

Отчёта по лабораторной работе 6

Освоение арифметических инструкций языка ассемблера NASM.

Гайбуллаев Фаррух Шухрат НПИМбв-01-21

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	23

Список иллюстраций

2.1	Код программы lab6-1.asm	7
2.2	Работа программы lab6-1.asm	8
2.3	Код программы lab6-1.asm	9
2.4	Работа программы lab6-1.asm	10
2.5	Код программы lab6-2.asm	11
2.6	Работа программы lab6-2.asm	12
2.7	Код программы lab6-2.asm	13
2.8	Работа программы lab6-2.asm	14
2.9	Работа программы lab6-2.asm	14
2.10	Код программы lab6-3.asm	15
2.11	Работа программы lab6-3.asm	16
2.12	Код программы lab6-3.asm	17
2.13	Работа программы lab6-3.asm	18
2.14	Код программы variant.asm	19
2.15	Работа программы variant.asm	20
2.16	Работа программы вычисления	22

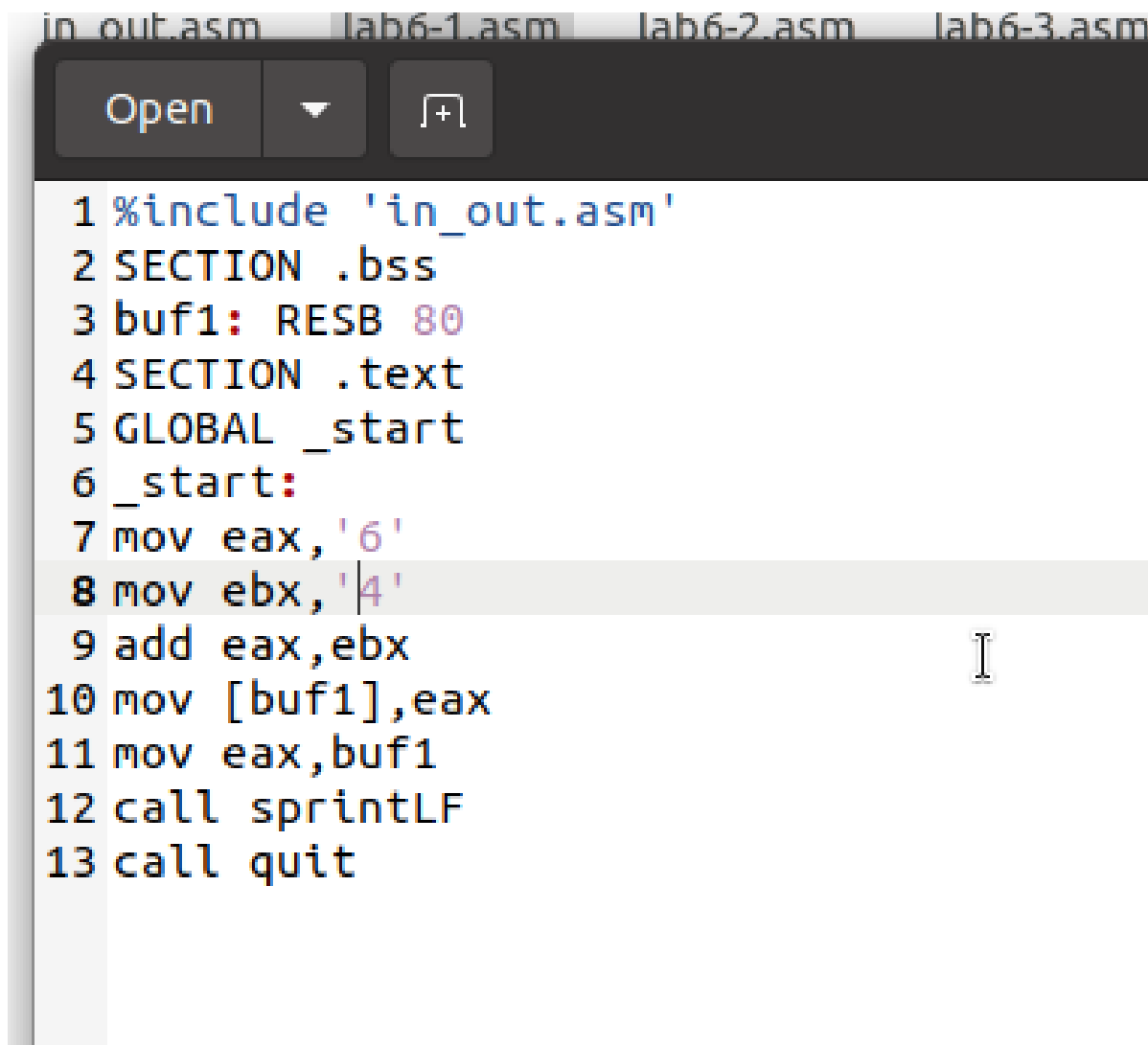
Список таблиц

1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

1. Создаем каталог для программ лабораторной работы № 6, переходим в него и создаем файл lab6-1.asm:
2. Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения, записанные в регистр eax.



