

Отчёт по лабораторной работе №6

Арифметические операции в NASM

Юсупова Ксения Равиловна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Символьные и численные данные в NASM	6
2.2	Выполнение арифметических операций в NASM	9
2.3	Ответы на вопросы	12
2.4	Задания для самостоятельной работы	13
3	Выводы	15

Список иллюстраций

2.1	Создали каталог с помощью команды mkdir и файл lab6-1.asm с помощью команды touch	6
2.2	Заполнили файл по листингу	6
2.3	Создали исполняемый файл и запустили его	7
2.4	Изменили файл	7
2.5	Создали исполняемый файл и запустили его	7
2.6	Создали файл lab6-2.asm	7
2.7	Заполнили файл lab6-2.asm	8
2.8	Создали исполняемый файл и запустили его	8
2.9	Изменили файл	8
2.10	Создали исполняемый файл и запустили его	8
2.11	Изменили файл	9
2.12	Создали исполняемый файл и запустили его	9
2.13	Создали файл lab6-3.asm	9
2.14	Заполнили файл lab6-3.asm	10
2.15	Создали исполняемый файл lab6-3.asm и запустили его	10
2.16	Изменили файл	11
2.17	Создали исполняемый файл lab6-3.asm и запустили его	11
2.18	Создали файл variant.asm	11
2.19	Заполнили файл variant.asm	12
2.20	Создали исполняемый файл variant.asm и запустили его	12
2.21	Создали файл lab6-4.asm	13
2.22	Заполнили файл lab6-4.asm	14
2.23	Создали исполняемый файл lab6-4.asm и запустили его, проверяя выведенный результат	14

Список таблиц

1 Цель работы

Освоить арифметические инструкции языка ассемблера NASM

2 Выполнение лабораторной работы

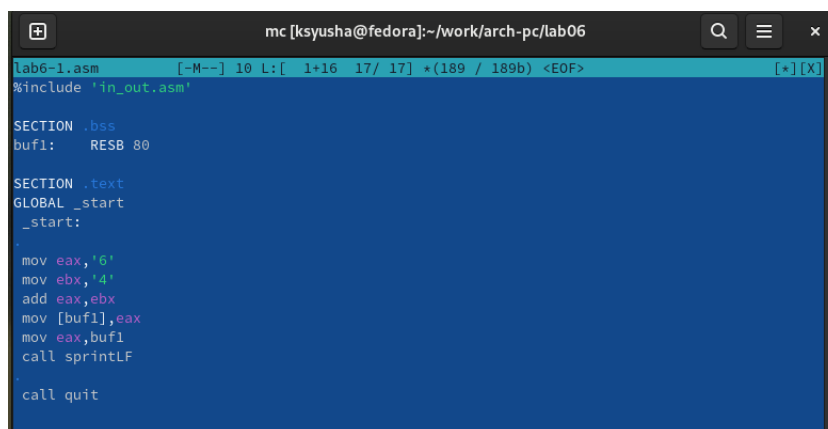
2.1 Символьные и численные данные в NASM

Создаём каталог для программ лабораторной работы № 6, переходим в него и создаём файл lab6-1.asm(рис. 2.1).

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

Рис. 2.1: Создали каталог с помощью команды mkdir и файл lab6-1.asm с помощью команды touch

Вводим в файл lab6-1.asm текст программы из листинга 6.1. В данной программе в регистр eax записывается символ 6 (mov eax,'6'), в регистр ebx символ 4 (mov ebx,'4'). (рис. 2.2).



```
lab6-1.asm [-M--] 10 L: [ 1+16 17/ 17] *(189 / 189b) <EOF> [*][X]
%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 sprintf
    call quit
```

