

Отчёта по лабораторной работе №6

Арифметические операции в NASM.

Сидорова Арина Валерьевна

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
2.1	1.Символьные и численные данные в NASM	5
2.2	2.Выполнение арифметических операций в NASM	11
2.3	Задание для самостоятельной работы	16
3	Выводы	19

Список иллюстраций

2.1	Создаем каталог и файл, проверяем	5
2.2	Вводим программу	6
2.3	Выводим результат	6
2.4	Изменяем файл	7
2.5	Выводим результат	7
2.6	Создаем файл lab6-2.asm	8
2.7	Вводим текст в файл	8
2.8	Смотрим на работу программы	9
2.9	Изменяем файл	9
2.10	Смотрим на работу программы	10
2.11	Изменяем файл	10
2.12	Смотрим на работу программы	11
2.13	Создаем файл и проверяем	11
2.14	Вводим программу	12
2.15	Смотрим на результат работы программы	12
2.16	Изменяем файл	13
2.17	Смотрим на результат работы программы	13
2.18	Создаем файл	14
2.19	Вводим програму	14
2.20	Проверим результат работы программы	15
2.21	Пишем программу	16
2.22	Пишем программу	17
2.23	Проверяем работу программы	18

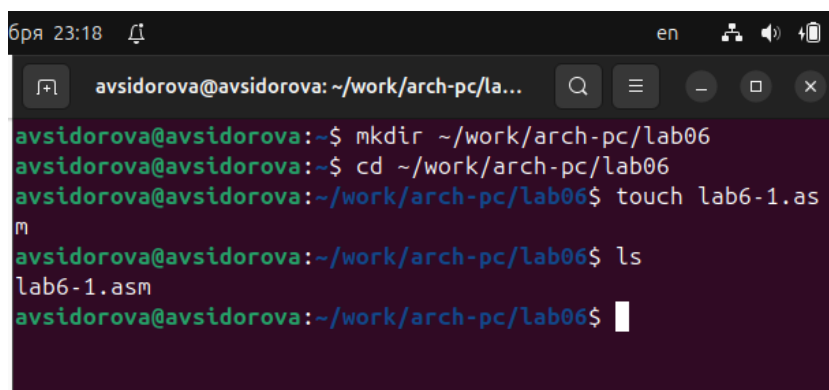
1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

2.1 1.Символьные и численные данные в NASM

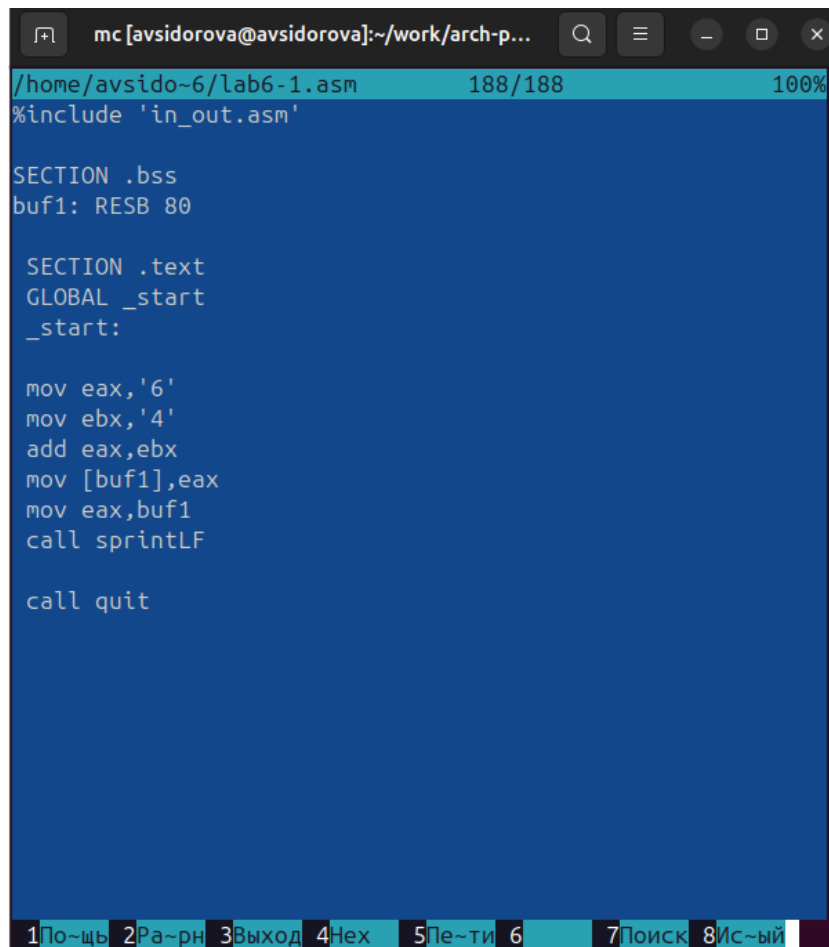
Создаем каталог для программ лабораторной работы № 6, переходим в него и создаем файл lab6-1.asm: (рис. fig. 2.1).

