

# **Шаблон отчёта по лабораторной работе**

6

Талебу Тенке Франк Устон , НКАбд-05-23

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы :</b>	<b>6</b>
2.1	Символьные и численные данные в NASM : . . . . .	6
2.2	Выполнение арифметических операций в NASM : . . . . .	12
2.3	Вопросы : . . . . .	17
2.4	Выводы по результатам выполнения заданий : . . . . .	18
<b>3</b>	<b>Задание для самостоятельной работы :</b>	<b>19</b>
3.1	Выводы по результатам выполнения заданий : . . . . .	21
<b>4</b>	<b>Выводы</b>	<b>22</b>

## Список иллюстраций

2.1	Ресунок 1	6
2.2	Ресунок 2	7
2.3	Ресунок 3	7
2.4	Ресунок 4	8
2.5	Ресунок 4	9
2.6	Ресунок 6	9
2.7	Ресунок 7	10
2.8	Ресунок 8	10
2.9	Ресунок 9	11
2.10	Ресунок 10	11
2.11	Ресунок 13	12
2.12	Ресунок 14	13
2.13	Ресунок 15	14
2.14	Ресунок 16	14
2.15	Ресунок 17	15
2.16	Ресунок 18	15
2.17	Ресунок 19	16
2.18	Ресунок 20	17
3.1	Ресунок 21	19
3.2	Ресунок 22	20

## **Список таблиц**

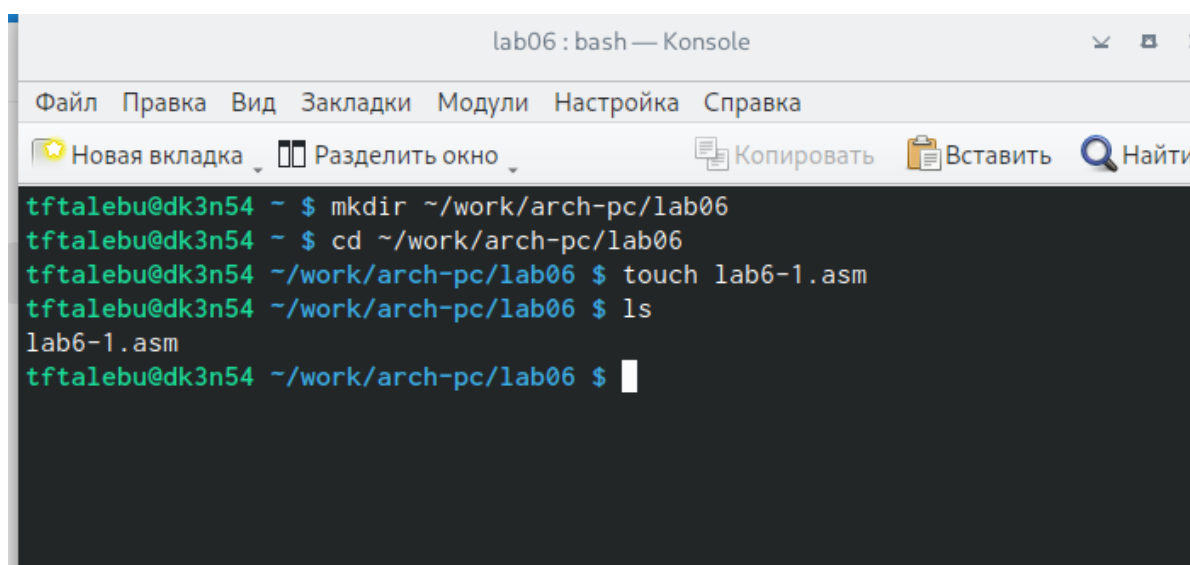
# 1 Цель работы

- В седьмой лабораторной работе можно будет освоить арифметические операции языка ассемблера.

## 2 Выполнение лабораторной работы :

### 2.1 Символьные и численные данные в NASM :

- Здесь мы начали с создания, а затем переместились в седьмой каталог лаборатории “~/work/arch-pc/lab07”, после чего мы создали файл “lab7-1.asm”.(рис. [??])