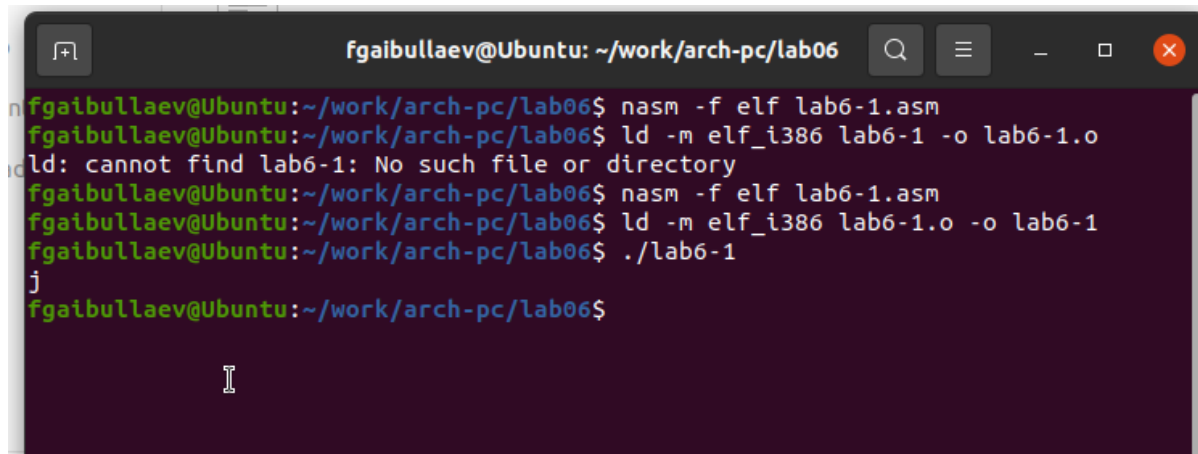
```
in_out.asm lab6-1.asm lab6-2.asm lab6-3.asm
Open ▼ [+]
```

```
1 %include 'in_out.asm'
2 SECTION .bss
3 buf1: RESB 80
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov eax, '6'
8 mov ebx, '4'
9 add eax, ebx
10 mov [buf1], eax
11 mov eax, buf1
12 call sprintf
13 call quit
```

Рис. 2.1: Код программы lab6-1.asm

```
%include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
```

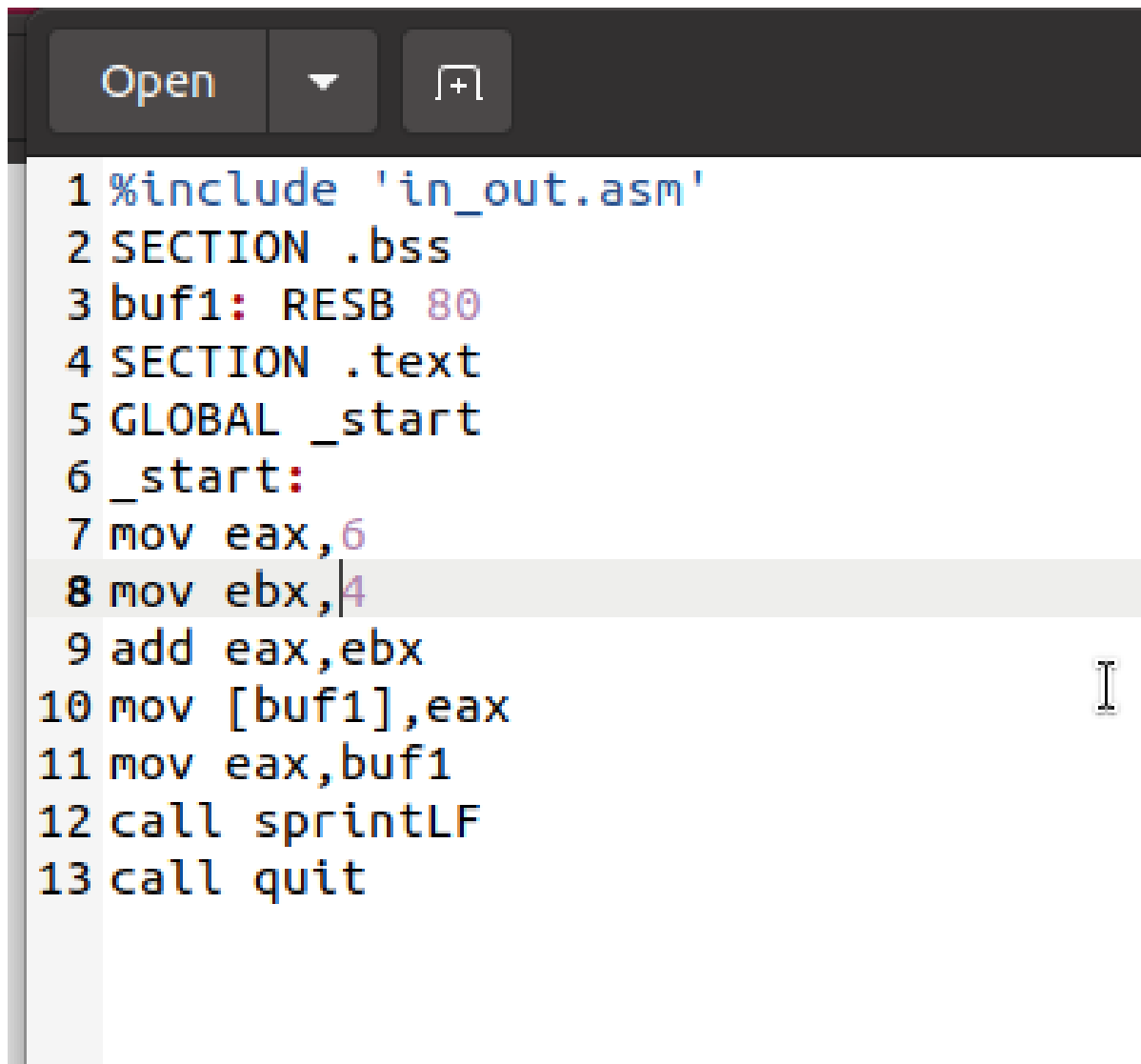
```
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit
```

