

Отчёт по лабораторной работе №6

дисциплина: архитектура компьютеров

Ведьмина Александра Сергеевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выполнение заданий для самостоятельной работы	17
6	Выводы	20

Список иллюстраций

4.1	Создание файла lab6-1.asm	8
4.2	Ввод программы в lab6-1.asm	8
4.3	Запуск файла lab6-1	9
4.4	Замена строк в lab6-1.asm	9
4.5	Запуск файла lab6-1	10
4.6	Создание файла lab6-2.asm	10
4.7	Ввод программы в lab6-2.asm	11
4.8	Запуск файла lab6-2	11
4.9	Замена строк в файле lab6-2.asm	12
4.10	Запуск файла lab6-2	12
4.11	Запуск изменённого файла lab6-2	12
4.12	Ввод программы в файл lab6-3.asm	13
4.13	Запуск файла lab6-3	13
4.14	Изменение текста программы lab6-3.asm	14
4.15	Запуск файла lab6-3	14
4.16	Создание файла variant.asm	14
4.17	Ввод программы variant.asm	15
4.18	Запуск файла variant	15
5.1	Программа для вычисления функции	18
5.2	Запуск файла sumrub при $x = 4$	18
5.3	Запуск файла sumrub при $x = 10$	19

Список таблиц

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Задание

1. Ознакомиться с теорией
2. Изучить простые арифметические операции NASM.
3. Выполнить задания для самостоятельной работы.

3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда - это место, где хранятся данные.

Способы адресации: 1. Регистровая 2. Непосредственная 3. Адресация памяти

Команда целочисленного сложения `add` (от англ. `addition` - добавление) выполняет сложение двух операндов и записывает результат по адресу первого операнда. `sub` отвечает за вычитание.

Для команд умножения один из сомножителей указывается в команде и должен находиться в регистре или в памяти, но не может быть непосредственным операндом. Второй сомножитель в команде явно не указывается и должен находиться в регистре `EAX, AX` или `AL`, а результат помещается в регистры `EDX:EAX, DX:AX` или `AX`, в зависимости от размера операнда.

Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от `American Standard Code for Information Interchange` (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом.

