

# **Лабораторная работа №6**

**Арифметические операции в NASM.**

Козлова Нонна Юрьевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Выводы</b>	<b>17</b>
	<b>Список литературы</b>	<b>18</b>

## Список иллюстраций

3.1	Используем команду touch . . . . .	7
3.2	Используем клавишу F4 . . . . .	8
3.3	Получаем символ j . . . . .	8
3.4	Используем клавишу F4 . . . . .	9
3.5	Получаем пустое поле . . . . .	10
3.6	Используем команду touch и клавишу F4 . . . . .	10
3.7	Получаем число 106 . . . . .	11
3.8	Используем клавишу F4 . . . . .	12
3.9	Получаем число 10 . . . . .	13
3.10	Пользуемся командой touch и клавишей F4 . . . . .	13
3.11	Получаем верный по заданию ответ . . . . .	13
3.12	Используем клавишу F4 . . . . .	14
3.13	Получаем верный по заданию ответ . . . . .	14
3.14	Пользуемся командой touch и клавишей F4 . . . . .	15
3.15	Получаем вариант 17 . . . . .	15

## **Список таблиц**

# 1 Цель работы

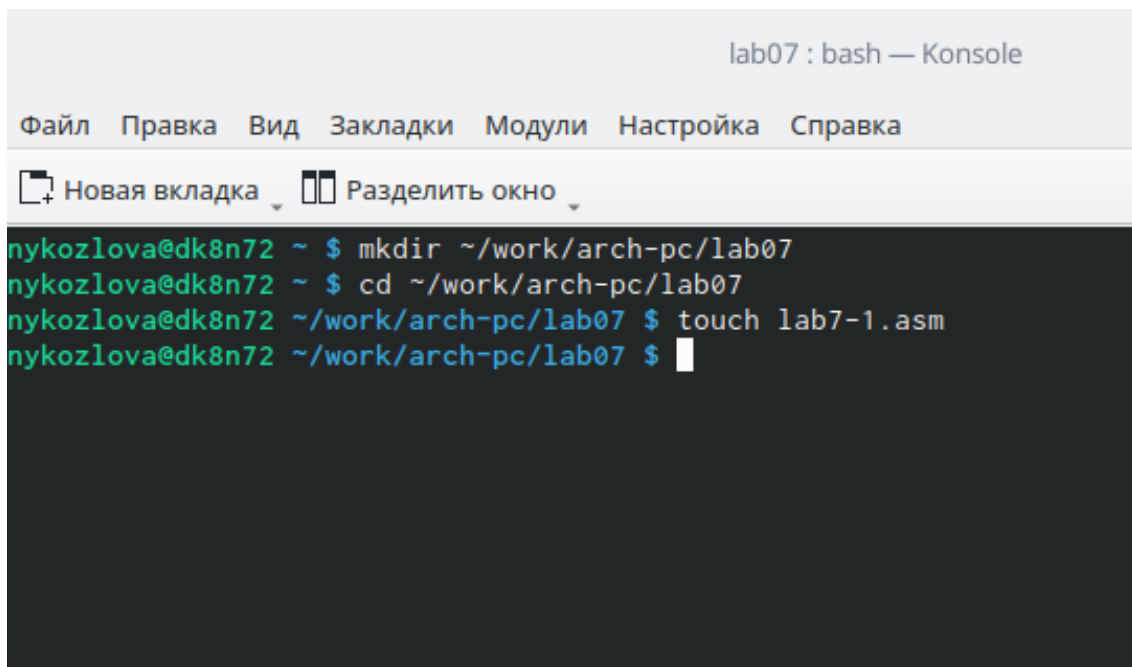
Освоение арифметических инструкций языка ассемблера NASM.

## 2 Задание

Практика в написании программ на языке ассемблера NASM.

### 3 Выполнение лабораторной работы

1. Создаем каталог для программ лабораторной работы, переходим в него и создаем файл lab7-1.asm (рис. 3.1)



The screenshot shows a terminal window titled "lab07 : bash — Konsole". The window has a menu bar with "Файл", "Правка", "Вид", "Закладки", "Модули", "Настройка", and "Справка". Below the menu bar is a toolbar with "Новая вкладка" and "Разделить окно". The terminal content shows the following commands and output:

```
nykozlova@dk8n72 ~ $ mkdir ~/work/arch-pc/lab07
nykozlova@dk8n72 ~ $ cd ~/work/arch-pc/lab07
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ touch lab7-1.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $
```

Рис. 3.1: Используем команду touch

2. Вводим в файл lab7-1.asm текст программы из листинга 7.1. (рис. 3.2)

```
lab7-1.asm [----] 0 L:[ 1+13 14/ 14] *(173 / 173b) <EOF;
%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
```

