

# **Отчёт по лабораторной работе №6**

**дисциплина: Архитектура компьютера**

Люкшина Влада Алексеевна

# Содержание

<b>1)Цель работы</b>	<b>5</b>
<b>2)Задание</b>	<b>6</b>
<b>3)Выполнение лабораторной работы</b>	<b>7</b>
3.1) Создаем каталог для программ лабораторной работы № 6, переходим в него и создаем файл lab6-1.asm . . . . .	7
3.2) Вводим в файл lab6-1.asm текст программы из листинга 6.1. . . . .	8
3.3) Создаем файл и запускаем его. . . . .	8
3.4) Изменяем текст программы и вместо символов записываем в регистры числа. . . . .	9
3.5) Создаем исполняемый файл и запускаем его . . . . .	9
3.6) Создаем файл lab6-2.asm в каталоге ~/work/arch-pc/lab06. . . . .	10
3.7) Вводим в него текст программы из листинга 6.2. . . . .	10
3.8) Создаем исполняемый файл и запускаем его. . . . .	11
3.9) Аналогично предыдущему примеру изменяем символы на числа. .	11
3.10) Создаем исполняемый файл и запускаем его. . . . .	12
3.11) Заменяем функцию iprintLF на iprint. . . . .	12
3.12) Создаем исполняемый файл и запускаем его. . . . .	13
3.13) Создаем файл lab6-3.asm в каталоге ~/work/arch-pc/lab06. . . . .	13
3.14) Вводим в lab6-3.asm текст из листинга 6.3. . . . .	14
3.15) Создаем исполняемый файл и запускаем его. . . . .	14
3.16) Изменяем текст программы для вычисления выражения. . . . .	15
3.17) Создаем исполняемый файл и проверяем его работу. . . . .	15
3.18) Создаем файл variant.asm в каталоге ~/work/arch-pc/lab06 . . . . .	16
3.19) Вводим текст из листинга 6.4 в файл variant.asm. . . . .	16
3.20) Создаем исполняемый файл и запускаем его. . . . .	17
3.21) Ответы на вопросы . . . . .	17
<b>4) Самостоятельная работа</b>	<b>19</b>
4.1) Создаем файл в каталоге . . . . .	19
4.2) Пишем программу для вычисления варианта №3 $(x + 2)^2$ . . . . .	20
4.3) Создаем файл и запускаем его . . . . .	21
<b>5)Выводы</b>	<b>22</b>

# Список иллюстраций

1	Создаем файл и каталог . . . . .	7
2	Вводим текст . . . . .	8
3	Создаем файл и запускаем . . . . .	8
4	Изменяем текст . . . . .	9
5	Запускаем файл . . . . .	9
6	Создаем файл . . . . .	10
7	Вводим текст . . . . .	10
8	Создаем и запускаем . . . . .	11
9	Редактируем файл . . . . .	11
10	Создаем и запускаем файл . . . . .	12
11	Редактируем файл . . . . .	12
12	Создаем и запускаем . . . . .	13
13	Создаем файл . . . . .	13
14	Вводим текст . . . . .	14
15	Создаем и запускаем файл . . . . .	14
16	Редактируем файл . . . . .	15
17	Создаем и проверяем . . . . .	15
18	Создаем файл . . . . .	16
19	Вводим текст . . . . .	16
20	Создаем и запускаем . . . . .	17
1	Создаем файл . . . . .	19
2	Пишем программу . . . . .	20
3	Создаем файл и запускаем . . . . .	21

## **Список таблиц**

# **1)Цель работы**

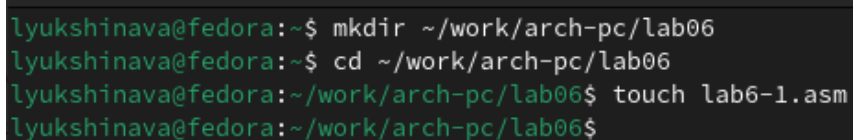
Освоение арифметических инструкций языка ассемблера NASM.