A screenshot of a terminal window with a dark background. The window title bar shows the date and time 'бря 23:18' and the language 'en'. The terminal shows the following commands and output:

```
avsidorova@avsidorova: ~/work/arch-pc/la...
avsidorova@avsidorova:~$ mkdir ~/work/arch-pc/lab06
avsidorova@avsidorova:~$ cd ~/work/arch-pc/lab06
avsidorova@avsidorova:~/work/arch-pc/lab06$ touch lab6-1.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$ ls
lab6-1.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$
```

Рис. 2.1: Создаем каталог и файл, проверяем

Введем в файл lab6-1.asm текст программы из листинга 6.1.(рис. fig. 2.2).



The screenshot shows a text editor window titled 'mc [avsidorova@avsidorova]:~/work/arch-p...'. The file being edited is '/home/avsidorova~6/lab6-1.asm' with 188/188 lines and 100% zoom. The code includes a header file 'in_out.asm'. It defines a .bss section with a buffer 'buf1' of 80 bytes. The .text section starts at '_start' and contains assembly instructions: 'mov eax, '6'', 'mov ebx, '4'', 'add eax, ebx', 'mov [buf1], eax', 'mov eax, buf1', 'call sprintLF', and 'call quit'. At the bottom, there is a tab bar with labels: '1По~щъ', '2Ра~рн', '3Выход', '4Нех', '5Пе~ти', '6', '7Поиск', '8Ис~ый'.

```
%include 'in_out.asm'

SECTION .bss
buf1: RESB 80

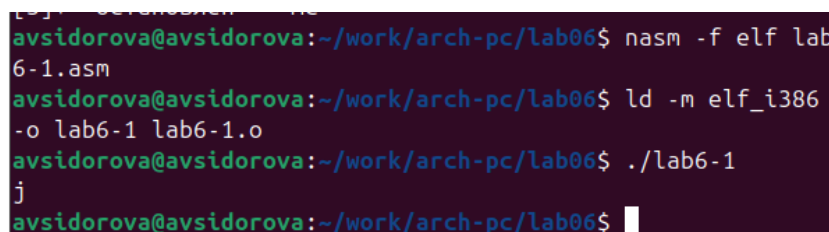
SECTION .text
GLOBAL _start
_start:

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

call quit
```

