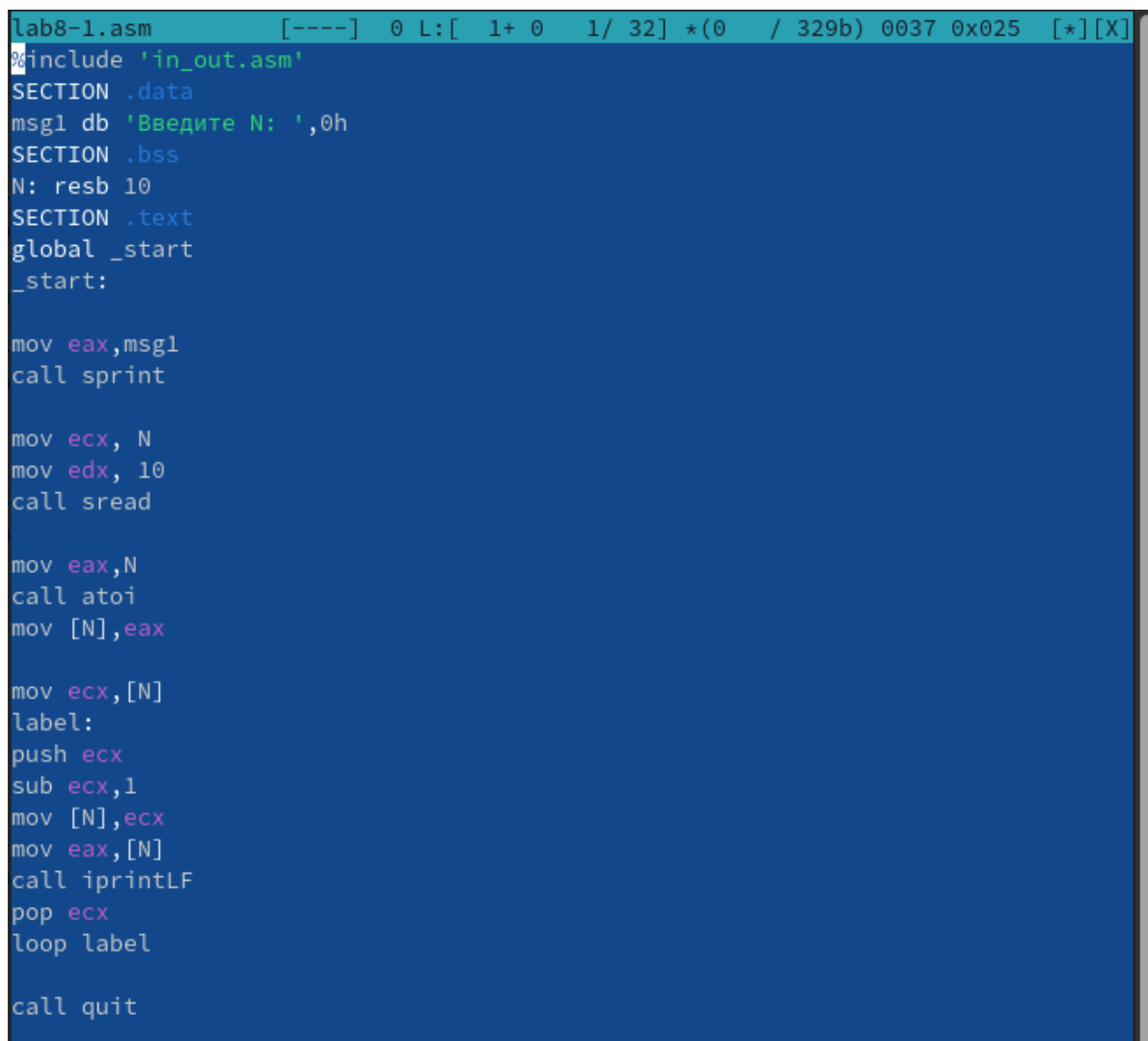
3.5) Создаем исполняемый файл и проверяем его работу.

```
lyukshinava@fedora:~$ cd ~/work/arch-pc/lab08
lyukshinava@fedora:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
7
5
3
1
lyukshinava@fedora:~/work/arch-pc/lab08$
```