The screenshot shows a terminal window titled "lab06 : bash — Konsole". The terminal has a menu bar with "Файл", "Правка", "Вид", "Закладки", "Модули", "Настройка", and "Справка". Below the menu bar is a toolbar with icons for "Новая вкладка", "Разделить окно", "Копировать", "Вставить", and "Найти". The terminal output shows the following commands and results:

```
tftalebu@dk3n54 ~ $ mkdir ~/work/arch-pc/lab06
tftalebu@dk3n54 ~ $ cd ~/work/arch-pc/lab06
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ touch lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

Рис. 2.1: Ресунок 1

- После этого мы заполнили файл .asm кодом программы, отображающей значение регистра eax. (рис. [??])

```
of directory
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-1
j
tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

Рис. 2.2: Ресунок 2

- Затем мы скомпилировали файл, создали исполняемый файл и запустили программу, все это после перемещения файла in\_out.asm в тот же каталог, где находится lab7-1.asm.(рис. [??])

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

Рис. 2.3: Ресунок 3

- После этого мы изменили код в листинге следующим образом : `mov eax,6`  
`mov ebx,4` (рис. [??])

```
монтирования /)
Останавливаю поиск на границе файловой системы (так как GIT_DISCOVERY_ACROSS
_FILESYSTEM не установлен).
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ rasm -f elf lab6-1.asm
bash: rasm: команда не найдена
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
lab6-1.asm:1: error: unable to open include file 'in_out.asm': No such file
or directory
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-1
j
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ gedit lab6-1.asm
Failed to register: Время ожидания истекло
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ gedit lab6-1.asm
^Z[1]  Убито          gedit lab6-1.asm

[2]+  Остановлен      gedit lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-1

tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

Рис. 2.4: Ресунок 4

- Затем мы снова скомпилировали файл и создали исполняемый файл.(рис. [??])



```

монтирования /)
Останавливаю поиск на границе файловой системы (так как GIT_DISCOVERY_ACROSS
_FILESysten не установлен).
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ rasm -f elf lab6-1.asm
bash: rasm: команда не найдена
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
lab6-1.asm:1: error: unable to open include file 'in_out.asm': No such file
or directory
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-1
j
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ gedit lab6-1.asm
Failed to register: Время ожидания истекло
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ gedit lab6-1.asm
^Z[1]  Убито          gedit lab6-1.asm

[2]+  Остановлен    gedit lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-1

tftalebu@dk3n54 ~/work/arch-pc/lab06 $ 

```

Рис. 2.5: Рисунок 4

- Проверив ASCII tble символ, соответствующий коду 10 это новая строка, и мы можем сказать, что это было отображено, потому что при запуске программы она отобразила новую строку в качестве вывода.
- После этого мы создали файл lab-2.asm, в котором мы использовали под-программы, расположенные в файле in\_out.asm. (рис. [??])

```

tftalebu@dk3n54 ~/work/arch-pc/lab06 $ touch lab6-2.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o  lab6-2.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ 

```

Рис. 2.6: Рисунок 6

- После этого мы заполнили файл необходимым кодом для вывода значения реестра с помощью подпрограммы. (рис. [??])



```

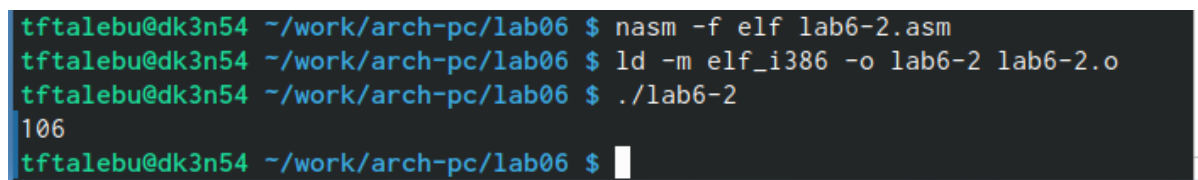
Открыть ▼ + lab6-2.asm
~/work/arch-pc/lab06

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

```

Рис. 2.7: Ресунок 7

- мы скомпилировали файл, создали исполняемый файл и запустили его. (рис. [??])



