

Отчёт по лабораторной работе 6

Архитектура компьютера

Довран Илиев

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	22

Список иллюстраций

2.1	Программа lab6-1.asm	7
2.2	Запуск программы lab6-1.asm	7
2.3	Программа lab6-1.asm	8
2.4	Запуск программы lab6-1.asm	9
2.5	Программа lab6-2.asm	10
2.6	Запуск программы lab6-2.asm	10
2.7	Программа lab6-2.asm	11
2.8	Запуск программы lab6-2.asm	12
2.9	Программа lab6-2.asm	12
2.10	Запуск программы lab6-2.asm	13
2.11	Программа lab6-3.asm	14
2.12	Запуск программы lab6-3.asm	14
2.13	Программа lab6-3.asm	15
2.14	Запуск программы lab6-3.asm	16
2.15	Программа variant.asm	17
2.16	Запуск программы variant.asm	17
2.17	Программа prog.asm	20
2.18	Запуск программы prog.asm	21

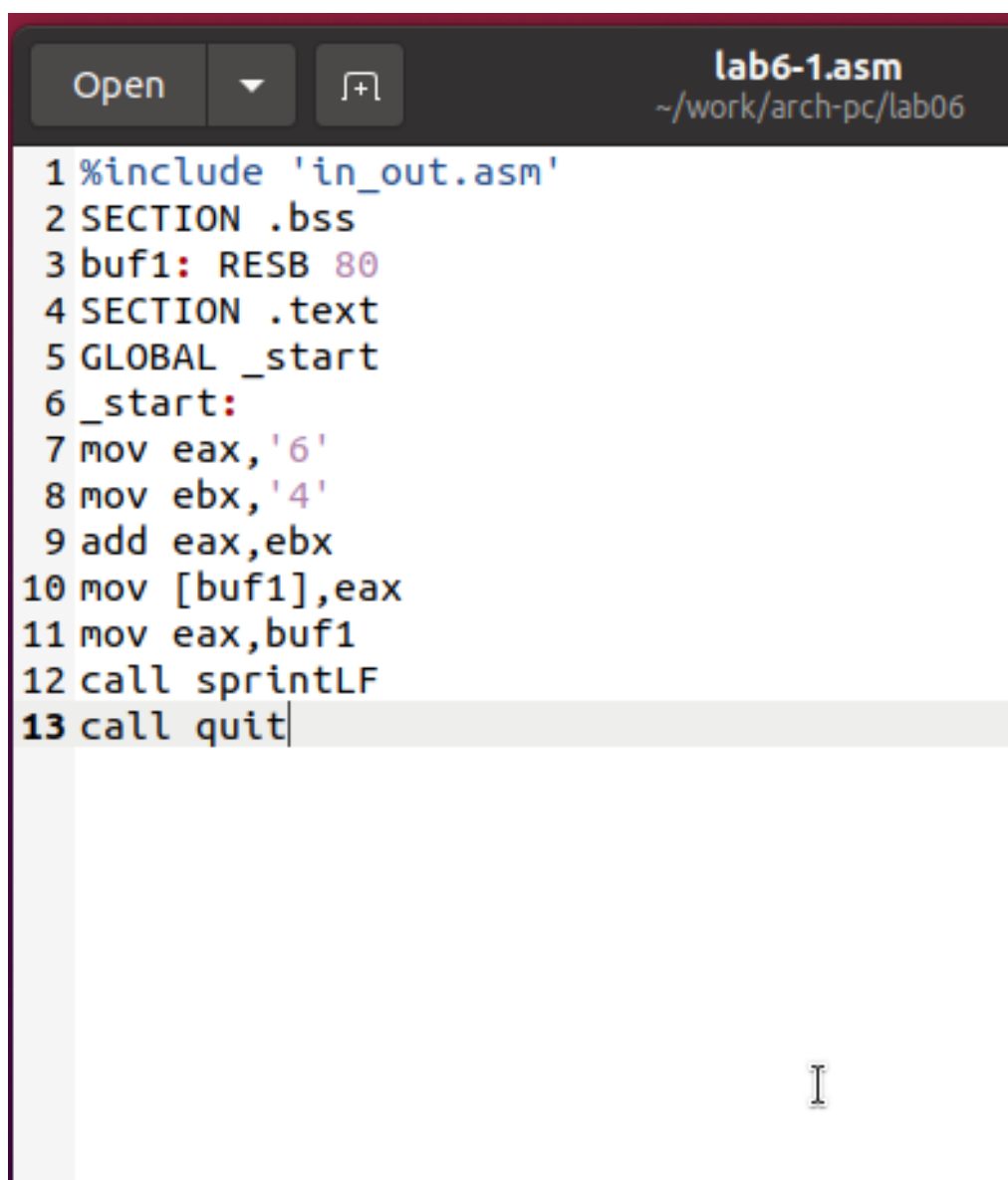
Список таблиц

1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