Рис. 2.2: Вводим программу

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

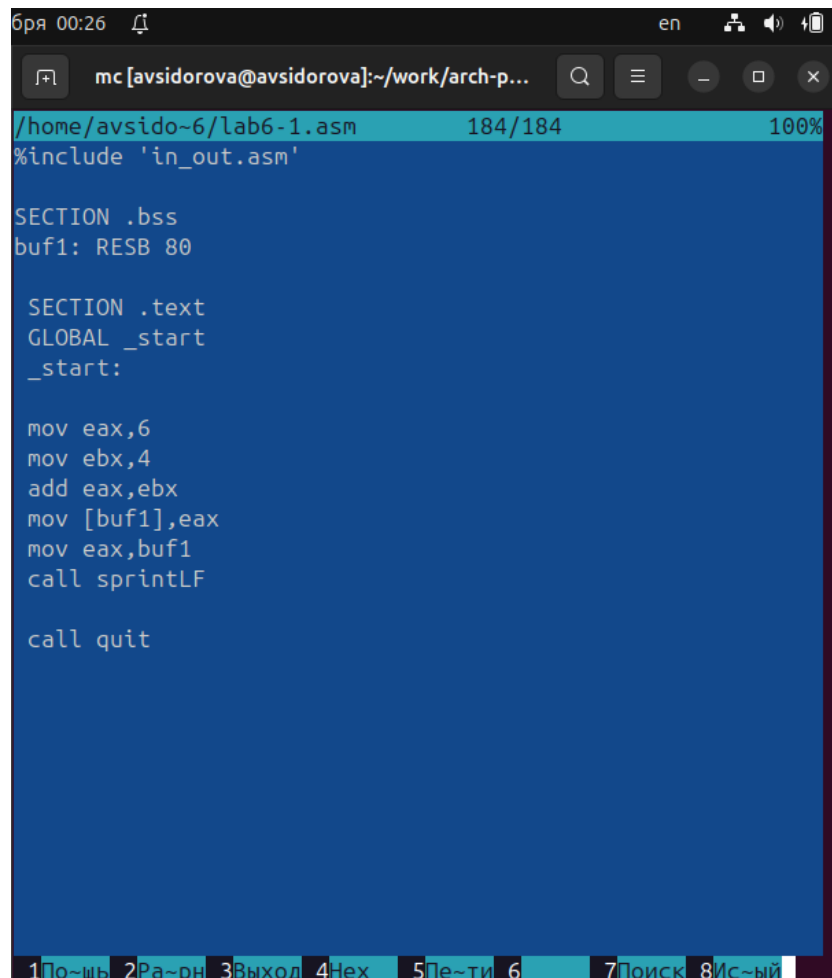


The screenshot shows a terminal window with the following commands and output: 'nasm -f elf lab6-1.asm', 'ld -m elf_i386 -o lab6-1 lab6-1.o', and './lab6-1'. The output shows the program running and printing 'j'.

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

Рис. 2.3: Выводим результат

Далее изменим текст программы и вместо символов, запишем в регистры числа. Исправим текст программы ((рис. fig. 2.4).

A screenshot of a code editor window. The title bar shows the date and time '6 ян 00:26' and the language 'en'. The editor's address bar shows the file path '/home/avsidorova-6/lab6-1.asm' with line numbers '184/184' and '100%'. The code content is as follows:

```
%include 'in_out.asm'

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:

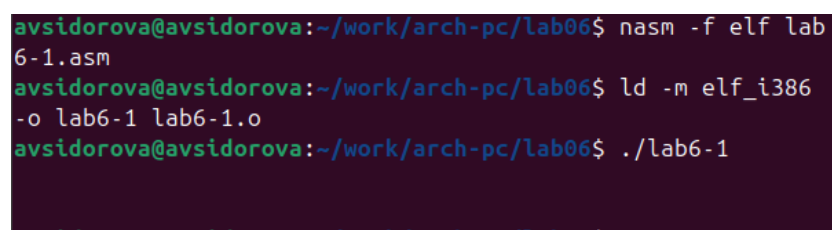
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF

call quit
```

At the bottom of the editor, there is a tab bar with several tabs labeled: '1По-щ', '2Ра-рн', '3Выход', '4Нех', '5Пе-ти', '6', '7Поиск', and '8Ис-ый'.

Рис. 2.4: Изменяем файл

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

A screenshot of a terminal window showing the following commands and their output:

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

Рис. 2.5: Выводим результат

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

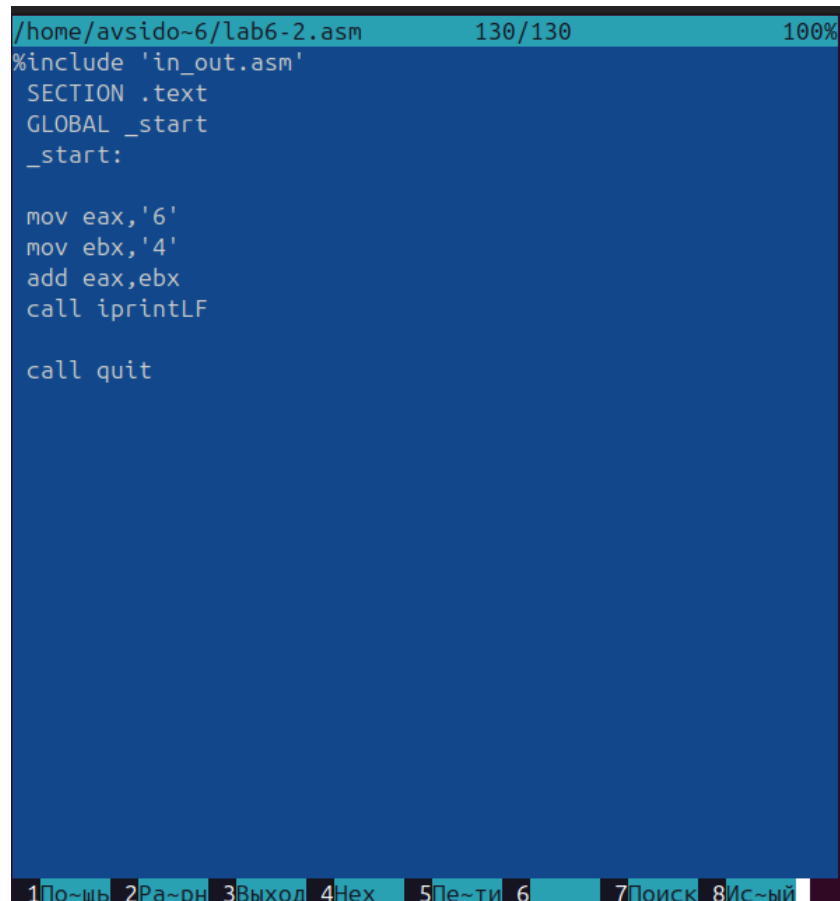
```

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