## **2)Задание**

Написать программы для вычисления арифметических выражений.

## 3)Выполнение лабораторной работы

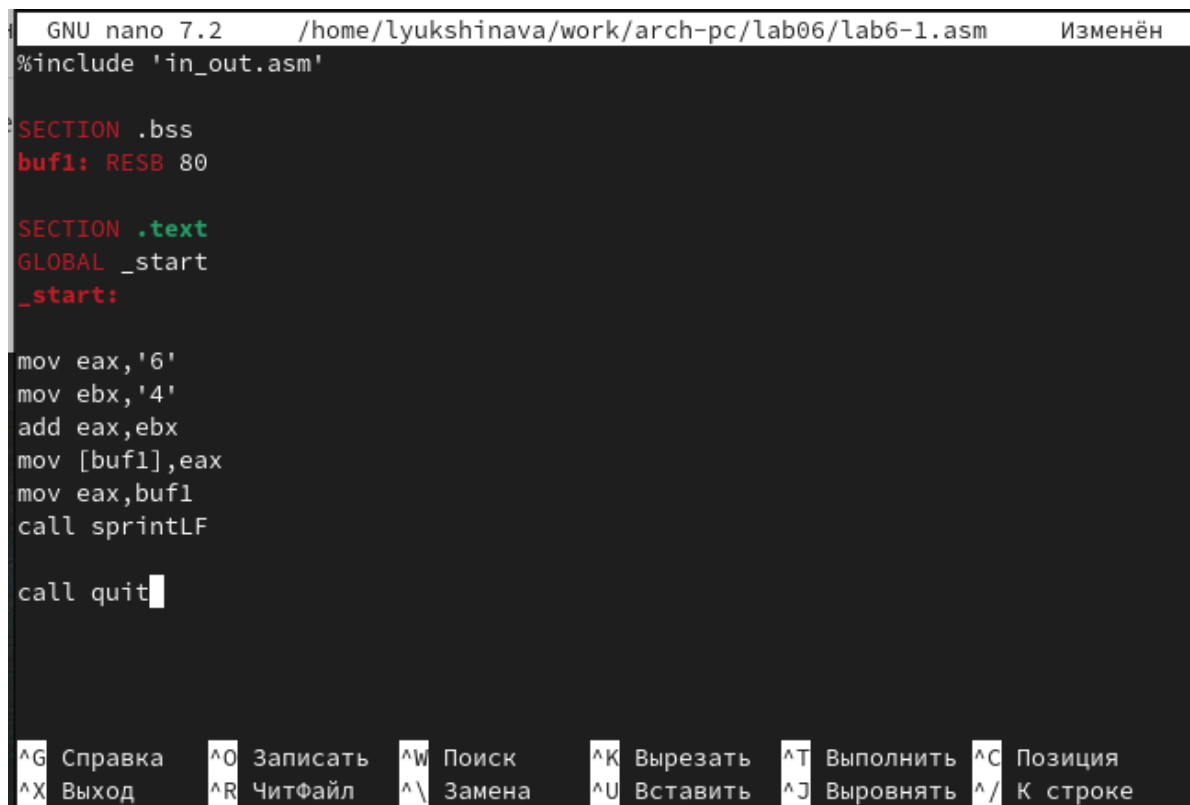
**3.1) Создаем каталог для программ лабораторной работы № 6, переходим в него и создаем файл lab6-1.asm**



```
lyukshinava@fedora:~$ mkdir ~/work/arch-pc/lab06
lyukshinava@fedora:~$ cd ~/work/arch-pc/lab06
lyukshinava@fedora:~/work/arch-pc/lab06$ touch lab6-1.asm
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 1: Создаем файл и каталог

### 3.2) Вводим в файл lab6-1.asm текст программы из листинга 6.1.