Рис. 5: Создаем и запускаем файл

Регистр `ecx` принимает значения 9,7,5,3,1(исходное число 10, в цикле данный регистр уменьшается на 2 командой `sub` и `loop`). Число проходов цикла не соответствует числу `N`, так как уменьшается на 2.

3.6) Вносим изменения в текст программы добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop.



```
lab8-1.asm  [----]  0 L:[ 1+ 0  1/ 32] *(0  / 329b) 0037 0x025  [*][X]
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_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:
push ecx
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
pop ecx
loop label

call quit
```

Рис. 6: Изменяем текст программы

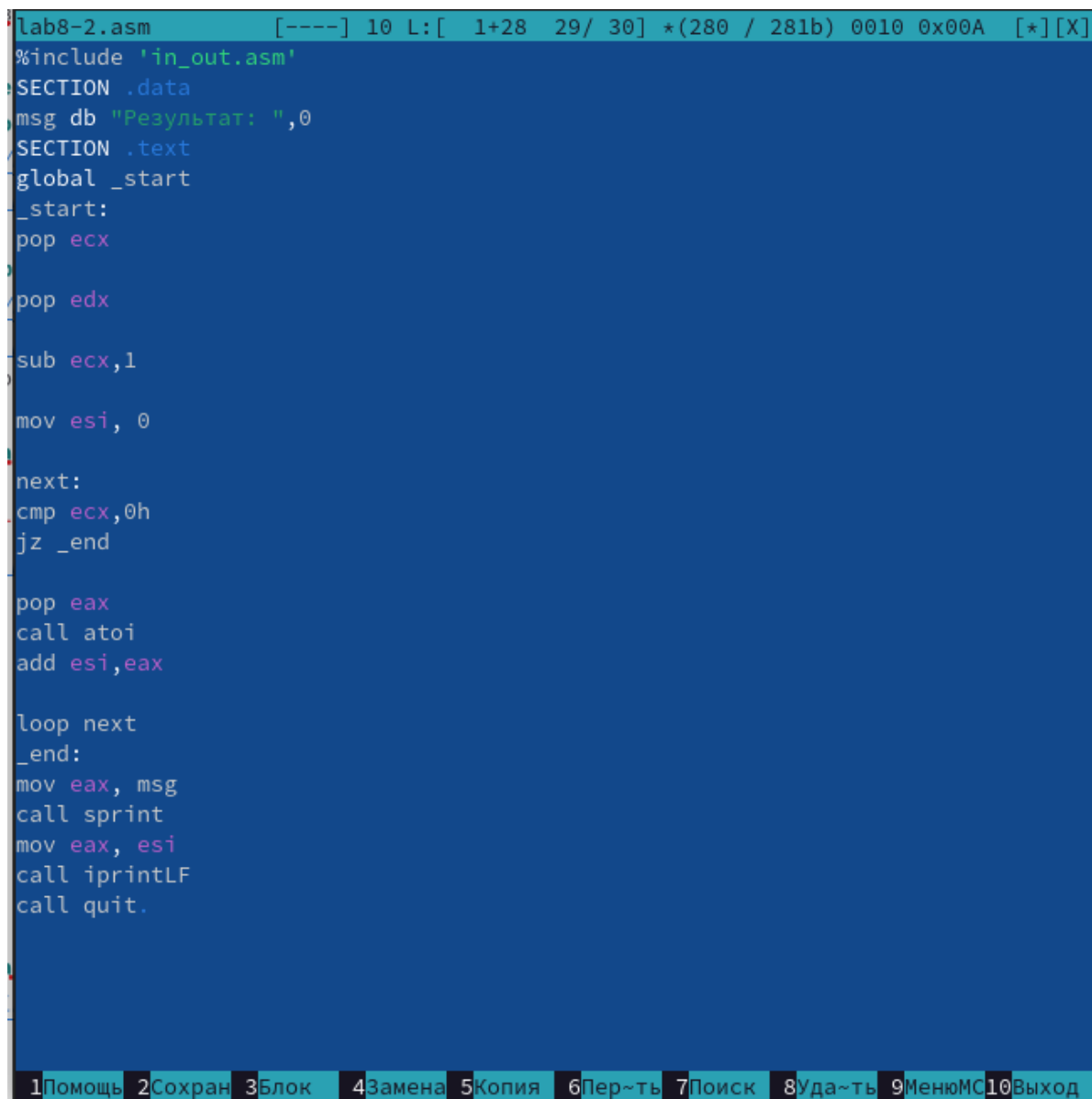
3.7) Создаем исполняемый файл и проверяем его работу.