```

Рис. 2.6: Создаем файл lab6-2.asm

Введем в него текст программы из листинга 6.2. (рис. fig. 2.7).



```

/home/avsidorova/lab6-2.asm 130/130 100%
#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: Вводим текст в файл

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

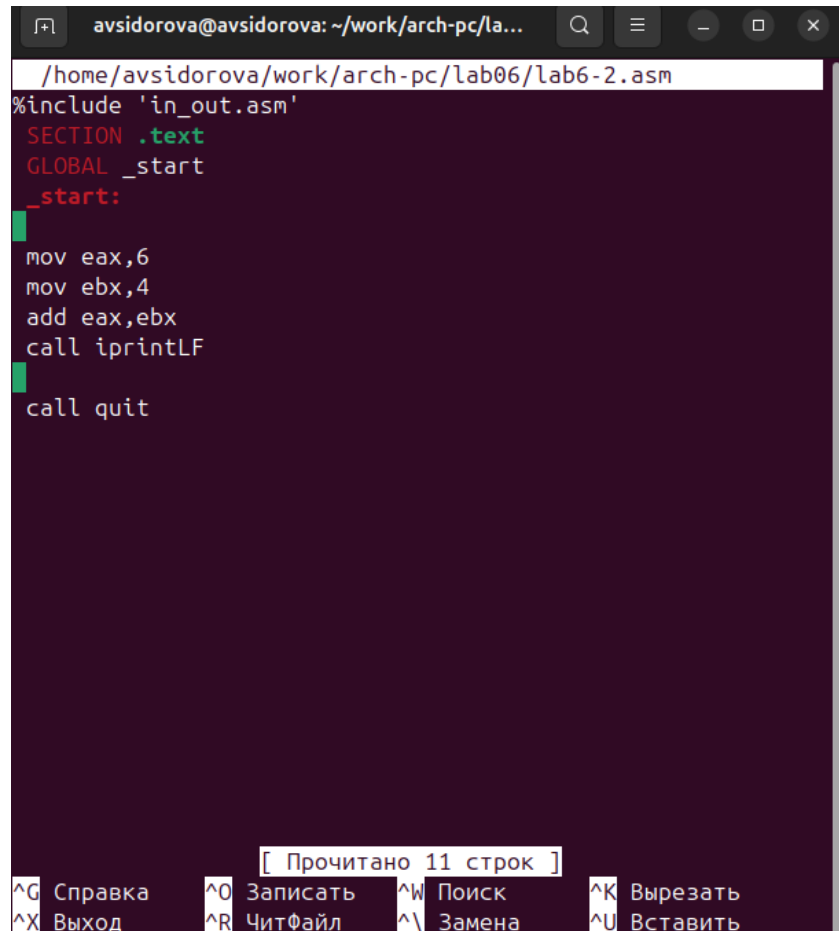

```

avsidorova@avsidorova:~/work/arch-pc/lab06$ nasm -f elf lab
6-2.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$ ld -m elf_i386
-o lab6-2 lab6-2.o
avsidorova@avsidorova:~/work/arch-pc/lab06$ ./lab6-2
106
avsidorova@avsidorova:~/work/arch-pc/lab06$

```

Рис. 2.8: Смотрим на работу программы

Аналогично предыдущему примеру изменим символы на числа. (рис. fig. 2.9).