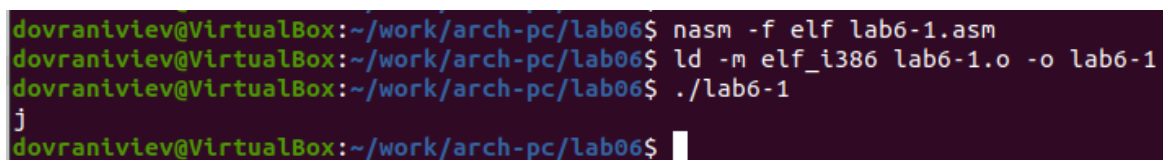
1. Установил директорию для файлов лабораторного занятия № 6, переместился в эту директорию и создал файл с именем lab6-1.asm.
2. Проанализируем примеры программ, которые осуществляют вывод символов и чисел. Эти программы будут отображать данные, помещенные в регистры.



```
lab6-1.asm
~/work/arch-pc/lab06

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

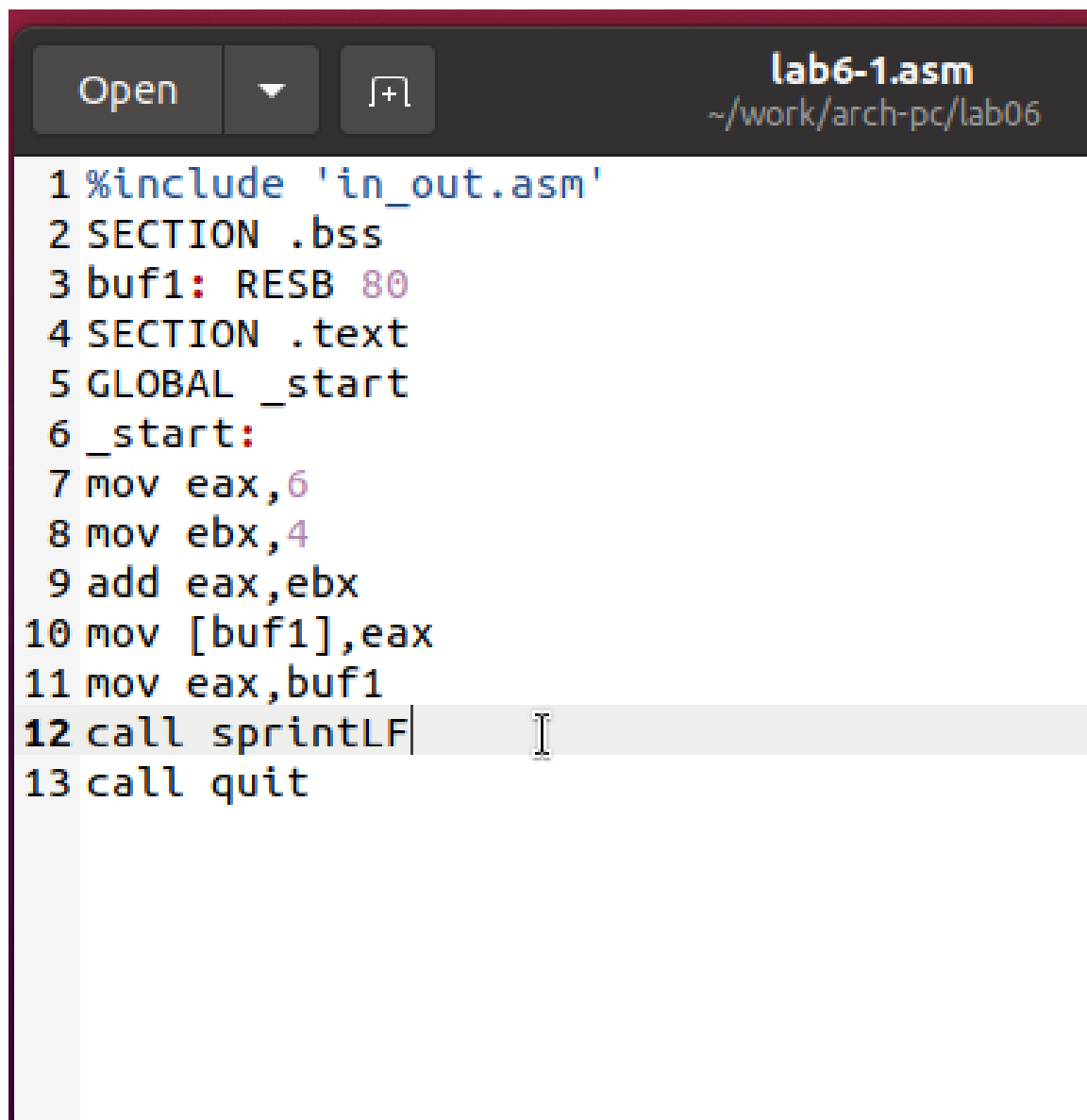
Рис. 2.1: Программа lab6-1.asm



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

Рис. 2.2: Запуск программы lab6-1.asm

- Затем модифицирую код программы, подставляя числа вместо символов в регистры.



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

Рис. 2.3: Программа lab6-1.asm

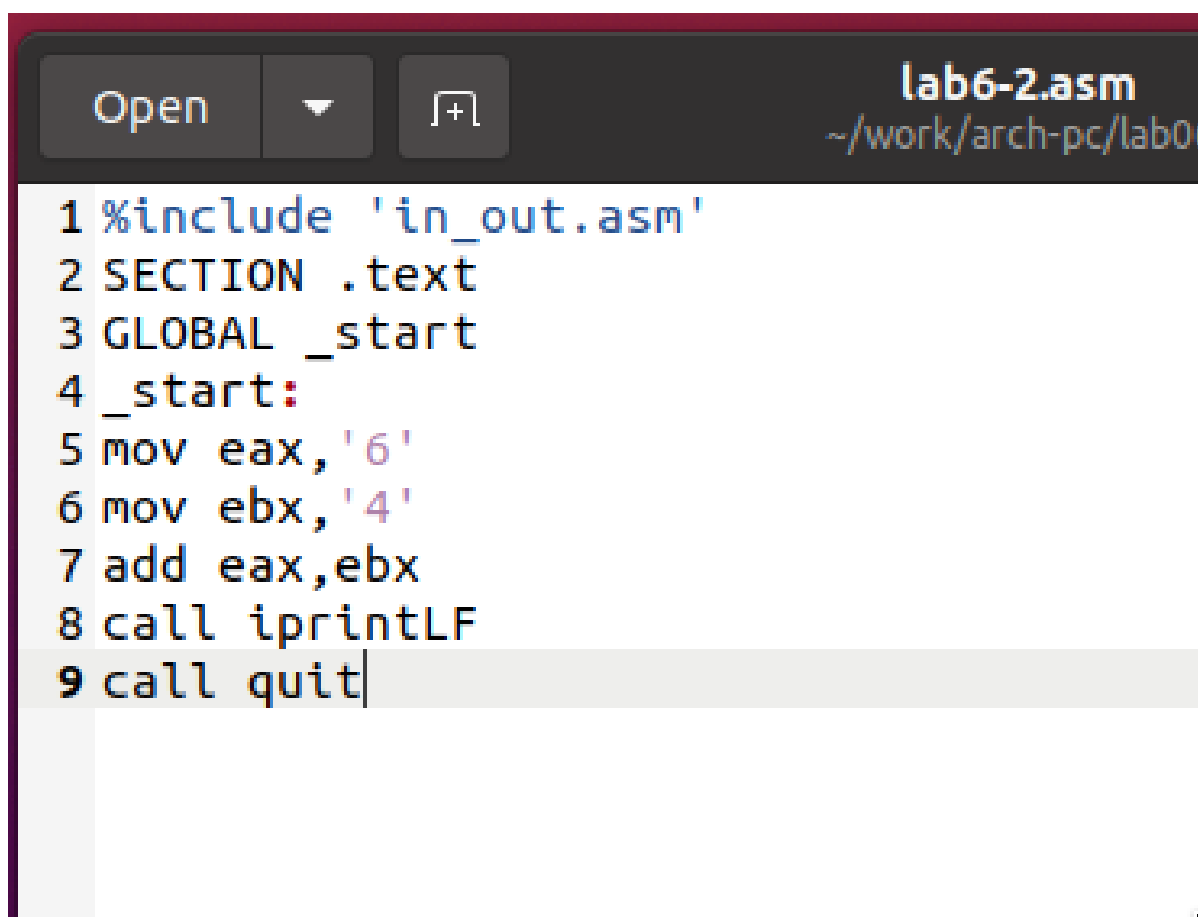