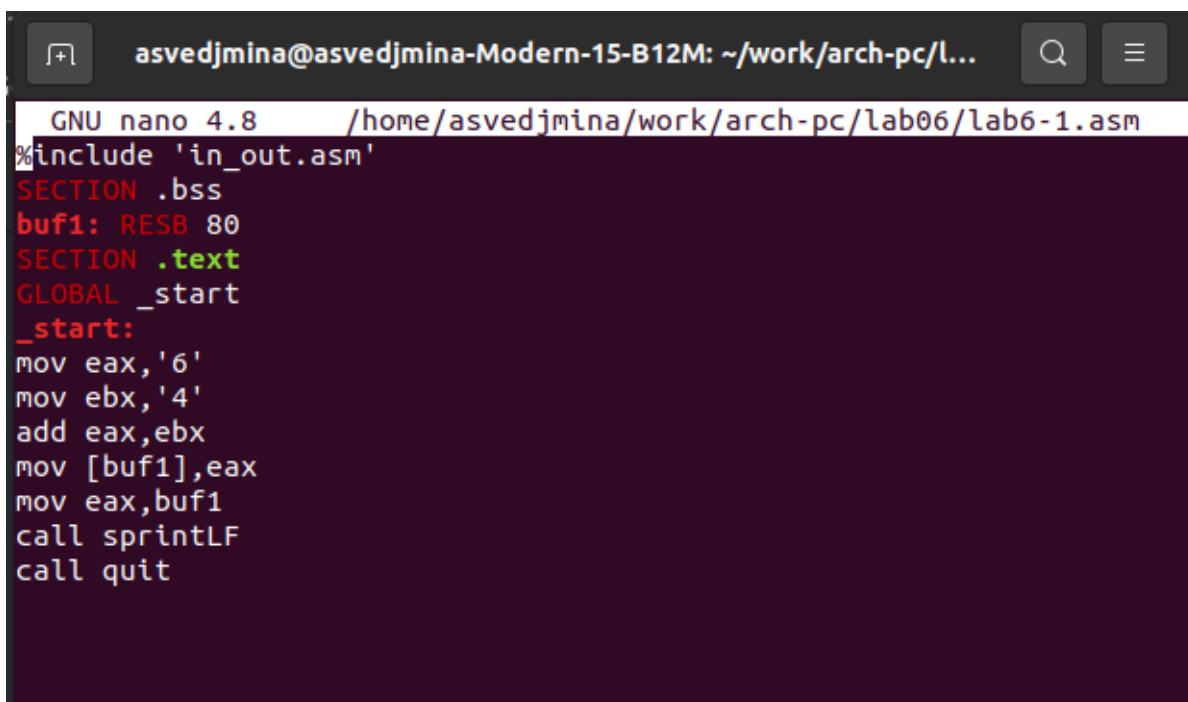
4 Выполнение лабораторной работы

Создаю каталог lab6, перехожу в него и создаю файл lab6-1.asm.

```
asvedjmina@asvedjmina-Modern-15-B12M:~$ cd ~/work/arch-pc/lab06
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ touch lab6-1.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.1: Создание файла lab6-1.asm

Ввожу в данный файл текст требуемой программы.



```
GNU nano 4.8 /home/asvedjmina/work/arch-pc/lab06/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
```

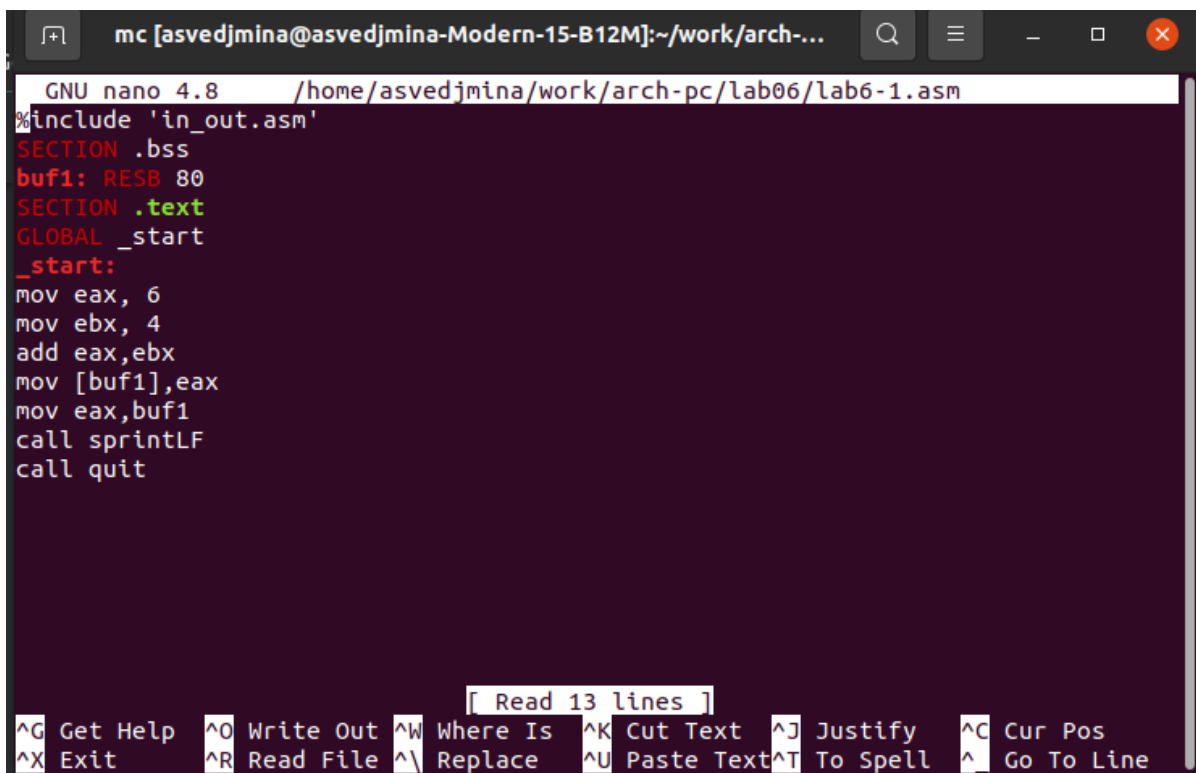
Рис. 4.2: Ввод программы в lab6-1.asm

Создаю исполняемый файл и запускаю его.