```

/home/avsidorova/work/arch-pc/lab06/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

```

[Прочитано 11 строк]

^G Справка ^O Записать ^W Поиск ^K Вырезать
 ^X Выход ^R ЧитФайл ^\ Замена ^U Вставить

Рис. 2.9: Изменяем файл

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

```

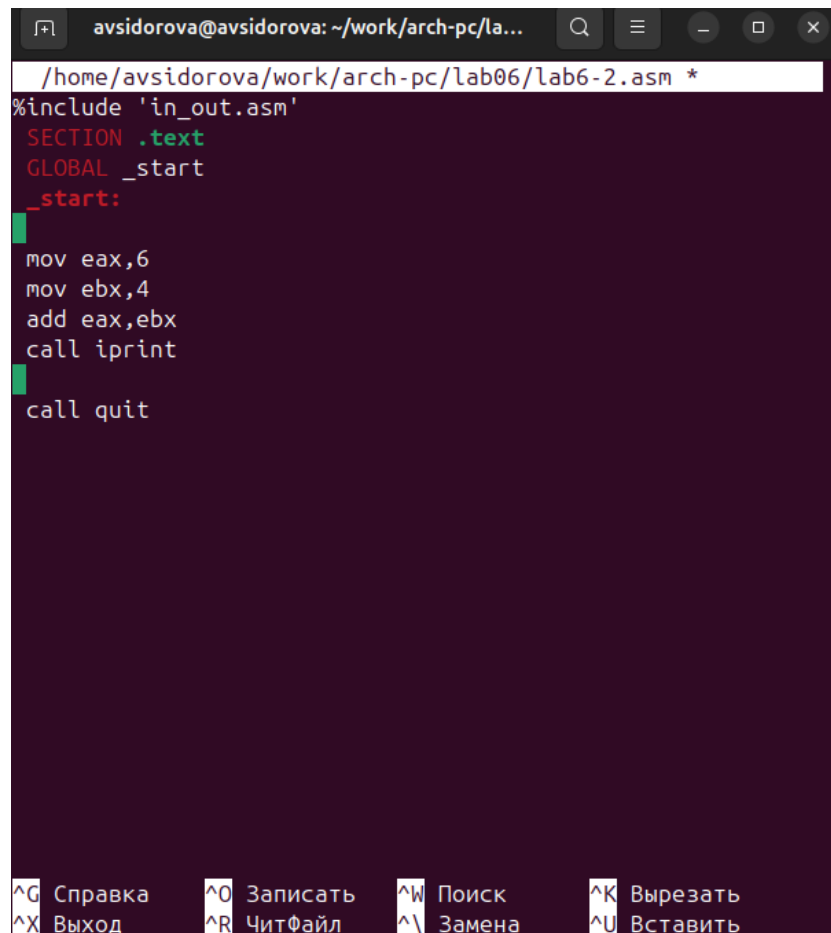
avsidorova@avsidorova:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
avsidorova@avsidorova:~/work/arch-pc/lab06$ ./lab6-2
10
avsidorova@avsidorova:~/work/arch-pc/lab06$

```

Рис. 2.10: Смотрим на работу программы

В результате программы получаем число 10

Заменяем функцию `iprintLF` на `iprint`. (рис. fig. 2.11).



```

/home/avsidorova/work/arch-pc/lab06/lab6-2.asm *
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit

```

[^]G Справка [^]O Записать [^]W Поиск [^]K Вырезать
[^]X Выход [^]R ЧитФайл [^]\ Замена [^]U Вставить

Рис. 2.11: Изменяем файл

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