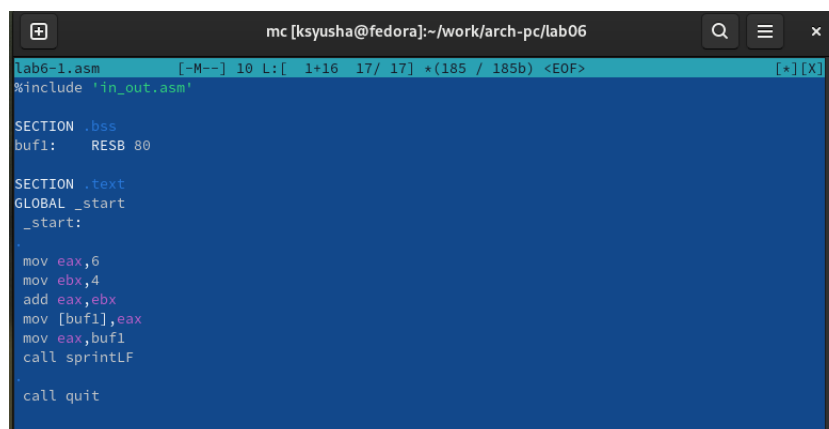
Рис. 2.2: Заполнили файл по листингу

Создаем исполняемый файл и запускаем его(рис. 2.3).

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

Рис. 2.3: Создали исполняемый файл и запустили его

Далее изменяем текст программы и вместо символов, записываем в регистры числа.(рис. 2.4).



```
lab6-1.asm [-M--] 10 L: [ 1+16 17/ 17] *(185 / 185b) <EOF> [*] [X]
#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 sprintf
call quit
```

Рис. 2.4: Изменили файл

Создаем исполняемый файл и запускаем его((рис. 2.5).

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

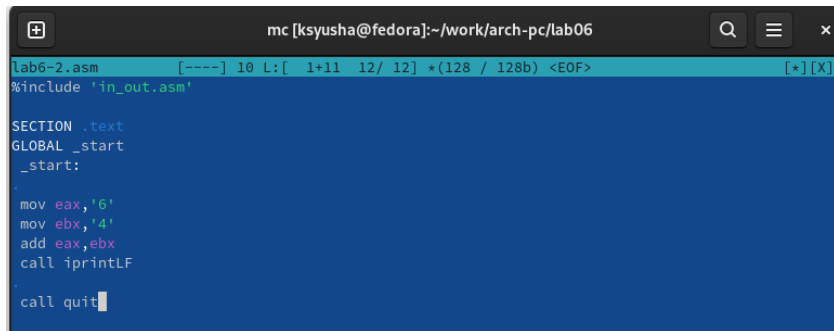
Рис. 2.5: Создали исполняемый файл и запустили его

Создаём файл lab6-2.asm в каталоге ~/work/arch-pc/lab06(рис. 2.6).

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

Рис. 2.6: Создали файл lab6-2.asm

Вводим в файл текст программы из листинга(рис. 2.7).

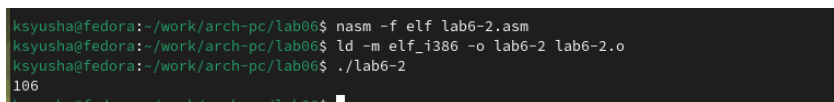


```
lab6-2.asm [----] 10 L: [ 1+11 12/ 12] *(128 / 128b) <EOF> [*][X]
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:
-
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
-
call quit
```

Рис. 2.7: Заполнили файл lab6-2.asm

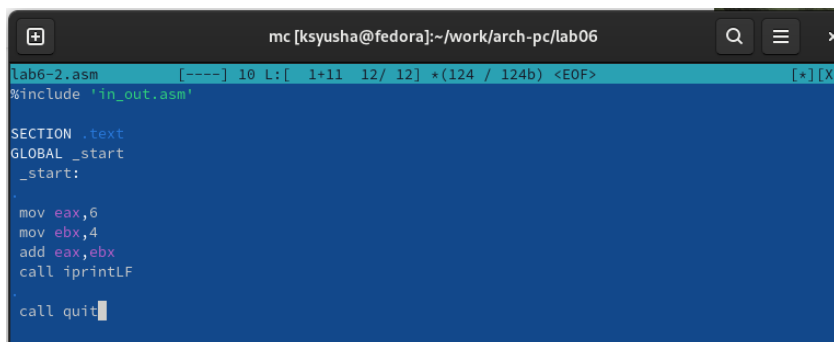
Создаем исполняемый файл и запускаем его(рис. 2.8).