A terminal window titled 'fgaibullaev@Ubuntu: ~/work/arch-pc/lab06' with standard window controls. The terminal shows the following commands and output:

```
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-1 -o lab6-1.o
ld: cannot find lab6-1: No such file or directory
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-1.o -o lab6-1
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ./lab6-1
j
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$
```

Рис. 2.2: Работа программы lab6-1.asm

3. Далее изменим текст программы и вместо символов, запишем в регистры числа.

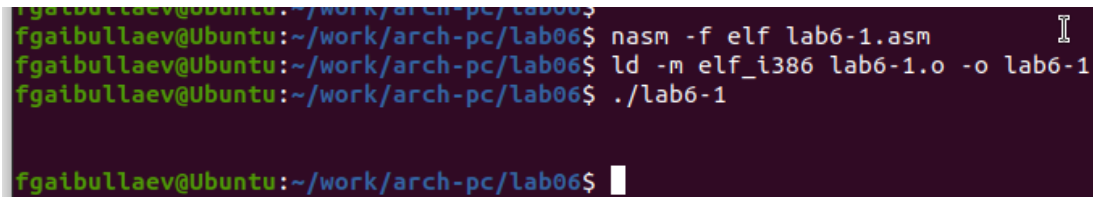


```
1 %include 'in_out.asm'
2 SECTION .bss
3 buf1: RESB 80
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov eax,6
8 mov ebx,4
9 add eax,ebx
10 mov [buf1],eax
11 mov eax,buf1
12 call sprintfLF
13 call quit
```

Рис. 2.3: Код программы lab6-1.asm

```
%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
```



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

fgaibullaev@Ubuntu:~/work/arch-pc/lab06$
```

Рис. 2.4: Работа программы lab6-1.asm

Никакой символ не виден, но он есть. Это возврат каретки LF.

4. Как отмечалось выше, для работы с числами в файле in_out.asm реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Преобразуем текст программы с использованием этих функций.

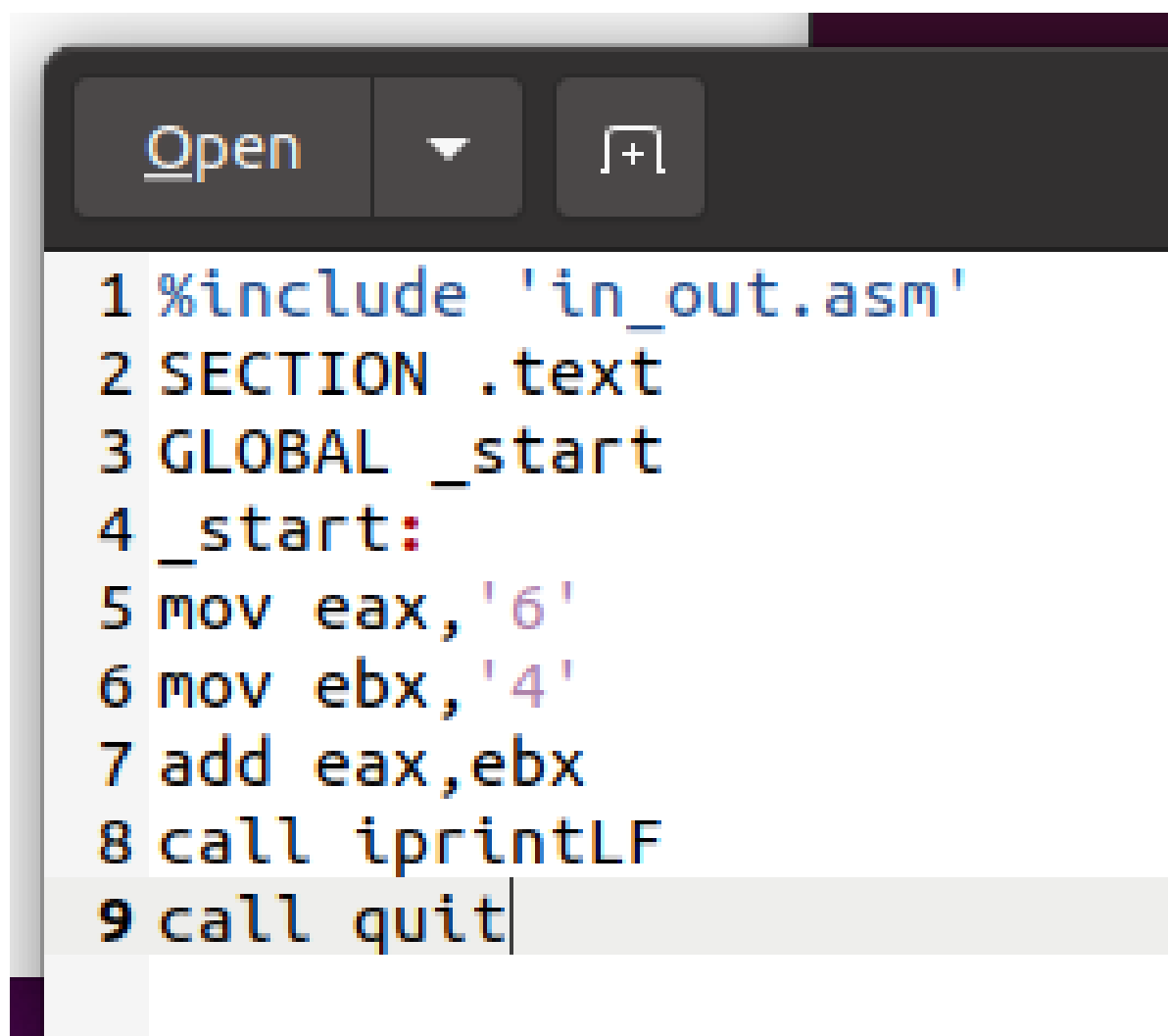
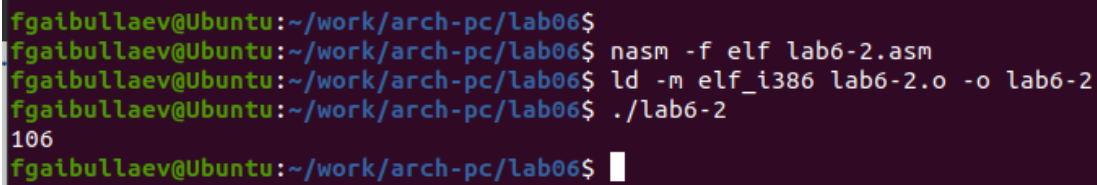


Рис. 2.5: Код программы lab6-2.asm

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
add eax, ebx
call iprintLF
```