```
GNU nano 7.2 /home/lyukshinava/work/arch-pc/lab06/lab6-1.asm  Изменён
#include 'in_out.asm'

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

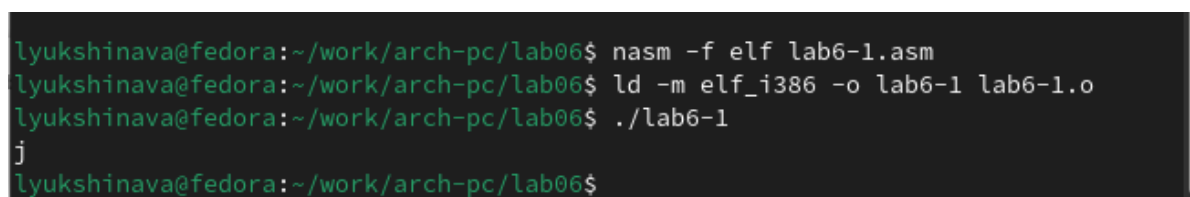
mov eax,'6'
mov ebx,'4'
add eax,ebx
mov [buf1],eax
mov eax,buf1
call printf

call _exit
```

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

Рис. 2: Вводим текст

### 3.3) Создаем файл и запускаем его.



```
lyukshinava@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
lyukshinava@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-1
j
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 3: Создаем файл и запускаем



### 3.4) Изменяем текст программы и вместо символов записываем в регистры числа.



```
GNU nano 7.2 /home/lyukshinava/work/arch-pc/lab06/lab6-1.asm  Изменён
#include 'in_out.asm'

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

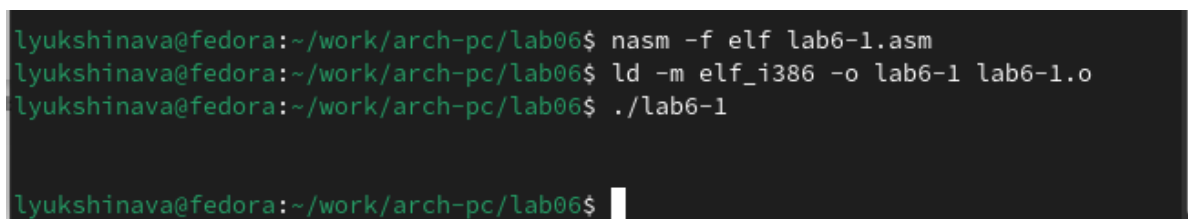
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF

call quit
```

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

Рис. 4: Изменяем текст

### 3.5) Создаем исполняемый файл и запускаем его



```
lyukshinava@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
lyukshinava@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-1

lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 5: Запускаем файл

### 3.6) Создаем файл lab6-2.asm в каталоге ~/work/arch-pc/lab06.

```
lyukshinava@fedora:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-2.asm  
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 6: Создаем файл

### 3.7) Вводим в него текст программы из листинга 6.2.

```
lab6-2.asm [-----] 0 L: [ 1+ 0 1/ 10] *(0 / 118b) 0037 0x025 [*][X]  
%include 'in_out.asm'  
SECTION .text  
GLOBAL _start  
_start:  
mov eax, '6'  
mov ebx, '4'  
add eax, ebx  
call iprintLF  
call quit
```