```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./lab6-1
j
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.3: Запуск файла lab6-1

Затем в тексте программы заменяю строки `mov eax, '6'`, `mov ebx, '4'` на строки `mov eax, 6`, `mov ebx, 4`.



```
GNU nano 4.8 /home/asvedjmina/work/arch-pc/lab06/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 sprintf
call quit
```

Рис. 4.4: Замена строк в lab6-1.asm

Создаю исполняемый файл и запускаю его.

```

asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ mc
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1.o
ld: no input files
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./lab6-1
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ █

```

Рис. 4.5: Запуск файла lab6-1

Символ, полученный в результате программы, не отображается. По таблице ASCII коду 10 соответствует символ, изображающий круг в квадрате.

Создаю файл lab6-2.asm и ввожу в него предложенный текст программы.

```

asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-2.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ █

```

Рис. 4.6: Создание файла lab6-2.asm

```
asvedjmina@asvedjmina-Modern-15-B12M: ~/work/arch-pc/lab06
GNU nano 4.8 /home/asvedjmina/work/arch-pc/lab06/lab6
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
```

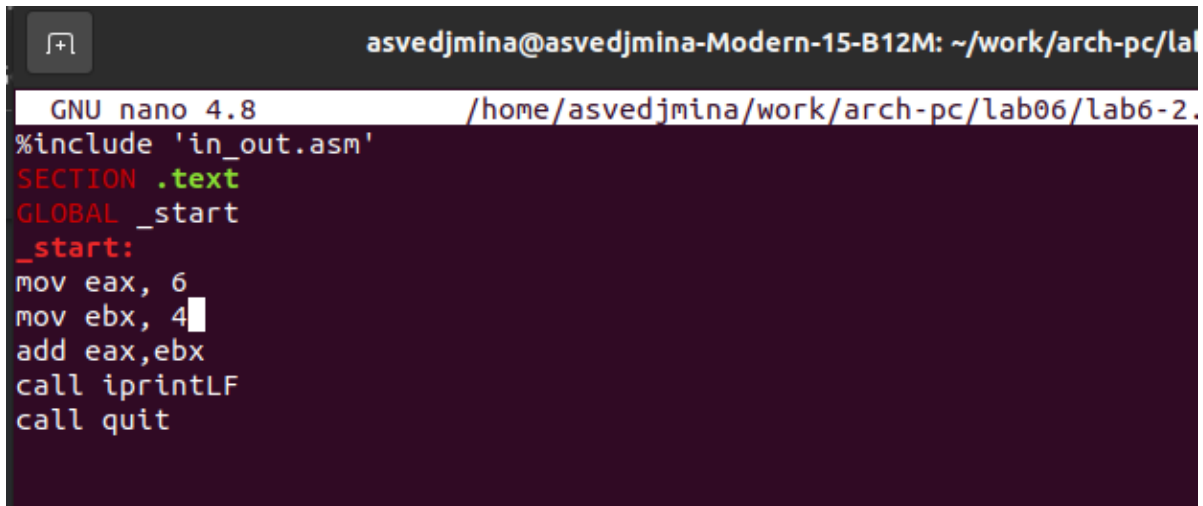
Рис. 4.7: Ввод программы в lab6-2.asm

Создаю исполняемый файл и запускаю его.

```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./lab6-2
106
start.
```

Рис. 4.8: Запуск файла lab6-2

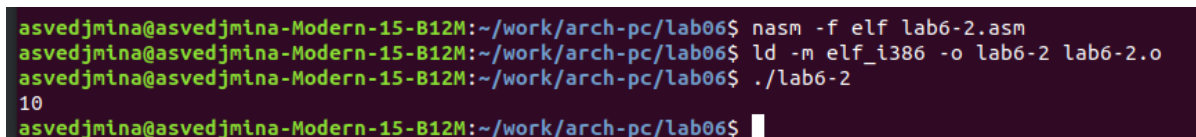
Заменяю `mov eax,'6'` и `mov ebx,'4'` на `mov eax,6` и `mov ebx,4` в файле lab6-2.asm.



```
GNU nano 4.8 /home/asvedjmina/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
```