```

tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-2
106
tftalebu@dk3n54 ~/work/arch-pc/lab06 $

```

Рис. 2.8: Ресунок 8

- Аналогично предыдущему примеру, мы меняем символы на цифры, заменяя строки на : mov eax,6 mov ebx,4 (рис. [??])

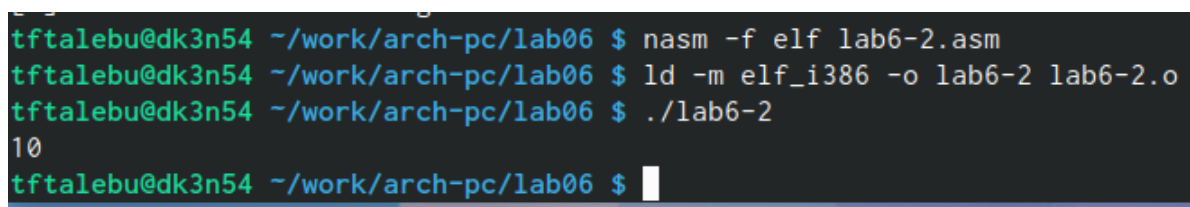


```
lab6-2.asm
~/work/arch-pc/lab06

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

Рис. 2.9: Рисунок 9

- Затем мы снова скомпилировали файл и создали исполняемый файл.(рис. [??])

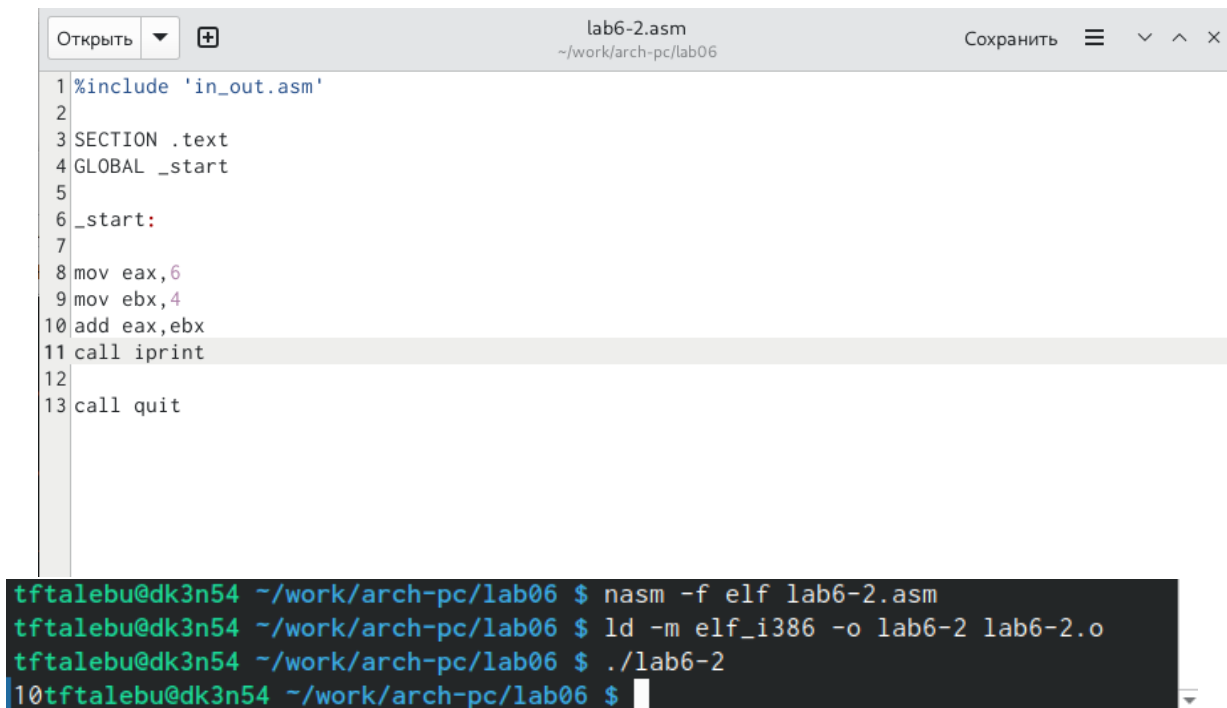


```
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-2
10
tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

Рис. 2.10: Рисунок 10

- На этот раз результатом, который мы получили, действительно было добавление 6 и 4 который 10.

-Затем мы заменили функцию iprintLF на iprint. После этого был создан исполняемый файл, и мы запустили его. (рис. [??]) (рис. [??])



The image shows a code editor window titled 'lab6-2.asm' with the following assembly code:

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

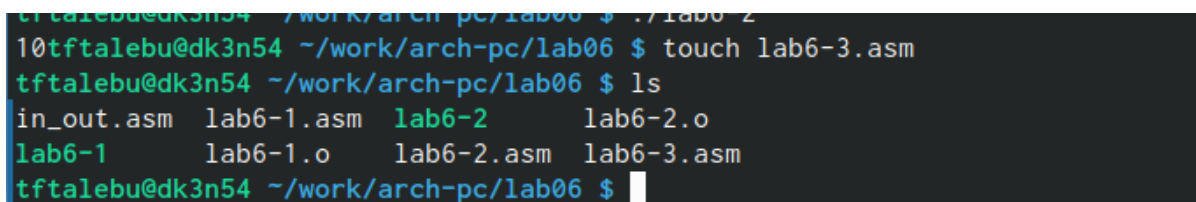
Below the editor is a terminal window showing the execution of the assembly code:

```
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-2
10tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

- Когда мы использовали подпрограмму `iprint`, мы заметили, что вывод отличается от предыдущего, потому что при использовании `iprint` не создается новая строка после вывода.

## 2.2 Выполнение арифметических операций в NASM :

- В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения  $\boxed{x}(\boxed{x}) = (5 \boxed{x} 2 + 3)/3$
- Мы создали файл `lab7-3.asm` в каталоге `~/work/arch-pc/lab07`. (рис. [??])

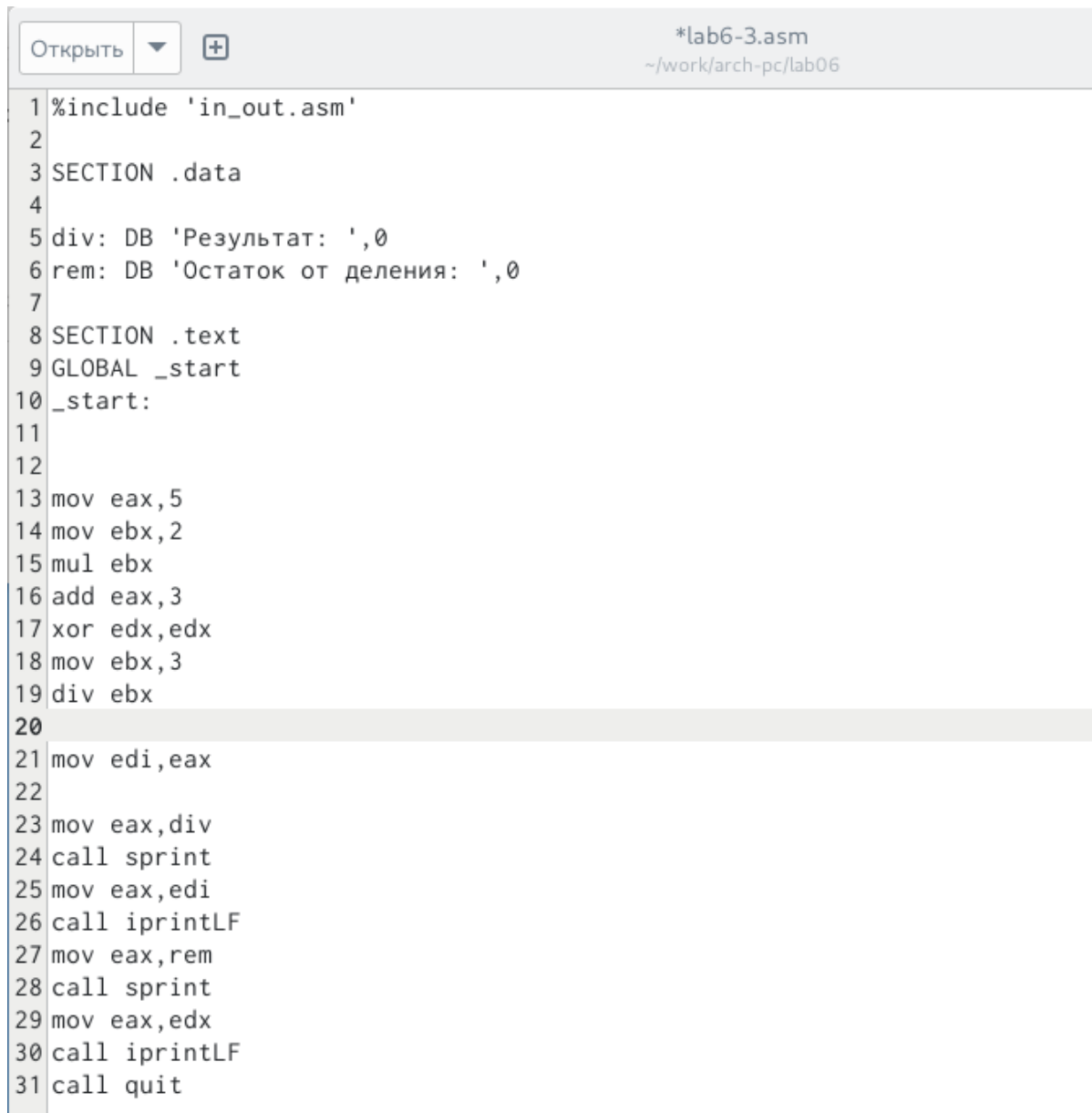