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

Рис. 7: Вводим текст

### 3.8) Создаем исполняемый файл и запускаем его.

```
lyukshinava@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
lyukshinava@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-2
106
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 8: Создаем и запускаем

### 3.9) Аналогично предыдущему примеру изменяем символы на числа.

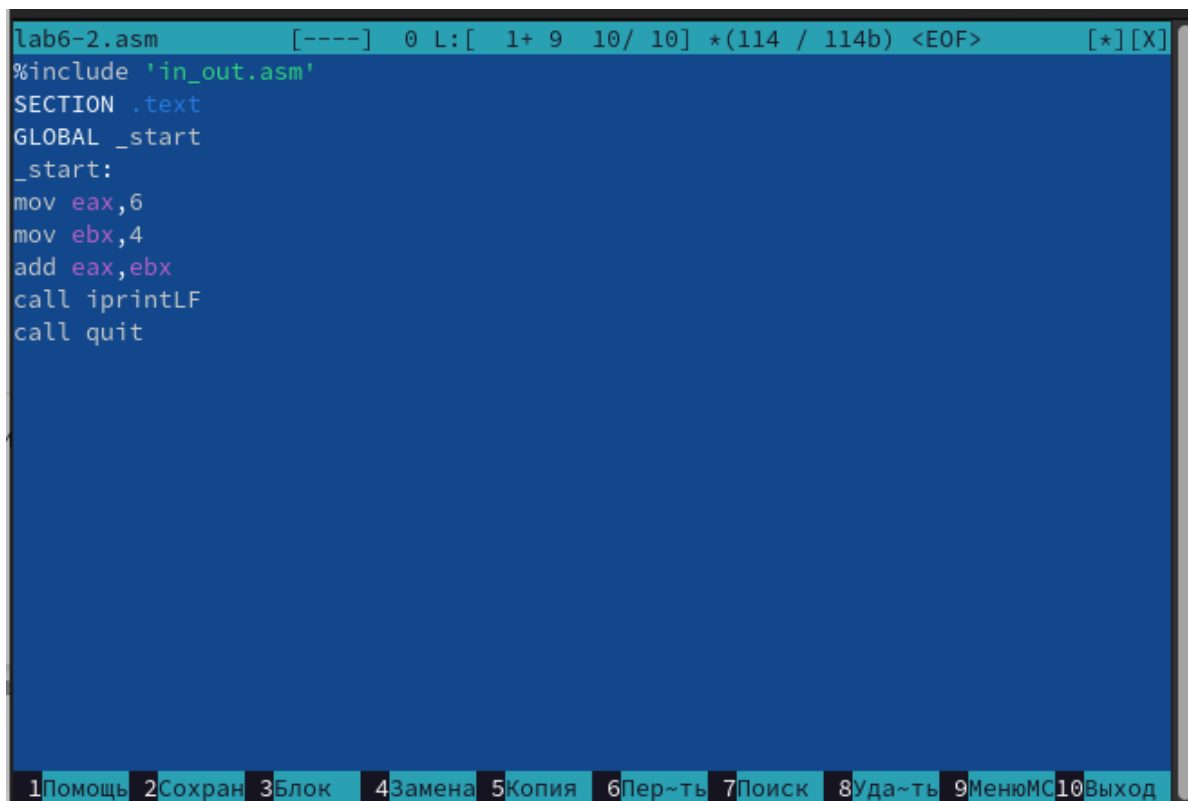


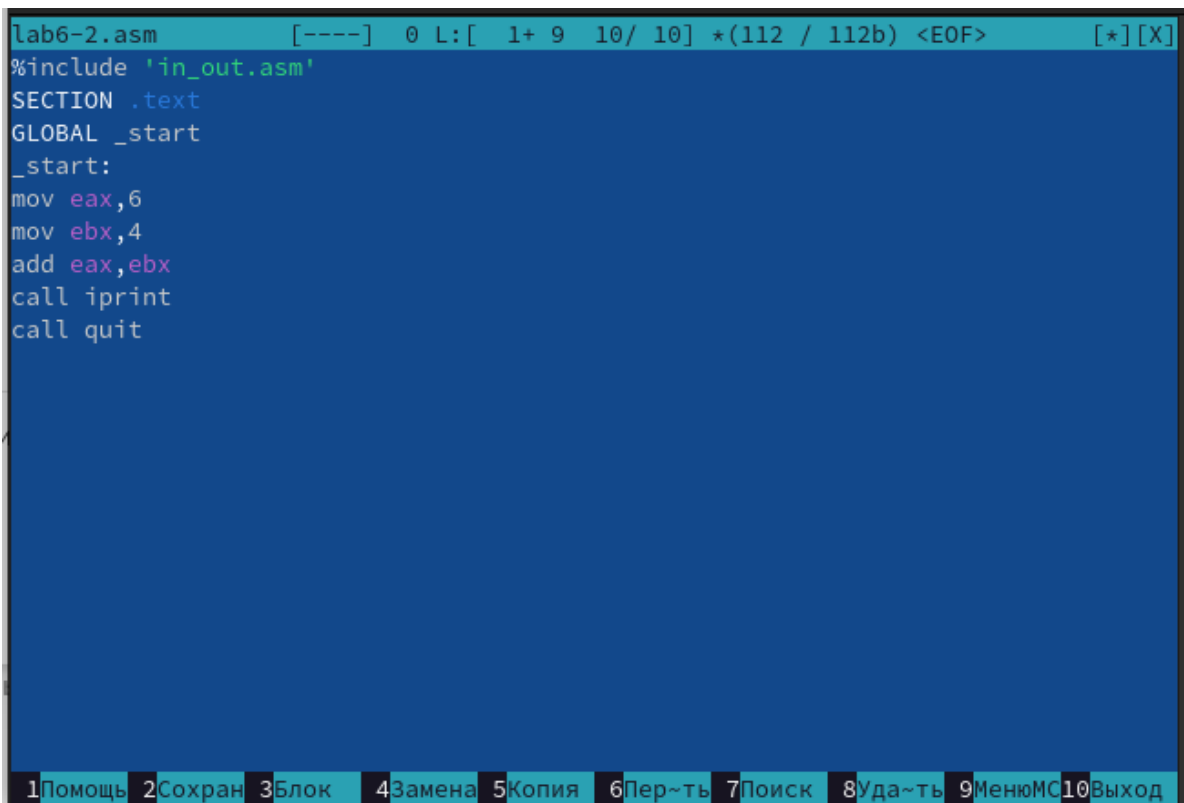
Рис. 9: Редактируем файл

### 3.10) Создаем исполняемый файл и запускаем его.