```

avsidorova@avsidorova:~/work/arch-pc/lab06$ nasm -f elf lab
6-2.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$ ld -m elf_i386
-o lab6-2 lab6-2.o
avsidorova@avsidorova:~/work/arch-pc/lab06$ ./lab6-2
10avsidorova@avsidorova:~/work/arch-pc/lab06$

```

Рис. 2.12: Смотрим на работу программы

Вывод функций `iprintLF` и `iprint` отличаются тем, что `iprintLF` делает перенос на новую строку.

2.2 2.Выполнение арифметических операций в NASM

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

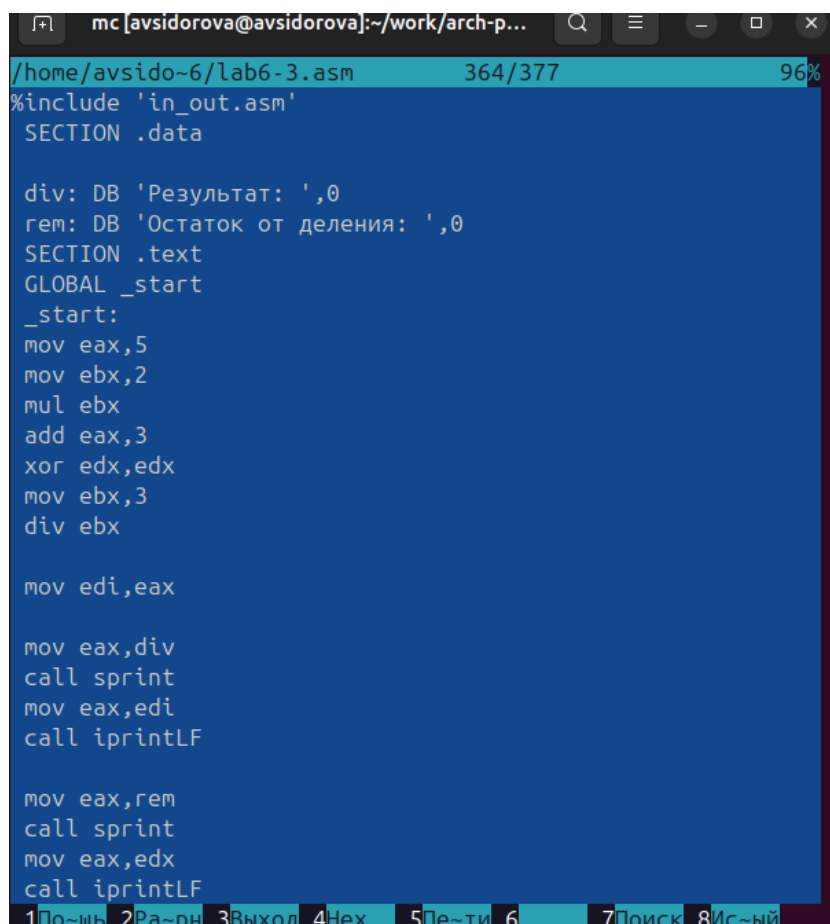
```

avsidorova@avsidorova:~/work/arch-pc/lab06$ touch ~/work/ar
ch-pc/lab06/lab6-3.asm
avsidorova@avsidorova:~/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
avsidorova@avsidorova:~/work/arch-pc/lab06$

```

Рис. 2.13: Создаем файл и проверяем

Внимательно изучим текст программы из листинга 6.3 и введем в `lab6-3.asm`. (рис. fig. 2.14).



```
mc [avsidorova@avsidorova]:~/work/arch-p...
/home/avsidorova~6/lab6-3.asm 364/377 96%
#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

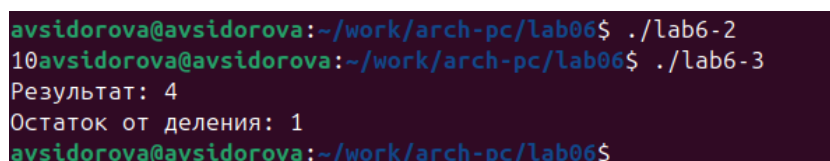
mov edi,eax

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

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

Рис. 2.14: Вводим программу

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



```
avsidorova@avsidorova:~/work/arch-pc/lab06$ ./lab6-2
10avsidorova@avsidorova:~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
avsidorova@avsidorova:~/work/arch-pc/lab06$
```

Рис. 2.15: Смотрим на результат работы программы

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

```

%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

```

Рис. 2.16: Изменяем файл

Создадим исполняемый файл и проверим его работу. (рис. fig. 2.17).

```

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