The image shows a terminal window with the following commands and output:

```
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-2
10tftalebu@dk3n54 ~/work/arch-pc/lab06 $ touch lab6-3.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm  lab6-2      lab6-2.o
lab6-1      lab6-1.o   lab6-2.asm  lab6-3.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

Рис. 2.11: Ресунок 13

- Затем мы заполнили файл необходимым кодом.(рис. [??])



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

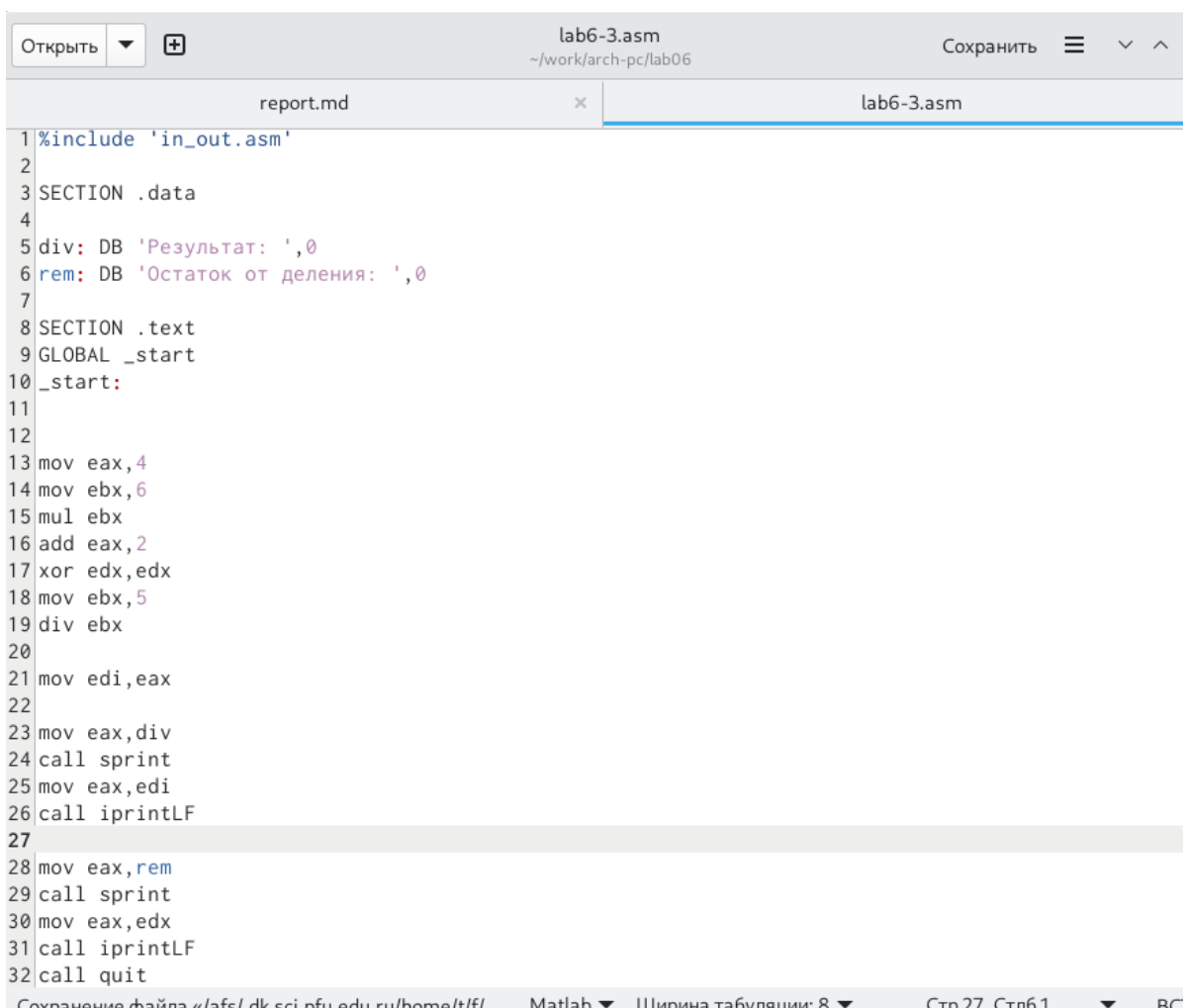
Рис. 2.12: Ресунок 14

- Создали исполняемый файл и запустили его.(рис. [??])