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

В данном случае число проходов цикла равна числу N.

3.8) Создаем файл lab8-2.asm в каталоге

~/work/arch-рс/lab08 и вводим в него текст программы из листинга 8.2.



```
lab8-2.asm [----] 10 L: [ 1+28 29/ 30] *(280 / 281b) 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
call atoi
add 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 9МенюМС10Выход

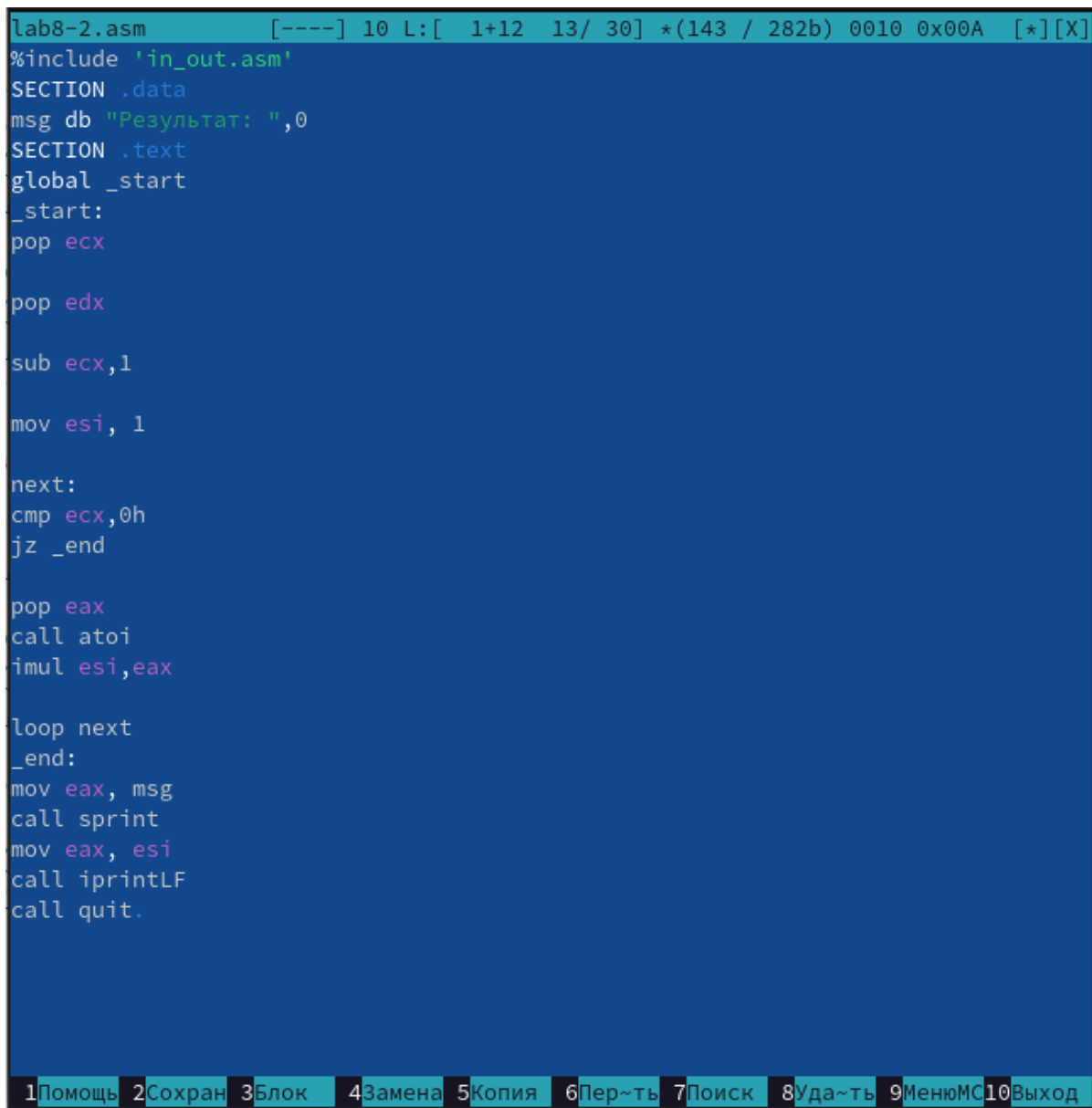
Рис. 7: Создаем файл и вводим текст

3.9) Создаем исполняемый файл и проверяем его работу, указав аргументы.