```

Рис. 2.17: Смотрим на результат работы программы

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

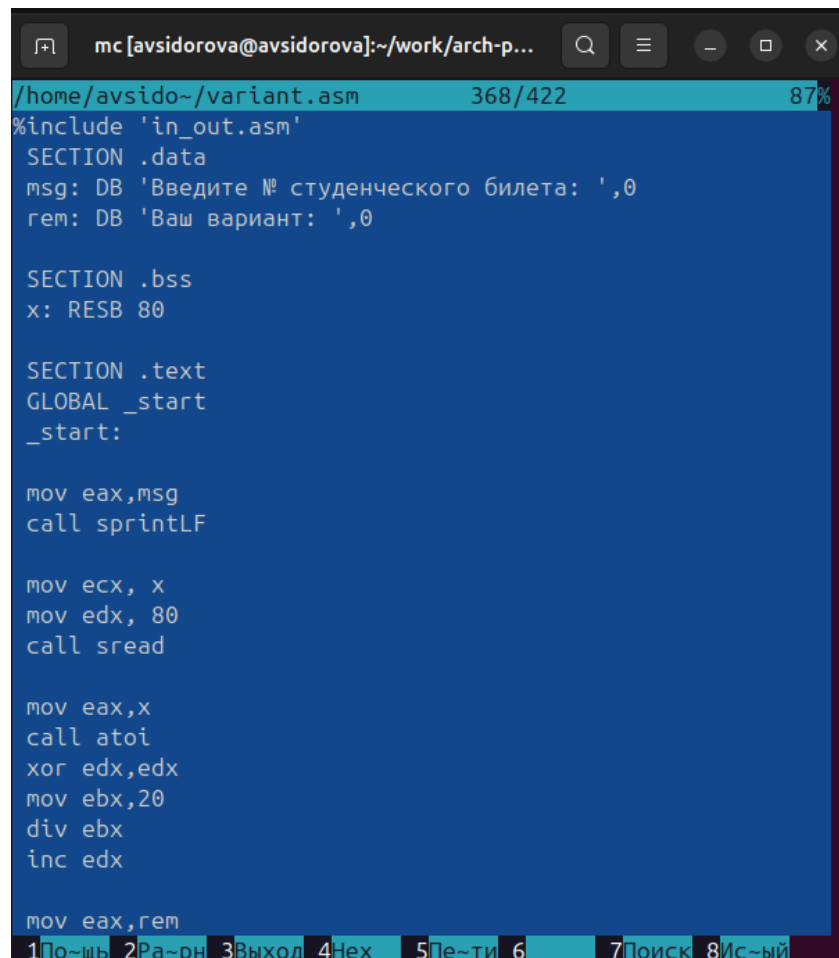
```

[18]# Остановлен mc
avsidorova@avsidorova:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1.o    lab6-2.o    lab6-3.o
lab6-1      lab6-2      lab6-3      variant.asm
lab6-1.asm  lab6-2.asm  lab6-3.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$

```

Рис. 2.18: Создаем файл

Внимательно изучим текст программы из листинга 6.4 и введем в файл variant.asm.(рис. fig. 2.19).



```

/home/avsidorova/variant.asm 368/422 87%
#include 'in_out.asm'
SECTION .data
msg: DB 'Введите № студенческого билета: ',0
gem: 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
call atoi
xor edx,edx
mov ebx,20
div ebx
inc edx

mov eax,gem

```

Рис. 2.19: Вводим программу

Создадим исполняемый файл и запустим его. (рис. fig. 2.20).

```
avsidorova@avsidorova:~/work/arch-pc/lab06$ nasm -f elf variant.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
avsidorova@avsidorova:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132242912
Ваш вариант: 13
avsidorova@avsidorova:~/work/arch-pc/lab06$
```

Рис. 2.20: Проверим результат работы программы

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? Строки “mov eax,rem”, “call sprint”
2. Для чего используются следующие инструкции? mov ecx, x mov edx, 80 call sread