```
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ gcc -f lab6-3.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

Рис. 2.13: Ресунок 15

- Затем мы изменили текст программы, чтобы вычислить выражение:  $\boxed{4}(\boxed{4}) = (4 \boxed{\times} 6 + 2)/5$  (рис. [??])



```
lab6-3.asm
~/work/arch-pc/lab06
Сохранить

report.md x lab6-3.asm

1 %include 'in_out.asm'
2
3 SECTION .data
4
5 div: DB 'Результат: ',0
6 rem: DB 'Остаток от деления: ',0
7
8 SECTION .text
9 GLOBAL _start
10 _start:
11
12
13 mov eax,4
14 mov ebx,6
15 mul ebx
16 add eax,2
17 xor edx,edx
18 mov ebx,5
19 div ebx
20
21 mov edi,eax
22
23 mov eax,div
24 call sprint
25 mov eax,edi
26 call iprintLF
27
28 mov eax,rem
29 call sprint
30 mov eax,edx
31 call iprintLF
32 call quit

Сохранение файла ~/work/arch-pc/lab06/lab6-3.asm: OK
Matlab Шрифт: Consolas, 12pt Стр 27 Стр 61
```

Рис. 2.14: Ресунок 16

- мы создали исполняемый файл и проверили его работу. (рис. [??])

```

Результат: 5
Остаток от деления: 1
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ gedit lab6-3.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1
tftalebu@dk3n54 ~/work/arch-pc/lab06 $

```

Рис. 2.15: Ресунок 17

- На этом шаге мы написали программу, которая может вычислить дисперсию, которую мы получаем из номера студенческого билета.
- Мы начали с создания файла variant.asm. (рис. [??])