Рис. 4.9: Замена строк в файле lab6-2.asm

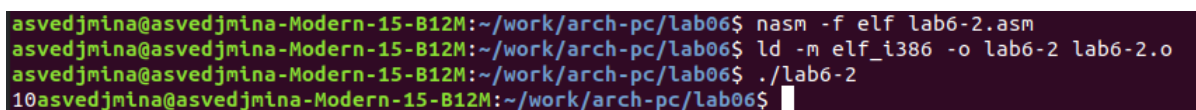
Создаю исполняемый файл и запускаю его.



```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./lab6-2
10
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.10: Запуск файла lab6-2

При исполнении программы получен результат 10. Заменяю функцию iprintLF на iprint в файле lab6-2.asm, затем создаю исполняемый файл и запускаю его.



```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./lab6-2
10asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.11: Запуск изменённого файла lab6-2

Функция iprint в отличие от iprintLF предлагает нам ввести следующую команду на той же строке, на которой был выведен результат выполнения программы.

Далее создаю файл lab6-3.asm и ввожу программу вычисления выражения $f(x)=(5*2+3)/3$.

```
GNU nano 4.8 /home/asvedjmina/work/arch-pc/lab06/lab6-3.asm Modified
%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

^G Get Help      ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos
^X Exit          ^R Read File    ^_ Replace      ^U Paste Text   ^T To Spell     ^_ Go To Line
```

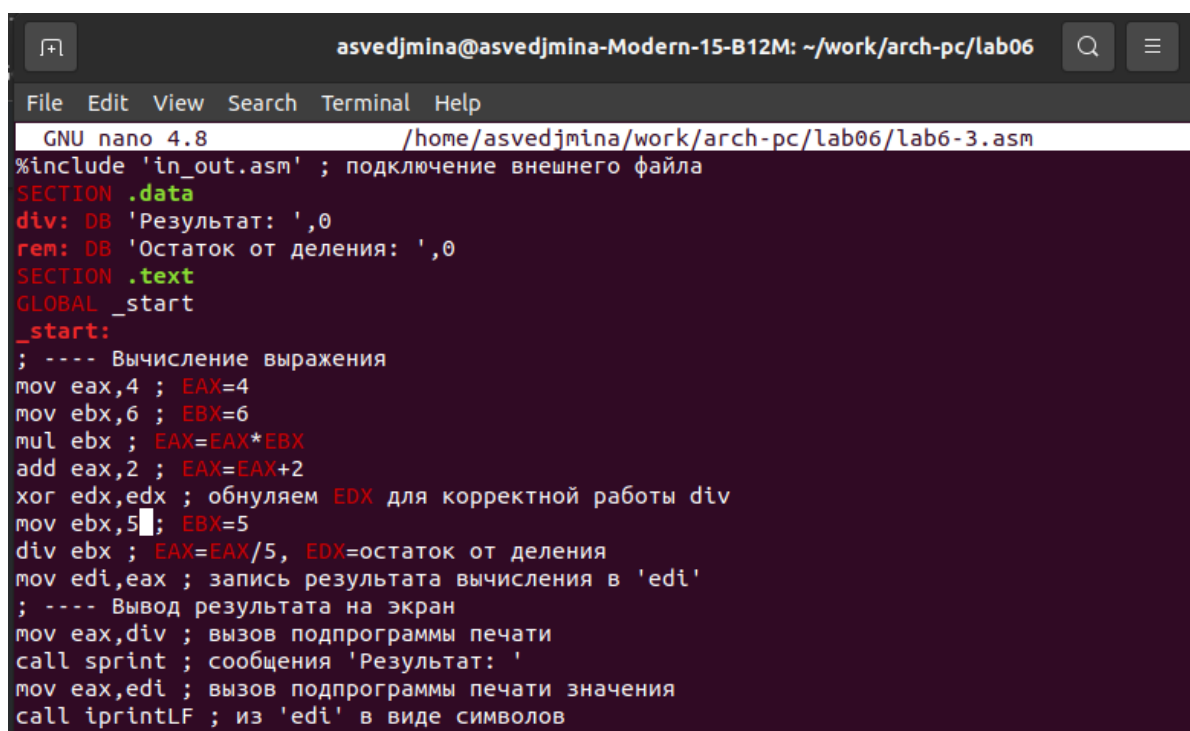
Рис. 4.12: Ввод программы в файл lab6-3.asm