Рис. 3.2: Используем клавишу F4

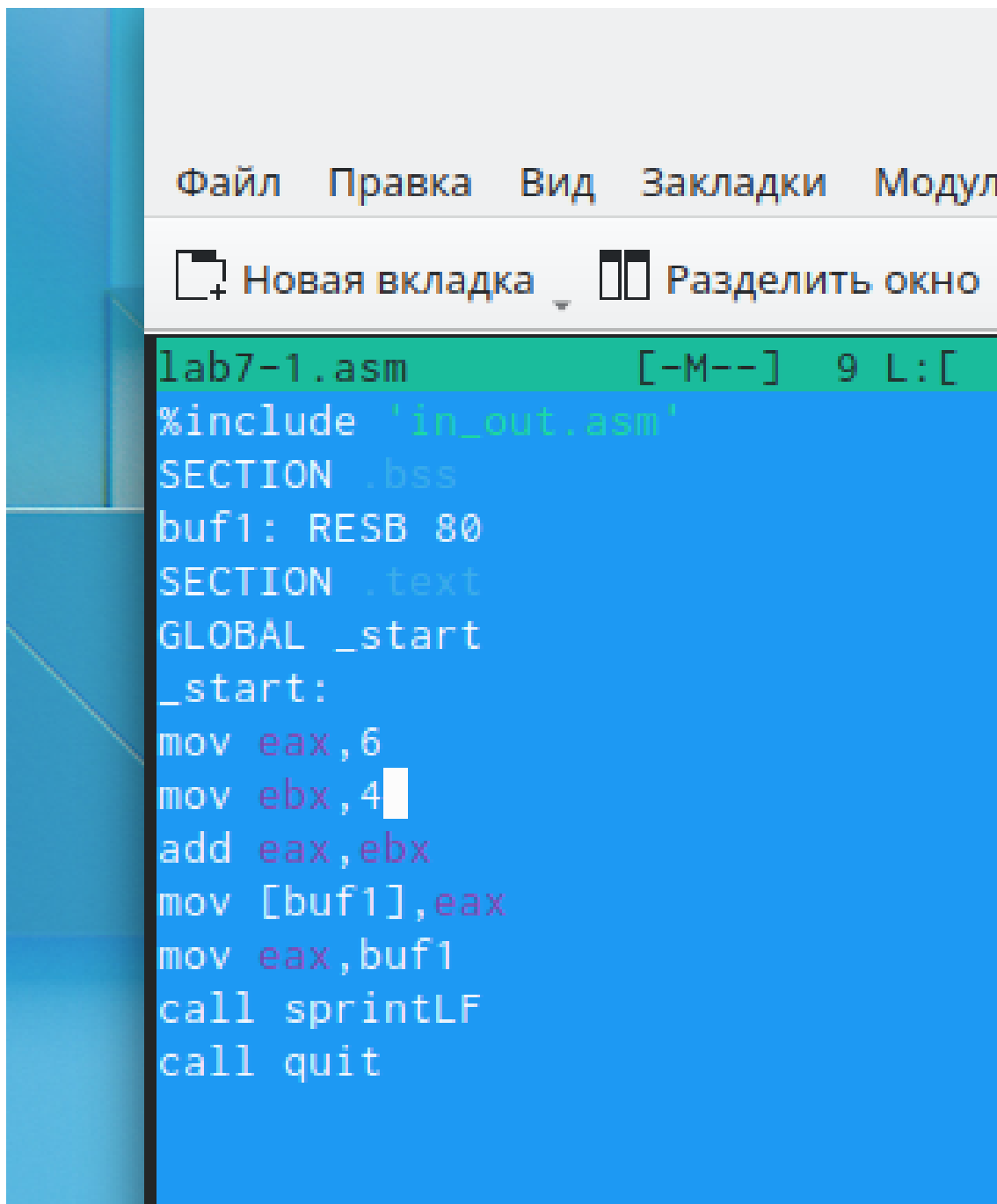
3. Создаем исполняемый файл и запускаем его. (рис. 3.3)

```
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./lab7-1
j
nykozlova@dk8n72 ~/work/arch-pc/lab07 $
```

Рис. 3.3: Получаем символ j



4. Изменим текст программы и вместо символов запишем в регистры числа.  
(рис. 3.4)



```
lab7-1.asm      [-M--]  9  L:[
#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. 3.4: Используем клавишу F4

5. Создаем исполняемый файл и запускаем его снова. (рис. 3.5)