```
lyukshinava@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
lyukshinava@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-2
10
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 10: Создаем и запускаем файл

### 3.11) Заменяем функцию iprintLF на iprint.



```
lab6-2.asm [----] 0 L: [ 1+ 9 10/ 10] *(112 / 112b) <EOF> [*] [X]
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

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

Рис. 11: Редактируем файл

### 3.12) Создаем исполняемый файл и запускаем его.

```
lyukshinava@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
lyukshinava@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-2
10lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 12: Создаем и запускаем

### 3.13) Создаем файл lab6-3.asm в каталоге ~/work/arch-pc/lab06.

```
10lyukshinava@fedora:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 13: Создаем файл

### 3.14) Вводим в lab6-3.asm текст из листинга 6.3.



```
lab6-3.asm [-M--] 17 L:[ 1+ 0 1/ 27] *(17 / 346b) 0097 0x061 [*] [X]
%include 'in_out.asm'
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:

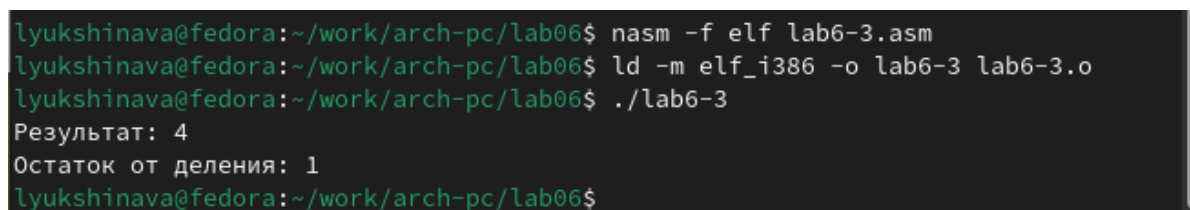
mov eax,5
mov ebx,2
mul ebx
add eax,3
xor edx,edx
mov ebx,3
div ebx
mov edi,eax