Создаю исполняемый файл и запускаю его.

```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.13: Запуск файла lab6-3

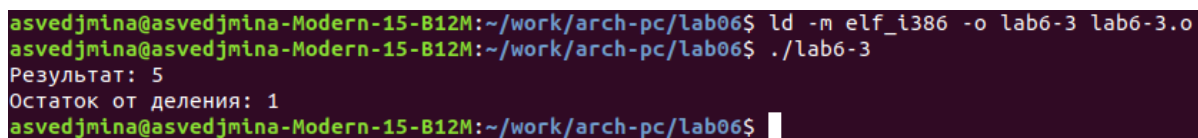
Изменяю текст программы так, чтобы она вычисляла выражение $f(x)=(4*6+2)/5$.



```
GNU nano 4.8 /home/asvedjmina/work/arch-pc/lab06/lab6-3.asm
File Edit View Search Terminal Help
#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=4
mov ebx,6 ; EBX=6
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+2
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=5
div ebx ; EAX=EAX/5, 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
```

Рис. 4.14: Изменение текста программы lab6-3.asm

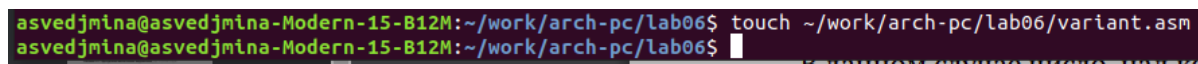
Создаю исполняемый файл и проверяю его работу.



```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.15: Запуск файла lab6-3

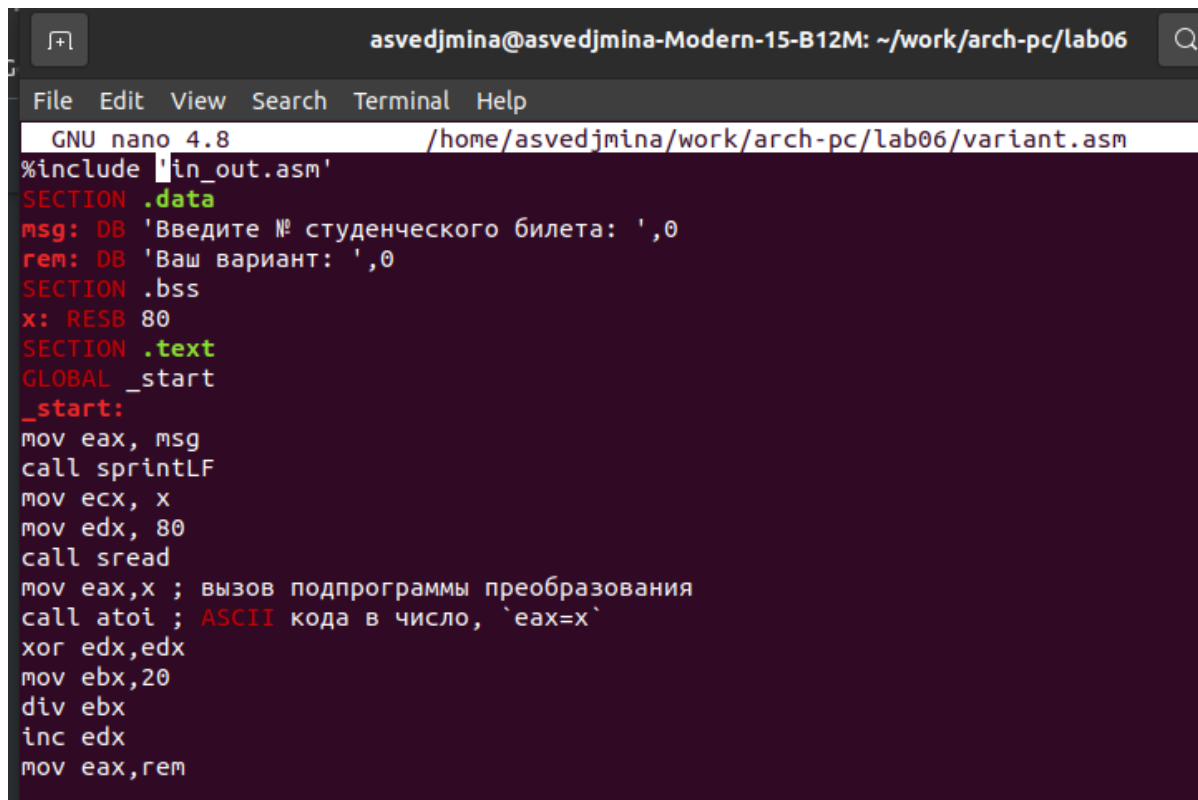
Создаю файл variant.asm.



```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.16: Создание файла variant.asm

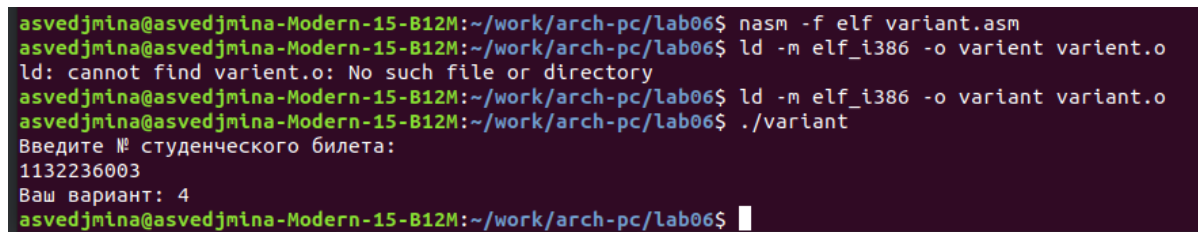
Ввожу в данный файл программу для вычисления варианта задания.