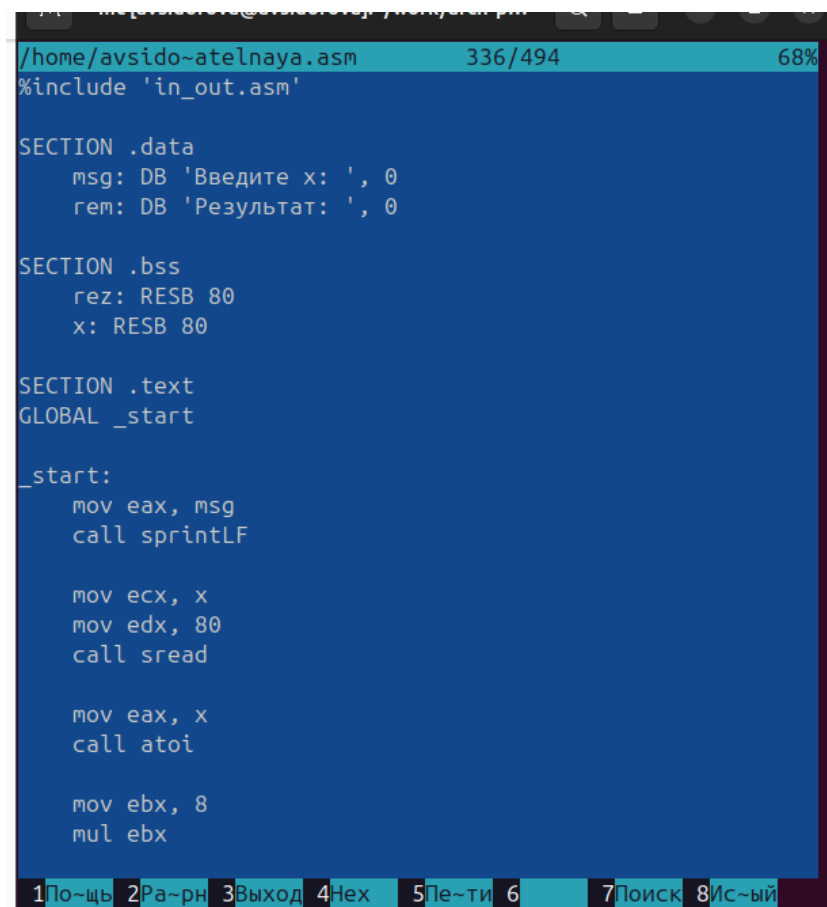
Эти инструкции вместе подготавливают необходимые параметры для вызова функции sread, которая затем считывает строку, введенную пользователем, и сохраняет ее в переменной x.

3. Для чего используется инструкция “call atoi”? call atoi используется для преобразования строки, введенной пользователем, в целое число
4. Какие строки листинга 6.4 отвечают за вычисления варианта? 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 eax на значение регистра ebx с сохранением частного в регистре eax и остатка в регистре edx.
5. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”? В регистр edx.
6. Для чего используется инструкция “inc edx”? Для увеличения значения в регистре edx на 1. В нашем коде она увеличивает остаток от деления на 1

7. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений? Строка “mov eax,edx” передает значение остатка от деления в регистр eax. Строка “call iprintLF” вызывает процедуру iprintLF для вывода значения на экран вместе с переводом строки.

2.3 Задание для самостоятельной работы

Создаем файл samostoyatel'naya.asm Открываем его и пишем программу для решения выражения $f(x) = (8x+6)*10$ (рис. fig. 2.21 fig. 2.22)



```
/home/avsidon~atel'naya.asm 336/494 68%
#include 'in_out.asm'

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

SECTION .bss
    rez: RESB 80
    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
```

Рис. 2.21: Пишем программу


```
/home/avsido~atelnaya.asm 494/494 100%
call sprintLF

mov ecx, x
mov edx, 80
call sread

mov eax, x
call atoi

mov ebx, 8
mul ebx

add eax, 6

mov ebx, 10
mul ebx

mov [rez], eax

mov eax, rem
call sprintLF

mov eax, [rez]
call iprintLF

call quit

1По-щъ 2Ра-рн 3Выход 4Нех 5Пе-ти 6 7Поиск 8Ис-ый
```

Рис. 2.22: Пишем программу

Компилируем программу и проверяем работу(рис. fig. 2.23).

```
avsidorova@avsidorova:~/work/arch-pc/lab06$ nasm -f elf samostoyatelnaya.asm
avsidorova@avsidorova:~/work/arch-pc/lab06$ ld -m elf_i386 -o samostoyatelnaya samostoyatelnaya.o
avsidorova@avsidorova:~/work/arch-pc/lab06$ ./samostoyatelnaya
Введите x:
1
Результат:
140
avsidorova@avsidorova:~/work/arch-pc/lab06$ mc
[25]+ Остановлен mc
avsidorova@avsidorova:~/work/arch-pc/lab06$ ./samostoyatelnaya
Введите x:
4
Результат:
380
avsidorova@avsidorova:~/work/arch-pc/lab06$
```

Рис. 2.23: Проверяем работу программы

3 Выводы

Освоили арифметические инструкции языка ассемблера NASM.