call quit

A terminal window with a dark purple background. The prompt is fgaibullaev@Ubuntu:~/work/arch-pc/lab06\$. The user enters 'nasm -f elf lab6-2.asm', then 'ld -m elf_i386 lab6-2.o -o lab6-2', and finally './lab6-2'. The output '106' is displayed on the line following the execution command.

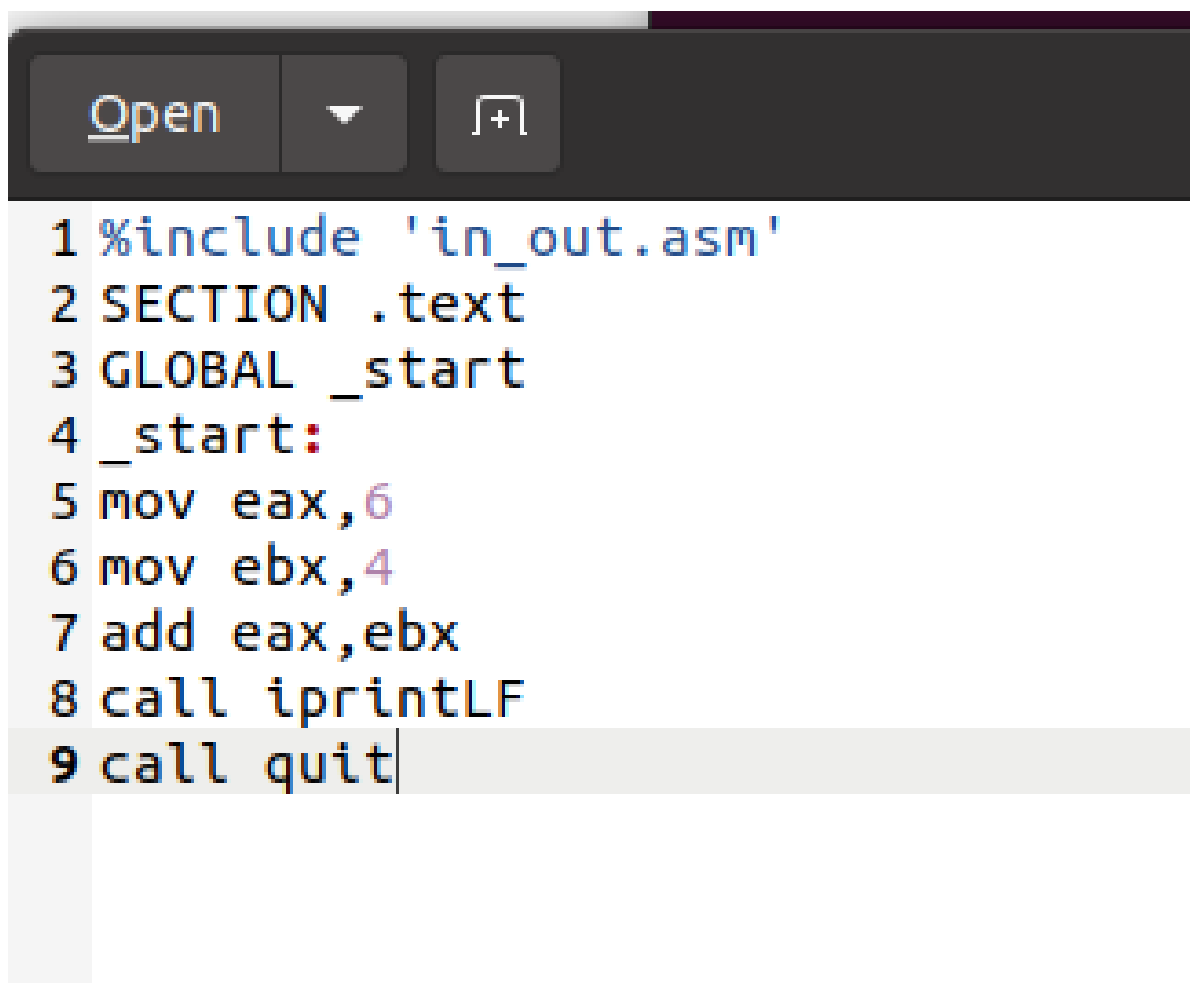
```
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$  
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm  
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-2.o -o lab6-2  
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ./lab6-2  
106  
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$
```