```
lab07 : bash — Konsole
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
Новая вкладка  Разделить окно  Копировать  Вставить  Найти
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-01.o
ld: невозможно найти lab7-01.o: Нет такого файла или каталога
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ d -m elf_i386 -o lab7-1 lab7-1.o
bash: d: команда не найдена
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./lab7-1

nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./lab7-1

nykozlova@dk8n72 ~/work/arch-pc/lab07 $
```

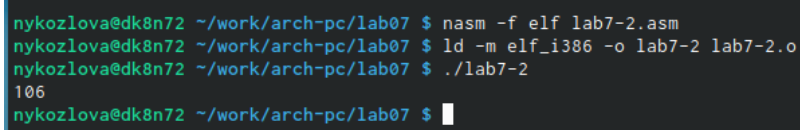
Рис. 3.5: Получаем пустое поле

6. Создаем файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 и вводим в него текст программы из листинга 7.2. (рис. 3.6)

```
lab07
Файл  Правка  Вид  Закладки  Модули  Настройка  С
Новая вкладка  Разделить окно
lab7-2.asm  [-M--]  9  L:[  1+ 8  9/ 9]
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
add eax, ebx
call iprintLF
call quit
```

Рис. 3.6: Используем команду touch и клавишу F4

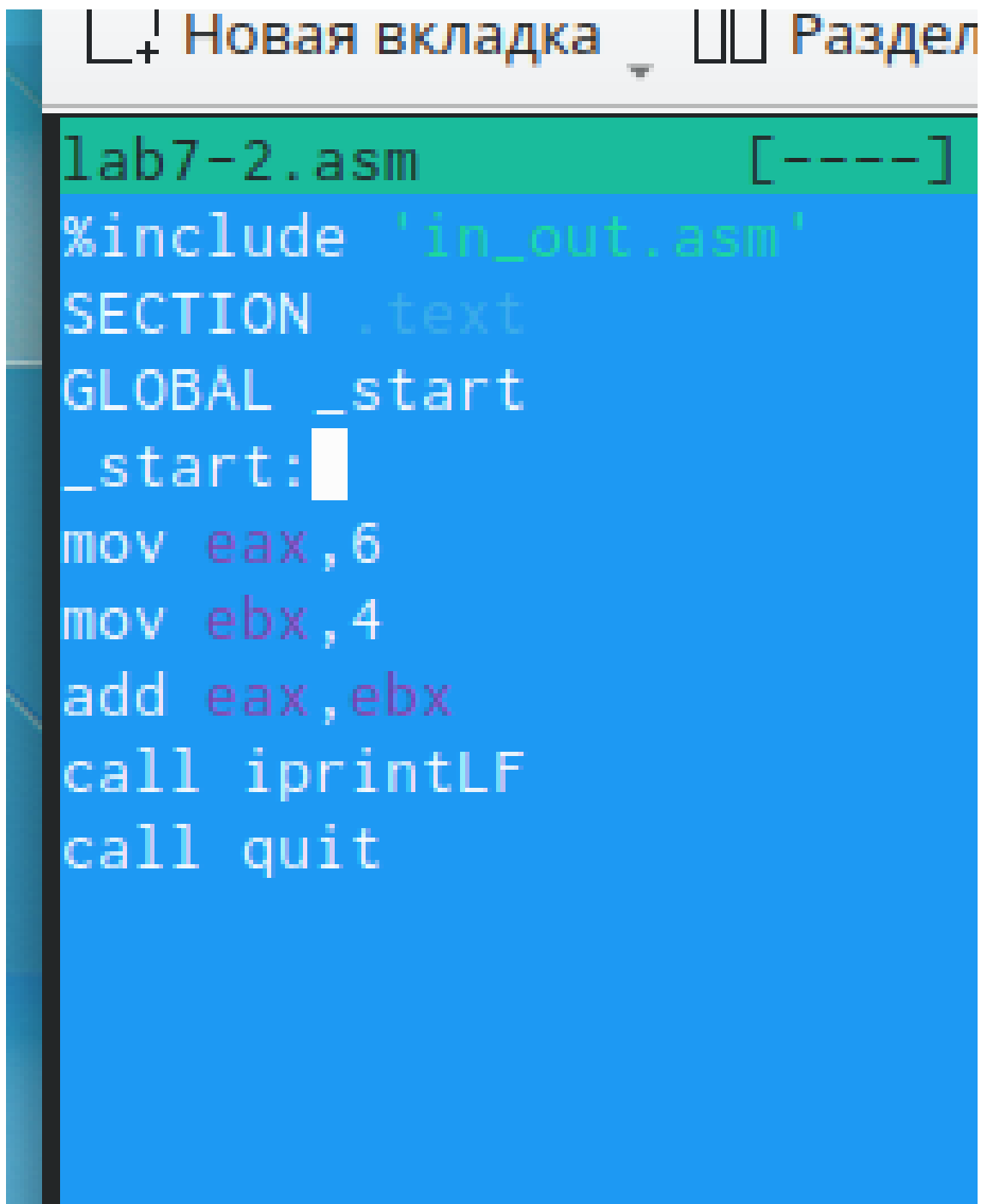
7. Создаем исполняемый файл и запускаем его. (рис. 3.7)