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

Рис. 2.8: Создали исполняемый файл и запустили его

Далее также как и ранее изменяем текст программы и вместо символов, записываем числа(рис. 2.9).

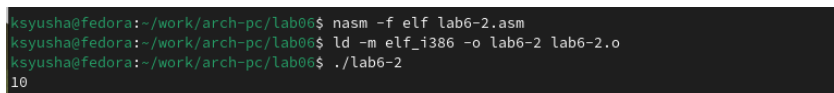


```
lab6-2.asm [----] 10 L: [ 1+11 12/ 12] *(124 / 124b) <EOF> [*][X]
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:
-
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
-
call quit
```

Рис. 2.9: Изменили файл

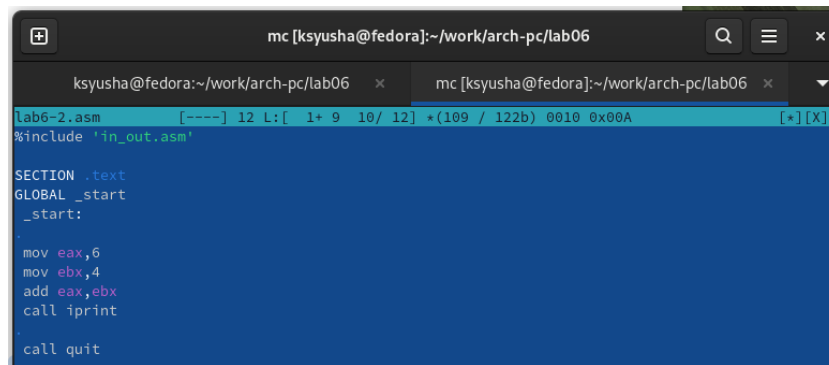
Создаем исполняемый файл и запускаем его(рис. 2.10).



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

Рис. 2.10: Создали исполняемый файл и запустили его

Открываем файл для редактирования и изменяем функцию iprintLF на iprint. (рис. 2.11).