mov eax,div
call sprint
mov eax,edi
call iprintLF
mov eax,rem

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

Рис. 14: Вводим текст

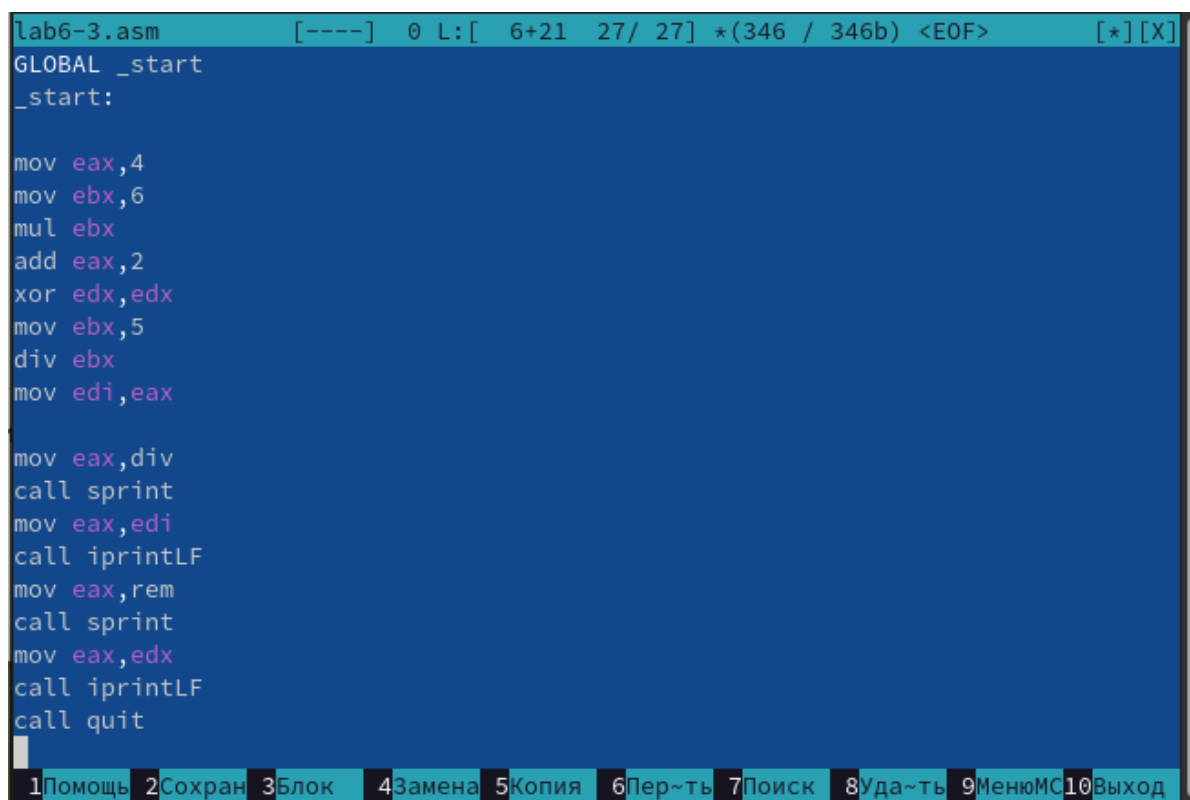
### 3.15) Создаем исполняемый файл и запускаем его.



```
lyukshinava@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
lyukshinava@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 15: Создаем и запускаем файл

### 3.16) Изменяем текст программы для вычисления выражения.



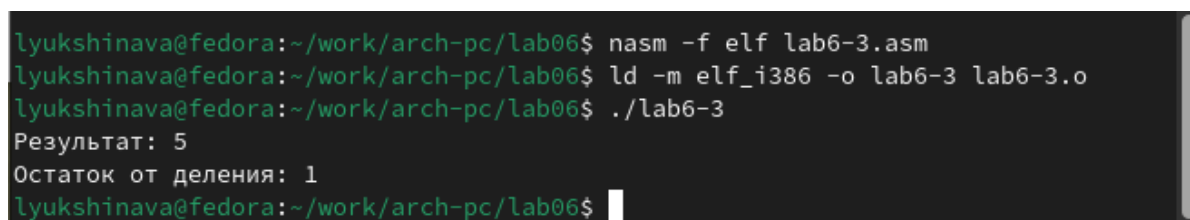
```
lab6-3.asm  [----]  0  L:[  6+21  27/ 27]  *(346 / 346b)  <EOF>  [*] [X]
GLOBAL _start
_start:

mov  eax,4
mov  ebx,6
mul  ebx
add  eax,2
xor  edx,edx
mov  ebx,5
div  ebx
mov  edi,eax

mov  eax,div
call sprint
mov  eax,edi
call iprintLF
mov  eax,rem
call sprint
mov  eax,edx
call iprintLF
call quit
```

Рис. 16: Редактируем файл

### 3.17) Создаем исполняемый файл и проверяем его работу.