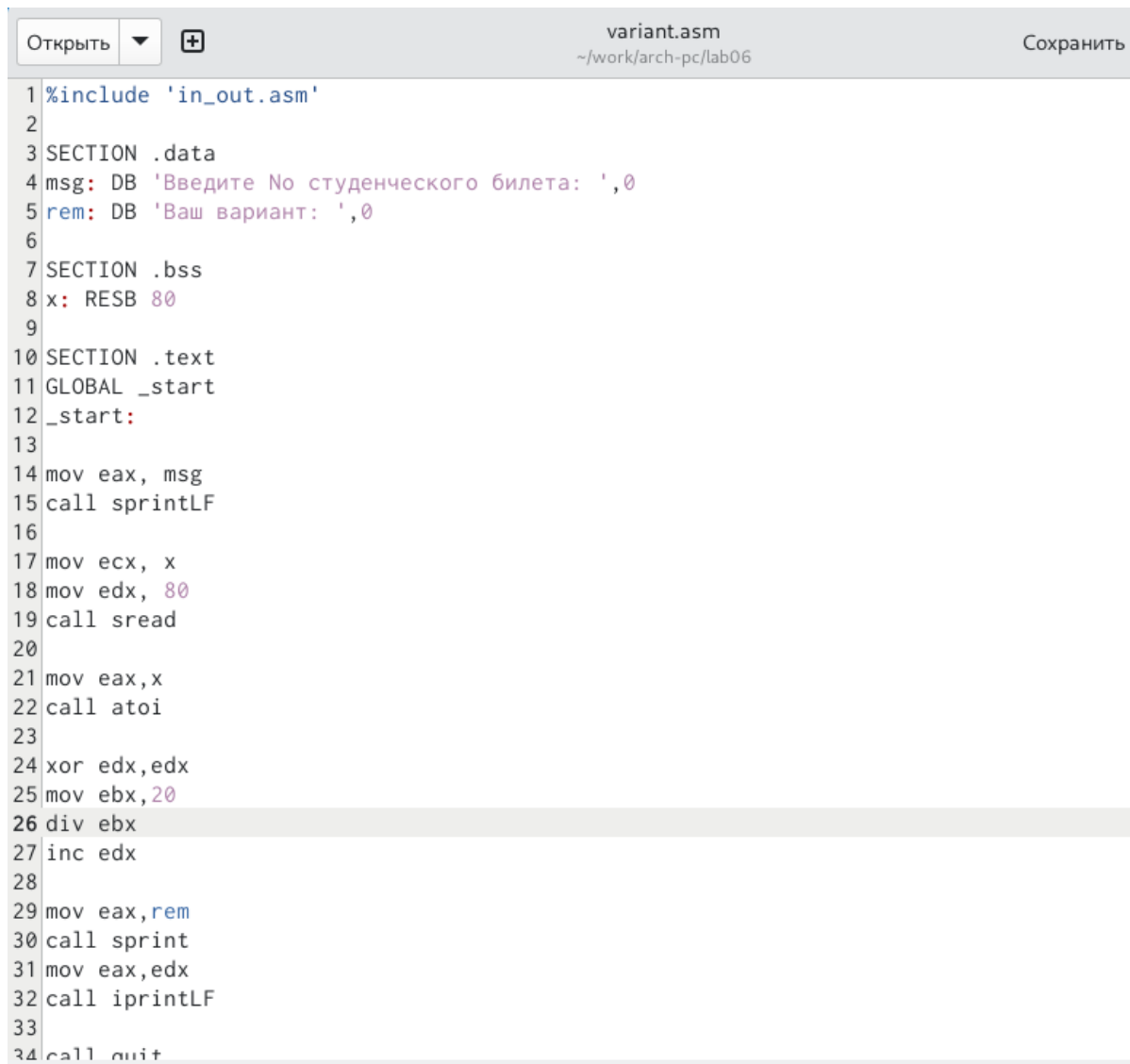
```

Остаток от деления: 1
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ touch variant.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.asm  lab6-2      lab6-2.o  lab6-3.asm  variant.asm
lab6-1      lab6-1.o   lab6-2.asm  lab6-3    lab6-3.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $

```

Рис. 2.16: Ресунок 18

- После этого мы написали код программы. (рис. [??])



```
1 %include 'in_out.asm'
2
3 SECTION .data
4 msg: DB 'Введите No студенческого билета: ',0
5 rem: DB 'Ваш вариант: ',0
6
7 SECTION .bss
8 x: RESB 80
9
10 SECTION .text
11 GLOBAL _start
12 _start:
13
14 mov eax, msg
15 call sprintf
16
17 mov ecx, x
18 mov edx, 80
19 call sread
20
21 mov eax, x
22 call atoi
23
24 xor edx, edx
25 mov ebx, 20
26 div ebx
27 inc edx
28
29 mov eax, rem
30 call sprintf
31 mov eax, edx
32 call iprintLF
33
34 call quit
```

Рис. 2.17: Ресунок 19

- мы создали исполняемый файл и проверили его работу, и действительно, в зависимости от номера студента он генерирует номер варианта. (рис. [??])



```
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ./variant
Введите No студенческого билета:
1032224534
Ваш вариант: 15
tftalebu@dk3n54 ~/work/arch-pc/lab06 $
```