```
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ nasm -f elf lab7-2.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-2 lab7-2.o
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./lab7-2
106
nykozlova@dk8n72 ~/work/arch-pc/lab07 $
```

Рис. 3.7: Получаем число 106

8. Заменяем символы на числа. (рис. 3.8)



```
lab7-2.asm [-----]
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Рис. 3.8: Используем клавишу F4

9. Создаем исполняемый файл и запускаем его. (рис. 3.9)

```
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ nasm -f elf lab7-2.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-2 lab7-2.o
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./lab7-2
10
nykozlova@dk8n72 ~/work/arch-pc/lab07 $
```

Рис. 3.9: Получаем число 10

10. Создаем файл lab7-3.asm в каталоге ~/work/arch-pc/lab07 и вводим туда текст программы из личтинга 7.3. (рис. 3.10)

```
lab7-3.asm [-M--] 21 L: [ 1+10 11/ 26] *(311 /1236b) 0010 0x00A [*][X]
#include "in_out.asm" ; подключение внешнего файла
SECTION .data
div: DB "Результат: ",0
rem: DB "Остаток от деления: ",0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,5 ; EAX=5
mov ebx,2 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,3 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,3 ; EBX=3
div ebx ; EAX=EAX/3, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения "Результат: "
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения "Остаток от деления: "
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 3.10: Пользуемся командой touch и клавишей F4

11. Создаем исполняемый файл и запускаем его. (рис. 3.11)

```
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ nasm -f elf lab7-3.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-3 lab7-3.o
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./lab7-3
Результат: 4
Остаток от деления: 1
nykozlova@dk8n72 ~/work/arch-pc/lab07 $
```

Рис. 3.11: Получаем верный по заданию ответ

12. Изменяем текст программы для вычисления выражения  $\text{X}(\text{X}) = (4 \times 6 + 2)/5$ .  
(рис. 3.12)

```
lab7-3.asm      [-M--]  9 L:[ 1+13 14/ 26] *(423 /1236b) 0032 0x020
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,4 ; EAX=5
mov ebx,6 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=3
div ebx ; EAX=EAX/3, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 3.12: Используем клавишу F4

13. Создаем исполняемый файл и проверяем его работу. (рис. 3.13)

```
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ nasm -f elf lab7-3.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-3 lab7-3.o
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./lab7-3
Результат: 5
Остаток от деления: 1
nykozlova@dk8n72 ~/work/arch-pc/lab07 $
```

Рис. 3.13: Получаем верный по заданию ответ

14. Создаем файл variant.asm в каталоге ~/work/arch-pc/lab07 и вводим туда текст из листинга 7.4. (рис. 3.14)

```

variant.asm      [-M--]  9 L:[  1+24  25/ 25] *(490 /
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите No студенческого билета: ',0
rem: DB 'Ваш вариант: ',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'
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprintf
mov eax, edx
call iprintLF
call quit

```

Рис. 3.14: Пользуемся командой touch и клавишей F4

15. Создаем исполняемый файл и проверяем его работу. (рис. 3.15)

```

nykozlova@dk8n72 ~/work/arch-pc/lab07 $ nasm -f elf variant.asm
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o variant variant.o
nykozlova@dk8n72 ~/work/arch-pc/lab07 $ ./variant
Введите No студенческого билета:
113220816
Ваш вариант: 17
nykozlova@dk8n72 ~/work/arch-pc/lab07 $

```

Рис. 3.15: Получаем вариант 17

## 16. Ответы на вопросы

- 1) Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

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

- 2) Для чего используются следующие инструкции? `mov ecx, x` `mov edx, 80` `call sread`

`mov ecx, x` - запись входной переменной в регистр `ecx`; `mov edx, 80` - запись размера переменной в регистр `edx`; `call sread` - вызов процедуры чтения данных;

- 3) Для чего используется инструкция “`call atoi`”?

Вызов `atoi` – функции преобразующей `ascii`-код символа в целое число и записывающий результат в регистр `eax`.

- 4) Какие строки листинга 7.4 отвечают за вычисления варианта?

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

- 5) В какой регистр записывается остаток от деления при выполнении инструкции “`div ebx`”?

В регистр `ebx`.

- 6) Для чего используется инструкция “`inc edx`”?

Инструкция `INC` используется для увеличения операнда на единицу.

- 7) Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?

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



## 4 Выводы

В ходе лабораторной работы я

## **Список литературы**