```
lyukshinava@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
lyukshinava@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 17: Создаем и проверяем

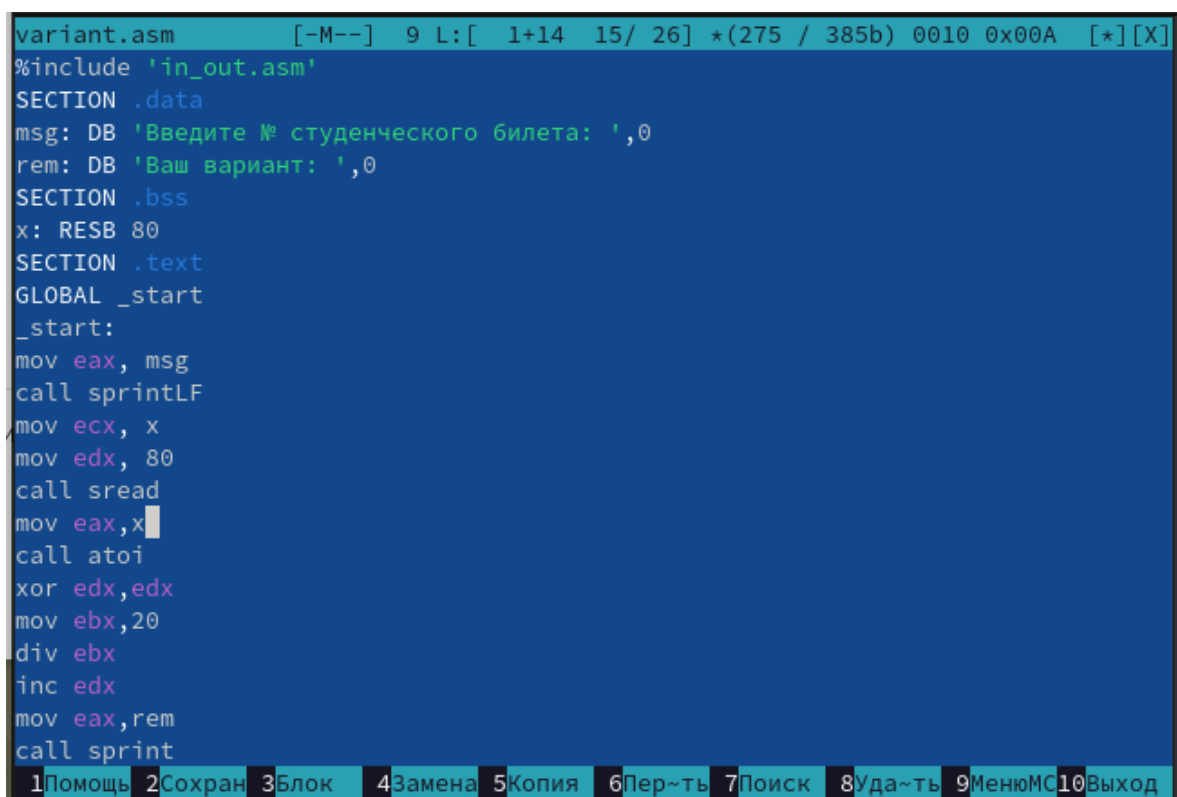
### 3.18) Создаем файл variant.asm в каталоге

~/work/arch-pc/lab06

```
lyukshinava@fedora:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
```