Рис. 2.6: Работа программы lab6-2.asm

В результате работы программы мы получим число 106. В данном случае, как и в первом, команда add складывает коды символов '6' и '4' ($54+52=106$). Однако, в отличие от прошлой программы, функция `iprintLF` позволяет вывести число, а не символ, кодом которого является это число.

5. Аналогично предыдущему примеру изменим символы на числа.

Создайте исполняемый файл и запустите его. Какой результат будет получен при исполнении программы? – получили число 10



```
1 %include 'in_out.asm'
2 SECTION .text
3 GLOBAL _start
4 _start:
5 mov eax,6
6 mov ebx,4
7 add eax,ebx
8 call iprintLF
9 call quit
```

Рис. 2.7: Код программы lab6-2.asm

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

```

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

```

Рис. 2.8: Работа программы lab6-2.asm

Замените функцию `iprintLF` на `iprint`. Создайте исполняемый файл и запустите его. Чем отличается вывод функций `iprintLF` и `iprint`? - Вывод отличается что нет переноса строки.

```

#include 'in_out.asm'

SECTION .text

GLOBAL _start

_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit

```

```

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

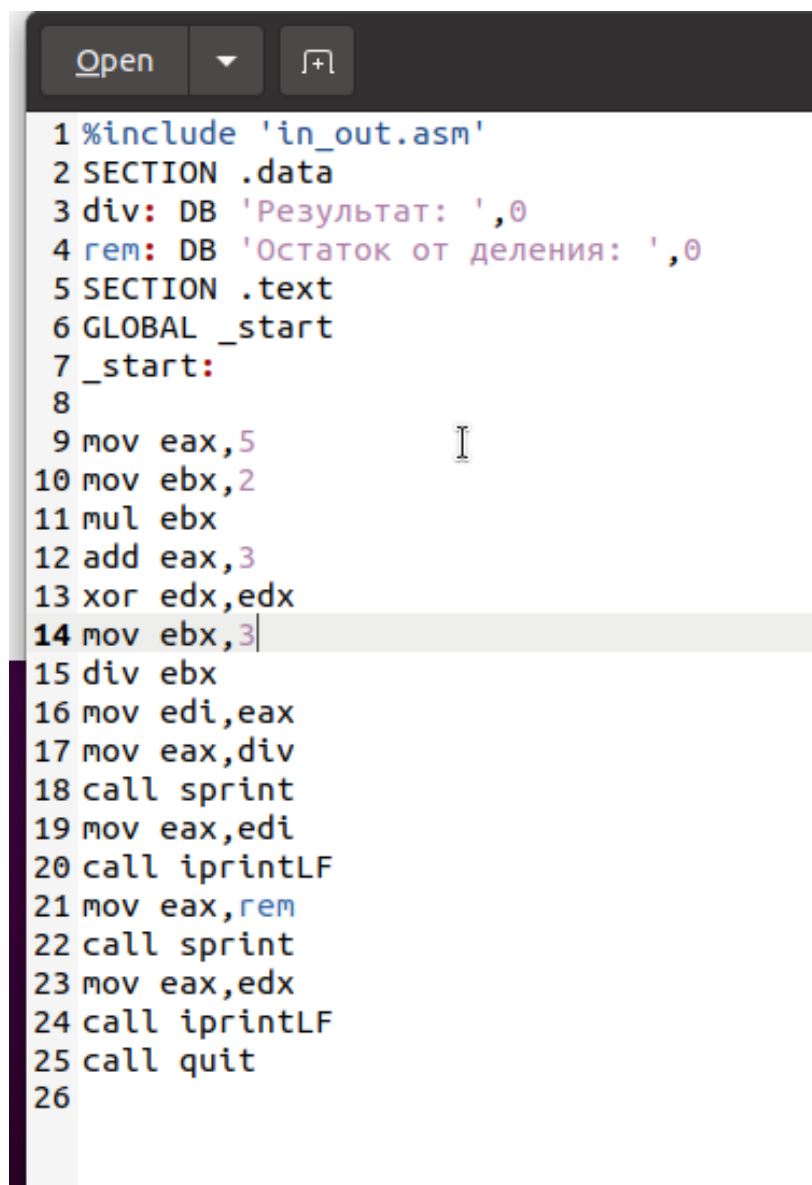
```