```
lyukshinava@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
lyukshinava@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
lyukshinava@fedora:~/work/arch-pc/lab08$ ./lab8-2 12 13 7 10 5
Результат: 47
lyukshinava@fedora:~/work/arch-pc/lab08$
```

Рис. 8: Создаем и запускаем файл

3.10) Изменяем текст программы из листинга 8.3 для вычисления произведения аргументов командной строки.



```
lab8-2.asm [----] 10 L: [ 1+12 13/ 30] *(143 / 282b) 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, 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 9МенюМС10Выход

Рис. 9: Изменяем текст программы

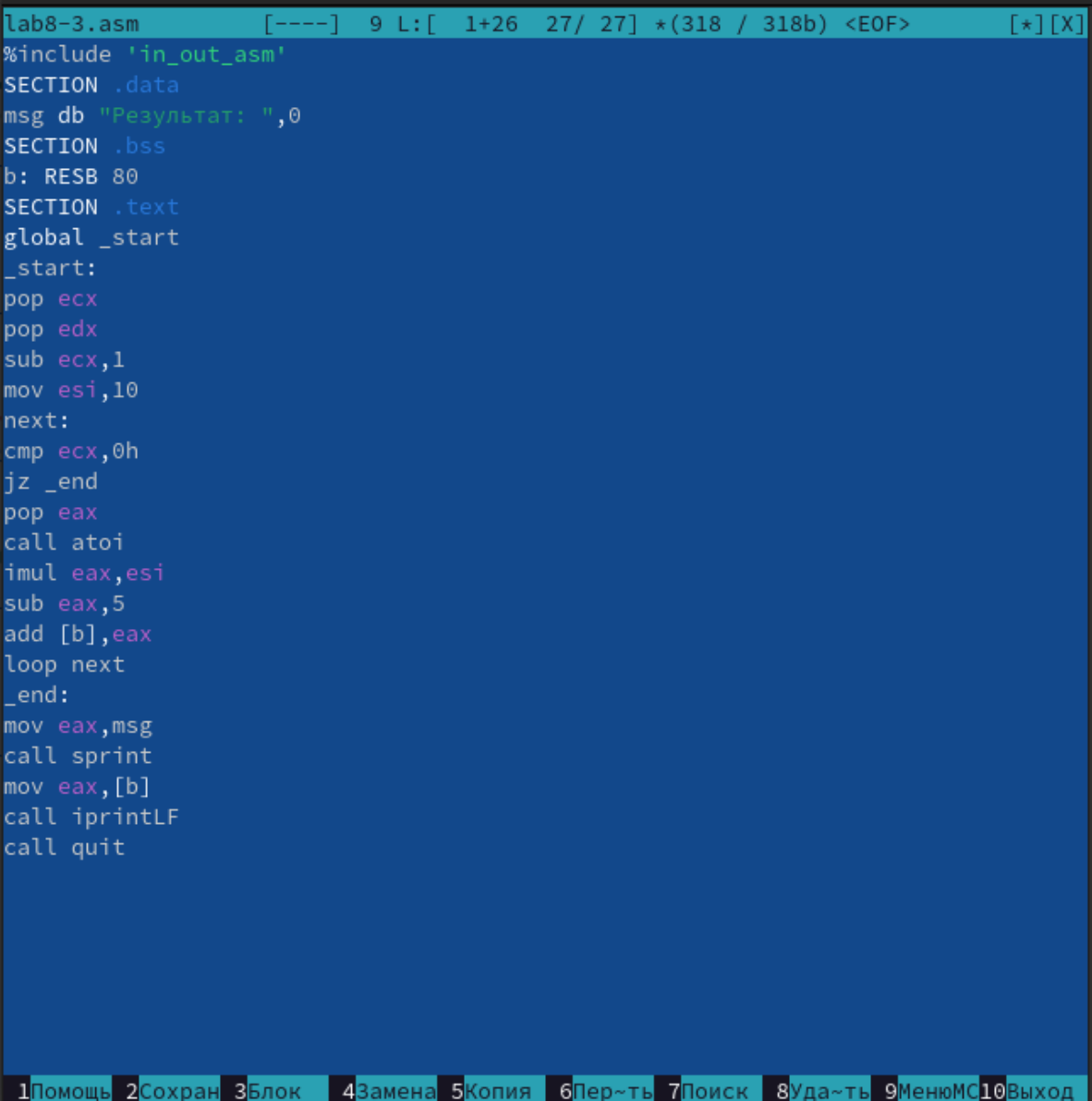
3.11) Создаем исполняемый файл и проверяем его работу, указав аргументы.

```
lyukshinava@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
lyukshinava@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
lyukshinava@fedora:~/work/arch-pc/lab08$ ./lab8-2 2 3 4
bash: ./lab8-2: Нет такого файла или каталога
lyukshinava@fedora:~/work/arch-pc/lab08$ ./lab8-2 2 3 4
Результат: 24
lyukshinava@fedora:~/work/arch-pc/lab08$
```

Рис. 10: Создаем и запускаем файл

4) Самостоятельная работа

4.2) Пишем программу, которая находит сумму значений функции(вариант 3, 10х-5)



```
lab8-3.asm      [----]  9  L:[ 1+26 27/ 27] *(318 / 318b) <EOF>  [*] [X]
%include 'in_out_asm'
SECTION .data
msg db "Результат: ",0
SECTION .bss
b: RESB 80
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx,1
mov esi,10
next:
cmp ecx,0h
jz _end
pop eax
call atoi
imul eax,esi
sub eax,5
add [b],eax
loop next
_end:
mov eax,msg
call sprint
mov eax,[b]
call iprintLF
call quit
```

1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер~ть 7Поиск 8Уда~ть 9МенюМС10Выход

Рис. 1: Пишем программу

4.3) Запускаем программу и вводим аргументы.

```
lyukshinava@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
lyukshinava@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
lyukshinava@fedora:~/work/arch-pc/lab08$ ./lab8-3 1 2 3
Результат: 45
lyukshinava@fedora:~/work/arch-pc/lab08$
```

Рис. 2: Запускаем программу

5)Выводы

Мы научились писать программы с использованием циклов и обработкой аргументов командной строки.