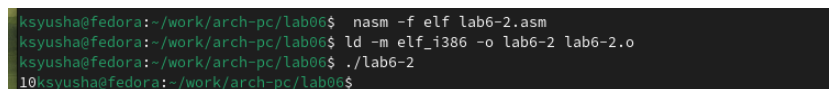
```
lab6-2.asm  [----] 12 L: [ 1+ 9 10/ 12] *(109 / 122b) 0010 0x00A [*][X]
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис. 2.11: Изменили файл

Создаем исполняемый файл и запускаем его(рис. 2.12).



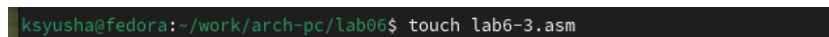
```
ksyusha@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
ksyusha@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
ksyusha@fedora:~/work/arch-pc/lab06$ ./lab6-2
10ksyusha@fedora:~/work/arch-pc/lab06$
```

Рис. 2.12: Создали исполняемый файл и запустили его

Таким образом, можем сделать вывод, что при использовании функции `iprintLF` переносит вывод на отдельную новую строку, а при использовании `iprint` этого не происходит.

2.2 Выполнение арифметических операций в NASM

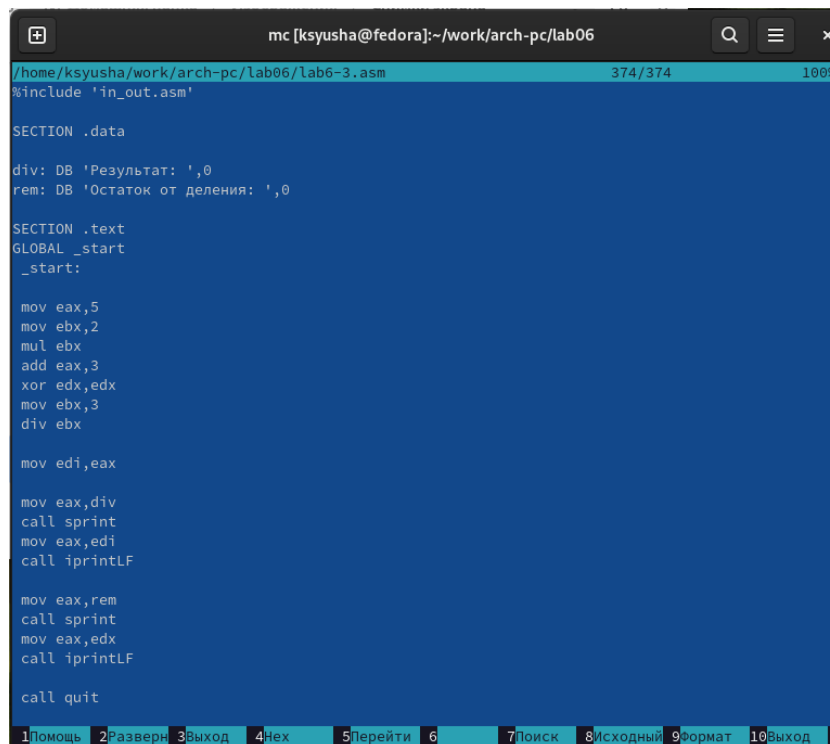
Создаём файл `lab6-3.asm` в каталоге `~/work/arch-pc/lab06`(рис. 2.13).



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

Рис. 2.13: Создали файл `lab6-3.asm`

Внимательно изучаем текст программы из листинга 6.3, в котором говорится как выполняется арифметическая операция в NASM для вычисления арифметического выражения $f(x) = (5 * 2 + 3)/3$. Вводим листинг в файл `lab6-3.asm`. (рис. 2.14).



```
mc [ksyusha@fedora]:~/work/arch-pc/lab06
/home/ksyusha/work/arch-pc/lab06/lab6-3.asm 374/374 100%
#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

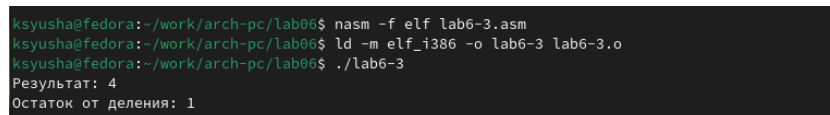
mov eax,div
call sprint
mov eax,edi
call iprintLF