```
GNU nano 4.8 /home/asvedjmina/work/arch-pc/lab06/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 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
```

Рис. 4.17: Ввод программы variant.asm

Создаю исполняемый файл и запускаю его.



```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ nasm -f elf variant.asm
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
ld: cannot find variant.o: No such file or directory
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132236003
Ваш вариант: 4
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 4.18: Запуск файла variant

Ответы на вопросы по листингу 6.4:

1. За вывод на экран сообщения “Ваш вариант:” в листинге 6.4 отвечают строки `mov eax, rem` и `call sprint`.

2. `mov ecx,x` прокладывает адрес введённой строки в `ecx`; `mov edx,80` обозначает запись в регистр `edx`; `call sread` используется для вызова подпрограммы, которая считывает текст с клавиатуры.
3. `call atoi` нужна для вызова подпрограммы, которая преобразовывает код символа из таблицы ASCII в сам символ и записывает его в `eax`.
4. За вычисление варианта отвечают строки `xor edx, edx`; `mov ebx,20`; `div ebx`; `inc edx`.
5. Остаток от деления от `div ebx` записывается в `edx`.
6. `inc edx` увеличивает значение регистра на один.
7. За вывод на экран результата вычислений отвечают `mov eax, edx` и `call iprintLF`.

5 Выполнение заданий для самостоятельной работы

Номер моего варианта - 4, поэтому я буду реализовывать функцию $f(x) = 4/3(x-1)+5$. Для выполнения задания создаю файл `sumrub.asm` и записываю в него необходимую программу.

```
asvedjmina@asvedjmina-Modern-15-B12M: ~/work/arch-pc/lab06
GNU nano 4.8 /home/asvedjmina/work/arch-pc/lab06/sumrub.asm
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите значение x',0
rem: DB 'Результат: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprint
mov ecx, x
mov edx, 80
call sread
mov eax, x
call atoi
add eax, -1
mov ebx, 4
mul ebx
mov ebx, 3
div ebx
add eax, 5
mov edi, eax
mov eax, rem
call sprint
mov eax, edi
call iprint
call quit
```

Рис. 5.1: Программа для вычисления функции

Создаю исполняемый файл и запускаю программу.

```
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ld -m elf_i386 -o sumrub sumrub.o
asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./sumrub
Введите значение x4
Результат: 9asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$
```

Рис. 5.2: Запуск файла sumrub при $x = 4$

```
Результат: 9asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ ./sumrub  
Введите значение x10  
Результат: 17asvedjmina@asvedjmina-Modern-15-B12M:~/work/arch-pc/lab06$ █
```

Рис. 5.3: Запуск файла sumrub при $x = 10$

6 Выводы

В ходе выполнения лабораторной работы я освоила арифметические операции в NASM.