Рис. 2.18: Ресунок 20

## 2.3 Вопросы :

- Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? **О** : rem: DB ‘Ваш вариант:’,0 mov eax,rem call sprint
- Для чего используются следующие инструкции? mov ecx, x / mov edx, 80 / call sread **О** : Эти инструкции были использованы для того, чтобы позволить пользователю вводить данные.
- Для чего используется инструкция “call atoi”? **О** : Эта инструкция используется для преобразования значения x из ASCII-кода в целое число.
- Какие строки листинга 7.4 отвечают за вычисления варианта? **О** : xor edx,edx  
mov ebx,20 div ebx inc edx
- В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”? **О** : Остаток был записан в регистре **edx**
- Для чего используется инструкция “inc edx”? **О** : Эта инструкция была использована для увеличения значения в регистре **edx**
- Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений? **О** : mov eax,edx call iprintLF

## **2.4 Выводы по результатам выполнения заданий :**

- В ходе лабораторной работы мы освоили выполнение арифметических операций на языке ассемблера и углубились в использование подпрограммы.

### 3 Задание для самостоятельной работы :

- В этой работе нам пришлось написать программу, которая просит пользователя ввести значение переменной и решить математическое выражение.
- Мой вариант : 16
- математическое выражение :  $(10x - 5)^2$
- Итак, мы начали с создания asm-файла, в котором будет находиться наш код.(рис. [??])

```
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ touch test.asm
tftalebu@dk3n54 ~/work/arch-pc/lab06 $ ls
in_out.asm  lab6-1.o    lab6-2.o    lab6-3.o    variant.asm
lab6-1      lab6-2      lab6-3      test.asm    variant.o
lab6-1.asm  lab6-2.asm  lab6-3.asm  variant
```

Рис. 3.1: Ресунок 21

- После этого мы написали код нашей программы. (рис. [??])

```

1 %include 'in_out.asm'
2
3 SECTION .data
4
5 msg1: DB 'Solve the equation (10*x - 5)^2 for x',0
6 msg:  DB 'Please enter the value of x: ',0
7 rem:  DB 'The resultat is : ',0
8 SECTION .bss
9 x :   RESB 80
10
11 SECTION .text
12 GLOBAL _start
13 _start:
14
15 mov  eax, msg1
16 call sprintLF
17
18 mov  eax, msg
19 call sprintLF
20
21 mov  ecx, x
22 mov  edx, 80
23 call sread
24
25 mov  eax, x
26 call atoi
27 mov  edi, eax
28
29 mov  eax, 10
30 mul  edi
31 sub  eax, 5
32 mov  edx, eax

```

Рис. 3.2: Ресунок 22

- и, наконец, мы проверяем корректность кода, который мы написали, используя два разных значения  $x_1 = 3$   $x_2 = 1$

Как указано на следующем рисунке (рис. [??])

```

tftalebu@dk8n51: ~/work/arch-pc/lab06
bash: gedit: команда не найдена
tftalebu@dk8n51: ~/work/arch-pc/lab06 $ gedit test.asm
tftalebu@dk8n51: ~/work/arch-pc/lab06 $ nasm -f elf test.asm
tftalebu@dk8n51: ~/work/arch-pc/lab06 $ ld -m elf_i386 -o test test.o
tftalebu@dk8n51: ~/work/arch-pc/lab06 $ ./test
Solve the equation (10*x - 5)^2 for x
Please enter the value of x:
3
The resultat is : 625
tftalebu@dk8n51: ~/work/arch-pc/lab06 $ ./test
Solve the equation (10*x - 5)^2 for x
Please enter the value of x:
1
The resultat is : 1225
tftalebu@dk8n51: ~/work/arch-pc/lab06 $

```

### **3.1 Выводы по результатам выполнения заданий :**

- В этой части мы смогли узнать, как преобразовать некоторые математические идеи в реальный код на ассемблере, что помогло нам получить более глубокое представление о том, как работать с регистрами.

## 4 Выводы

- В седьмой лаборатории мы в основном научились писать программы, выполняющие арифметические операции, и научились вычислять математические выражения средней сложности.