Рис. 2.9: Работа программы lab6-2.asm

- В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения

$$f(x) = (5 * 2 + 3) / 3$$

.



```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8
9 mov eax,5
10 mov ebx,2
11 mul ebx
12 add eax,3
13 xor edx,edx
14 mov ebx,3
15 div ebx
16 mov edi,eax
17 mov eax,div
18 call sprint
19 mov eax,edi
20 call iprintLF
21 mov eax,rem
22 call sprint
23 mov eax,edx
24 call iprintLF
25 call quit
26
```

Рис. 2.10: Код программы lab6-3.asm

```
%include 'in_out.asm'

SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0

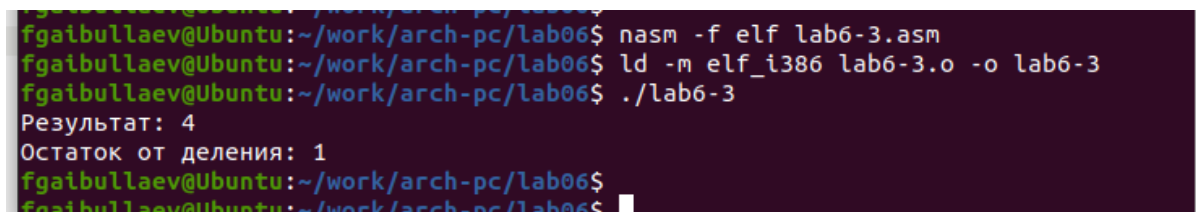
SECTION .text
GLOBAL _start
```

```

_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
call sprint
mov  eax,edx
call iprintLF
call quit

```



```

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

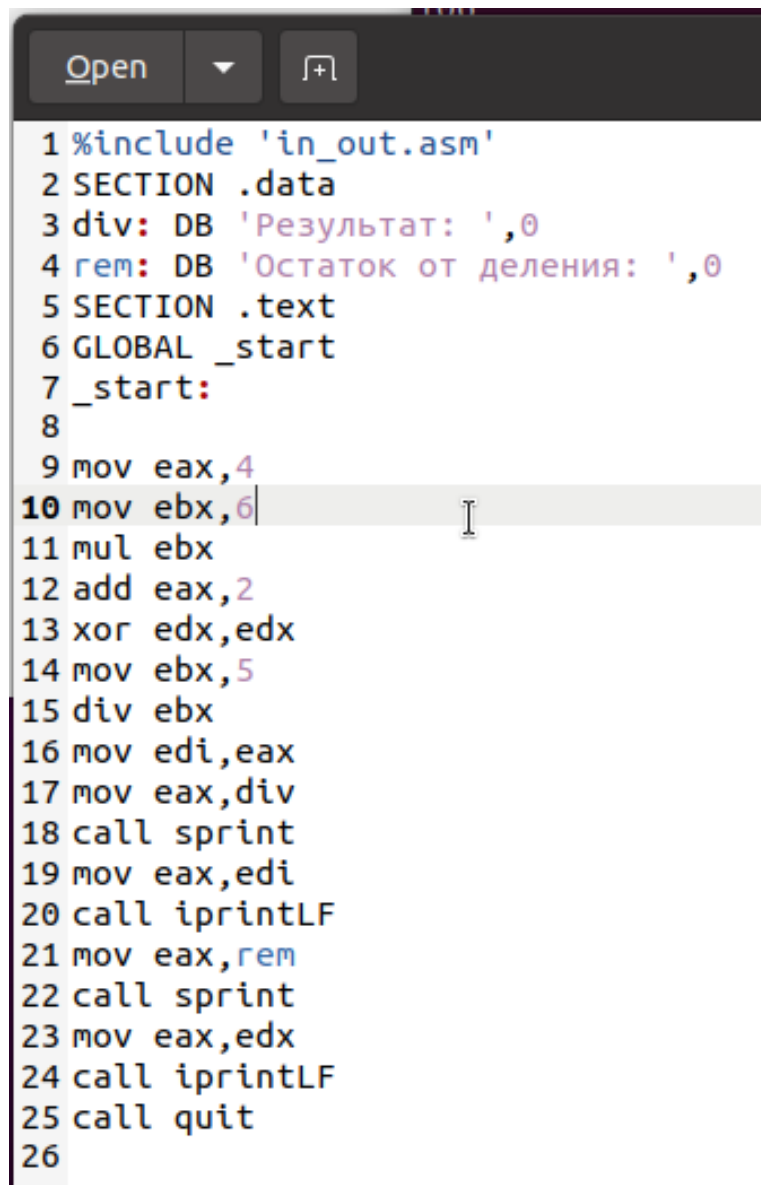
```

Рис. 2.11: Работа программы lab6-3.asm

Измените текст программы для вычисления выражения

$$f(x) = (4 * 6 + 2) / 5$$

. Создайте исполняемый файл и проверьте его работу.