mov eax,rem
call sprint
mov eax,edx
call iprintLF

call quit
```

Рис. 2.14: Заполнили файл lab6-3.asm

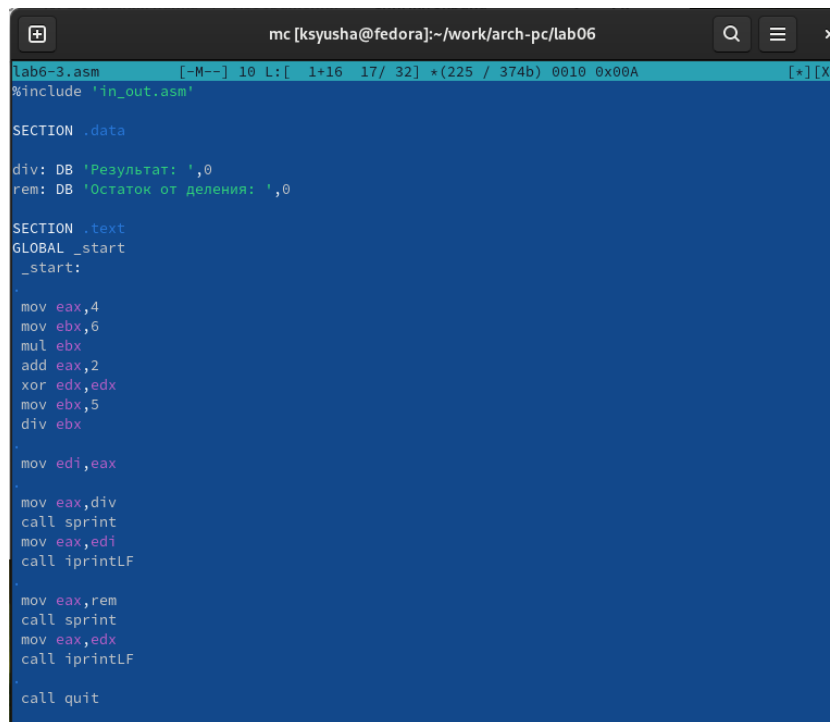
Создаем исполняемый файл и запускаем его(рис. 2.15).



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

Рис. 2.15: Создали исполняемый файл lab6-3.asm и запустили его

Изменяем текст программы для вычисления выражения $f(x) = (4 * 6 + 2)/5$.(рис. 2.16).



```
mc [ksyusha@fedora]:~/work/arch-pc/lab06
lab6-3.asm [-M--] 10 L: [ 1+16 17/ 32] *(225 / 374b) 0010 0x00A [*][X]
#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

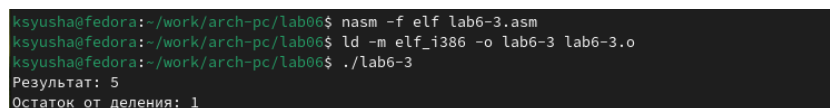
    mov edi,eax
    mov eax,div
    call sprint
    mov eax,edi
    call iprintLF

    mov eax,rem
    call sprint
    mov eax,edx
    call iprintLF

    call quit
```

Рис. 2.16: Изменили файл

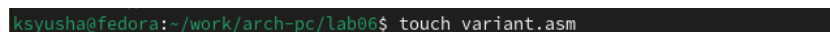
Создаем исполняемый файл и запускаем его(рис. 2.17).



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

Рис. 2.17: Создали исполняемый файл lab6-3.asm и запустили его

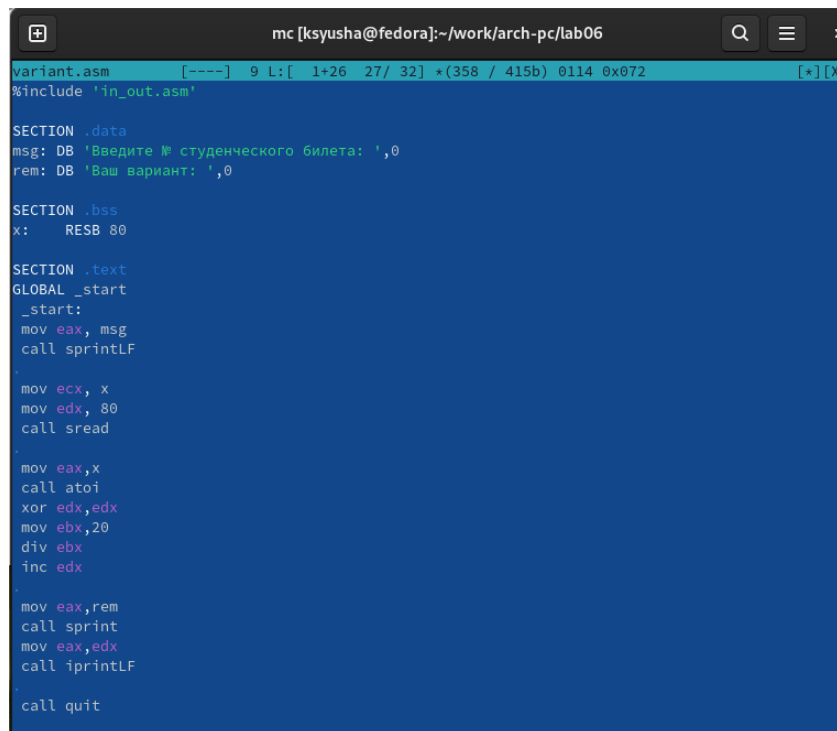
Создаём файл variant.asm в каталоге ~/work/arch-pc/lab06(рис. 2.18).



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

Рис. 2.18: Создали файл variant.asm

Внимательно изучаем текст программы из листинга 6.4 и вводим в файл variant.asm.(рис. 2.19).