Рис. 18: Создаем файл

### 3.19) Вводим текст из листинга 6.4 в файл variant.asm.



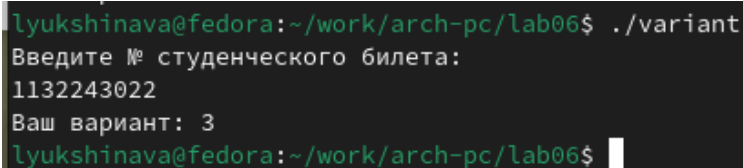
```
variant.asm [-M--] 9 L:[ 1+14 15/ 26] *(275 / 385b) 0010 0x00A [*][X]
#include 'in_out.asm'
SECTION .data
msg: DB 'Введите № студенческого билета: ',0
rem: DB 'Ваш вариант: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintLF
mov ecx, x
mov edx, 80
call sread
mov eax, x
call atoi
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprint
```

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

Рис. 19: Вводим текст



### 3.20) Создаем исполняемый файл и запускаем его.



```
lyukshinava@fedora:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132243022
Ваш вариант: 3
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 20: Создаем и запускаем

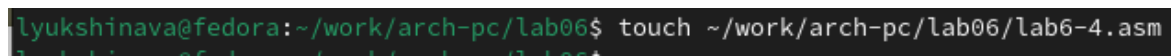
### 3.21) Ответы на вопросы

1. За вывод на экран сообщения 'Ваш вариант:' отвечают строки "mov eax,rem" и "call sprint".
2. Эти инструкции используются для чтения строки с вводом данных от пользователя. Начальный адрес строки сохраняется в регистре есх, а количество символов в строке сохраняется в регистре edx. После этого вызывается команда sread, которая производит чтение строки.
3. Инструкция "call atoi" используется для преобразования строки в целое число.
4. За вычисление варианта отвечают строки "xor edx,edx", "mov ebx,20" и "div ebx". Строка "xor edx,edx": обнуляет регистр edx перед выполнением деления. Строка "mov ebx,20": загружает значение 20 в регистр ebx. Строка "div ebx": выполняет деление регистра еах на значение регистра ebx с сохранением частного в регистре еах и остатка в регистре edx.
5. Остаток от деления при выполнении инструкции "div ebx" записывается в регистр edx.
6. Инструкция "inc edx" используется для увеличения значения в регистре edx на 1.
7. За вывод на экран результата вычислений отвечают строки "mov eax,edx" и "call iprintLF". "mov eax,edx" передает значение остатка от деления в регистр

eax, а “call iprintLF” вызывает процедуру iprintLF для вывода значения на экран вместе с переводом строки.

## 4) Самостоятельная работа

### 4.1) Создаем файл в каталоге

A terminal window with a dark background. The prompt is 'lyukshinava@fedora:~/work/arch-pc/lab06\$'. The command entered is 'touch ~/work/arch-pc/lab06/lab6-4.asm'.

```
lyukshinava@fedora:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-4.asm
```

Рис. 1: Создаем файл

## 4.2) Пишем программу для вычисления варианта №3 ( $x + 2)^2$ )

```
GNU nano 1.2 /home/tyukshimava/work/at-spi-2/calc.c
#include 'in_out.asm'

SECTION .data
msg : DB 'Введите X',0
rez : DB 'Ответ: ',0

SECTION .bss
x: RESB 80
rezl: RESB 80

SECTION .start
GLOBAL _start
_start:

mov eax,msg
call sprintf

mov ecx,x
mov edx, 80
call sread
mov eax,x
call atoi

add eax,2
mul eax
mov [rezl],eax

mov eax, rez
call sprintf
mov eax,[rezl]
call sprintf

call quit
```

Рис. 2: Пишем программу

### 4.3) Создаем файл и запускаем его

```
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите X
2
Ответ: 16
lyukshinava@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите X
8
Ответ: 100
lyukshinava@fedora:~/work/arch-pc/lab06$
```

Рис. 3: Создаем файл и запускаем

## **5)Выводы**

Мы научились писать программы для произведения расчетов на языке NASM.