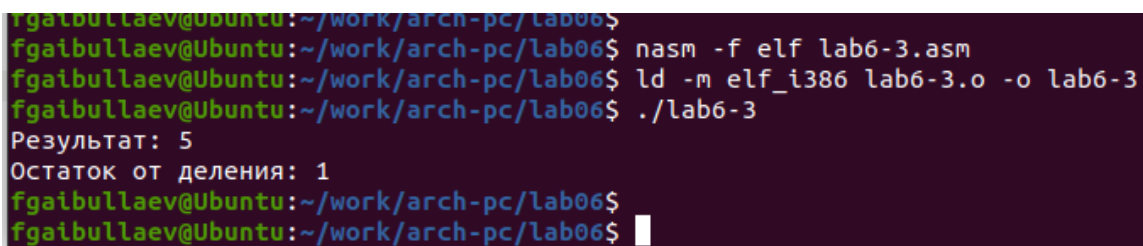
```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8
9 mov eax,4
10 mov ebx,6
11 mul ebx
12 add eax,2
13 xor edx,edx
14 mov ebx,5
15 div ebx
16 mov edi,eax
17 mov eax,div
18 call sprint
19 mov eax,edi
20 call iprintLF
21 mov eax,rem
22 call sprint
23 mov eax,edx
24 call iprintLF
25 call quit
26
```

Рис. 2.12: Код программы lab6-3.asm

```
%include 'in_out.asm'
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
```

```
GLOBAL _start
_start:
```

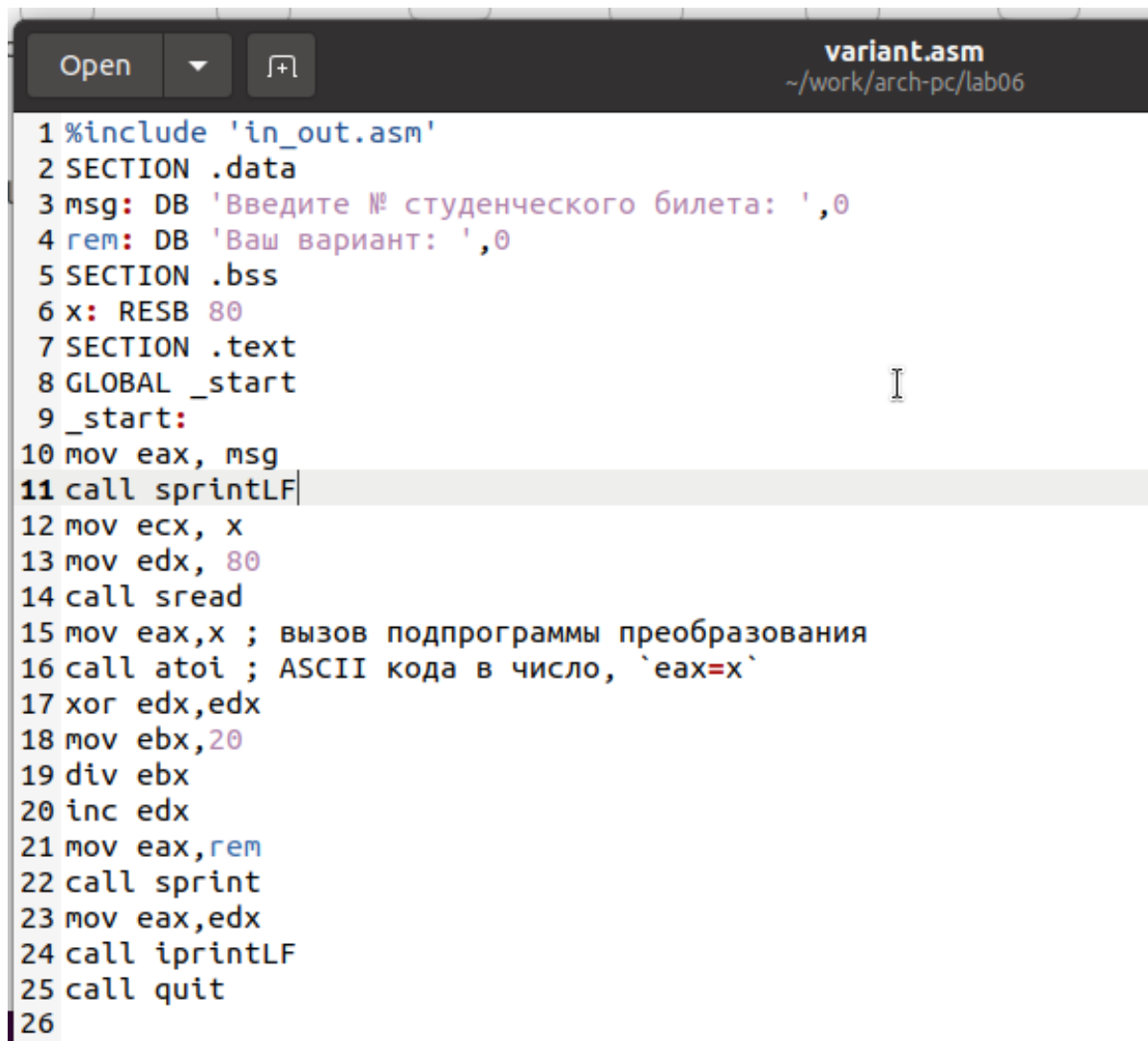
```
mov eax,4
mov ebx,6
mul ebx
add eax,2
xor edx,edx
mov ebx,5
div ebx
mov edi,eax
mov eax,div
call sprint
mov eax,edi
call iprintLF
mov eax,rem
call sprint
mov eax,edx
call iprintLF
call quit
```