```
variant.asm  [----]  9 L: [ 1+26 27/ 32] *(358 / 415b) 0114 0x072  [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
mov eax, edx
call iprintLF

call quit
```

Рис. 2.19: Заполнили файл variant.asm

Создаем исполняемый файл и запускаем его(рис. 2.20).



```
ksyusha@fedora:~/work/arch-pc/lab06$ nasm -f elf variant.asm
ksyusha@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
ksyusha@fedora:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132247531
Ваш вариант: 12
```

Рис. 2.20: Создали исполняемый файл variant.asm и запустили его

2.3 Ответы на вопросы

1. За вывод на экран сообщения 'Ваш вариант:' в листинге 6.4 отвечают строки:

```
mov eax, rem
call sprint
```

2. Эти инструкции используются для считывания строки с предполагаемым вводом значений.

`mov ecx, x` - адрес строки сохраняется в регистре `ecx`
`mov edx, 80` - задаётся максимальное количество символов для считывания строки, и сохраняются в `edx`
`call sread` - инструкция для чтения строки

3. Инструкция `atoi` используется для корректной работы арифметических операций в NASM, так как символы необходимо преобразовать в числа, а ввод с клавиатуры осуществляется в символьном виде.

4. За вычисление варианта в листинге 6.4 отвечают строки:

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

5. Остаток от деления при выполнении инструкции `“div ebx”` записывается в регистр `edx`.

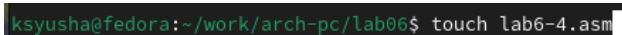
6. Инструкция `“inc edx”` используется для увеличения на 1 регистра `edx`

7. За вывод на экран результата вычислений листинга 6.4 отвечают строки:

```
mov eax,edx  
call iprintLF
```

2.4 Задания для самостоятельной работы

Создаем файл `lab6-4.asm` в каталоге `~/work/arch-pc/lab06`(рис. 2.21).



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

Рис. 2.21: Создали файл `lab6-4.asm`

Пишем программу вычисления выражения $y = f(x)$. Программа должна вывести выражение для вычисления функции $f(x) = (8 \cdot x - 6) / 2$, выводить запрос на ввод значения x , вычислять заданное выражение в зависимости от введенного x , выводить результат вычислений. (рис. 2.22).

```

lab6-4.asm  [----]  1 L: [ 1+ 6 7/ 35] *(128 / 459b) 0083 0x053  [*][X]
#include 'in_out.asm'

SECTION .data
msg: DB 'Введите x: ',0
rem: DB 'Результат вычислений: ',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

    mov ebx, 8
    mul ebx
    sub eax, 6
    xor edx, edx
    mov ebx, 2
    div ebx
    mov edx, eax

    mov eax, rem
    call sprint
    mov eax, edx
    call iprintLF

    call quit
  
```

Рис. 2.22: Заполнили файл lab6-4.asm

Создаем исполняемый файл и запускаем его (рис. 2.23).

```

ksyusha@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
ksyusha@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
ksyusha@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
5
Результат вычислений: 17
ksyusha@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
1
Результат вычислений: 1
  
```

Рис. 2.23: Создали исполняемый файл lab6-4.asm и запустили его, проверяя выведенный результат

3 Выводы

В ходе лабораторной работы мы освоили арифметические инструкции языка ассемблера NASM.