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

Рис. 2.13: Работа программы lab6-3.asm

7. В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета:



```
variant.asm
~/work/arch-pc/lab06

1 %include 'in_out.asm'
2 SECTION .data
3 msg: DB 'Введите № студенческого билета: ',0
4 rem: DB 'Ваш вариант: ',0
5 SECTION .bss
6 x: RESB 80
7 SECTION .text
8 GLOBAL _start
9 _start:
10 mov eax, msg
11 call sprintf
12 mov ecx, x
13 mov edx, 80
14 call sread
15 mov eax, x ; вызов подпрограммы преобразования
16 call atoi ; ASCII кода в число, `eax=x`
17 xor edx, edx
18 mov ebx, 20
19 div ebx
20 inc edx
21 mov eax, rem
22 call sprintf
23 mov eax, edx
24 call iprintLF
25 call quit
26
```

Рис. 2.14: Код программы variant.asm

```
%include 'in_out.asm'

SECTION .data
msg: DB 'Введите № студенческого билета: ',0
rem: DB 'Ваш вариант: ',0

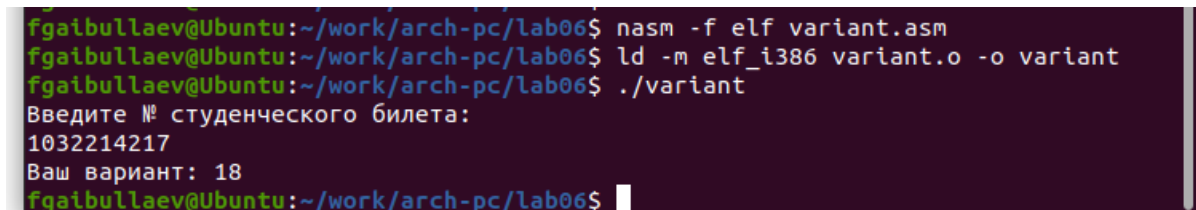
SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start
```

```

_start:
mov eax, msg
call sprintLF
mov ecx, x
mov edx, 80
call sread
mov eax,x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, `eax=x`
xor edx,edx
mov ebx,20
div ebx
inc edx
mov eax,rem
call sprint
mov eax,edx
call iprintLF
call quit

```



```

fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ nasm -f elf variant.asm
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 variant.o -o variant
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032214217
Ваш вариант: 18
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$

```

Рис. 2.15: Работа программы variant.asm

- Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? – mov eax,rem – перекладывает в регистр значение переменной с фразой ‘Ваш вариант:’ call sprint – вызов подпрограммы вывода строки
- Для чего используются следующие инструкции?

```

mov ecx, x
mov edx, 80
call sread``

```

Считывает значение студбилета в переменную X из консоли

* Для чего используется инструкция “call atoi”? - эта подпрограмма переводит вв

* Какие строки листинга 6.4 отвечают за вычисления варианта?

```

``xor edx,edx
mov ebx,20
div ebx``

```

* В какой регистр записывается остаток от деления при выполнении инструкции “div

``1 байт AH

2 байта DX

4 байта EDX – наш случай``

* Для чего используется инструкция “inc edx”? по формуле вычисления варианта нуж

* Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений

```

mov eax,edx – результат перекладывается в регистр eax
call iprintLF – вызов подпрограммы вывода

```

8. Написать программу вычисления выражения $y = f(x)$. Программа должна выводить вы
для вычисления, выводить запрос на ввод значения x,

вычислять заданное выражение в зависимости от введенного x , выводить результат вычисления. Вид функции $f(x)$ выбрать из таблицы 6.3 вариантов заданий в соответствии с номером задания, полученным при выполнении лабораторной работы.

Создайте исполняемый файл и проверьте его работу для значений x_1 и x_2 из 6.3.

Получили вариант 18 - $3(x + 10) - 20$ для $x=1$ и 5

![Код программы вычисления](image/16.png){ #fig:016 width=70%, height=70% }

```
%include 'in_out.asm' SECTION .data msg: DB 'Введите X',0 rem: DB 'выражение = :',0 SECTION .bss x: RESB 80 SECTION .text GLOBAL _start _start: mov eax, msg call sprintf mov ecx, x mov edx, 80 call sread mov eax, x ; вызов подпрограммы преобразования call atoi ; ASCII кода в число, eax=x add eax, 10 mov ebx, 3 mul ebx sub eax, 20 mov ebx, eax mov eax, rem call sprintf mov eax, ebx call iprintLF call quit
```



```
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ nasm -f elf calc.asm
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 calc.o -o calc
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ./calc
Введите X
1
выражение = : 13
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ nasm -f elf calc.asm
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ld -m elf_i386 calc.o -o calc
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$ ./calc
Введите X
5
выражение = : 25
fgaibullaev@Ubuntu:~/work/arch-pc/lab06$
```

Рис. 2.16: Работа программы вычисления

3 Выводы

Изучили работу с арифметическими операциями