```
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-1.o -o lab6-1
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ./lab6-1

dovraniviev@VirtualBox:~/work/arch-pc/lab06$ █
```

Рис. 2.4: Запуск программы lab6-1.asm

Хотя символ не отображается на экране, он присутствует. Это символ перевода строки LF.

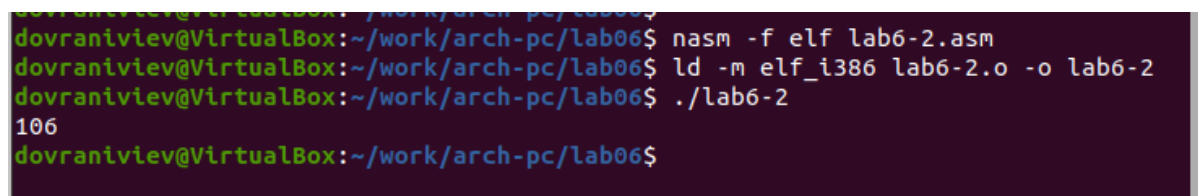
4. Как упоминалось ранее, для манипуляций с числами в файле in_out.asm предусмотрены специальные подпрограммы, которые преобразуют ASCII символы в числовые значения и наоборот. Применил эти функции для преобразования кода программы



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

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

Рис. 2.5: Программа lab6-2.asm

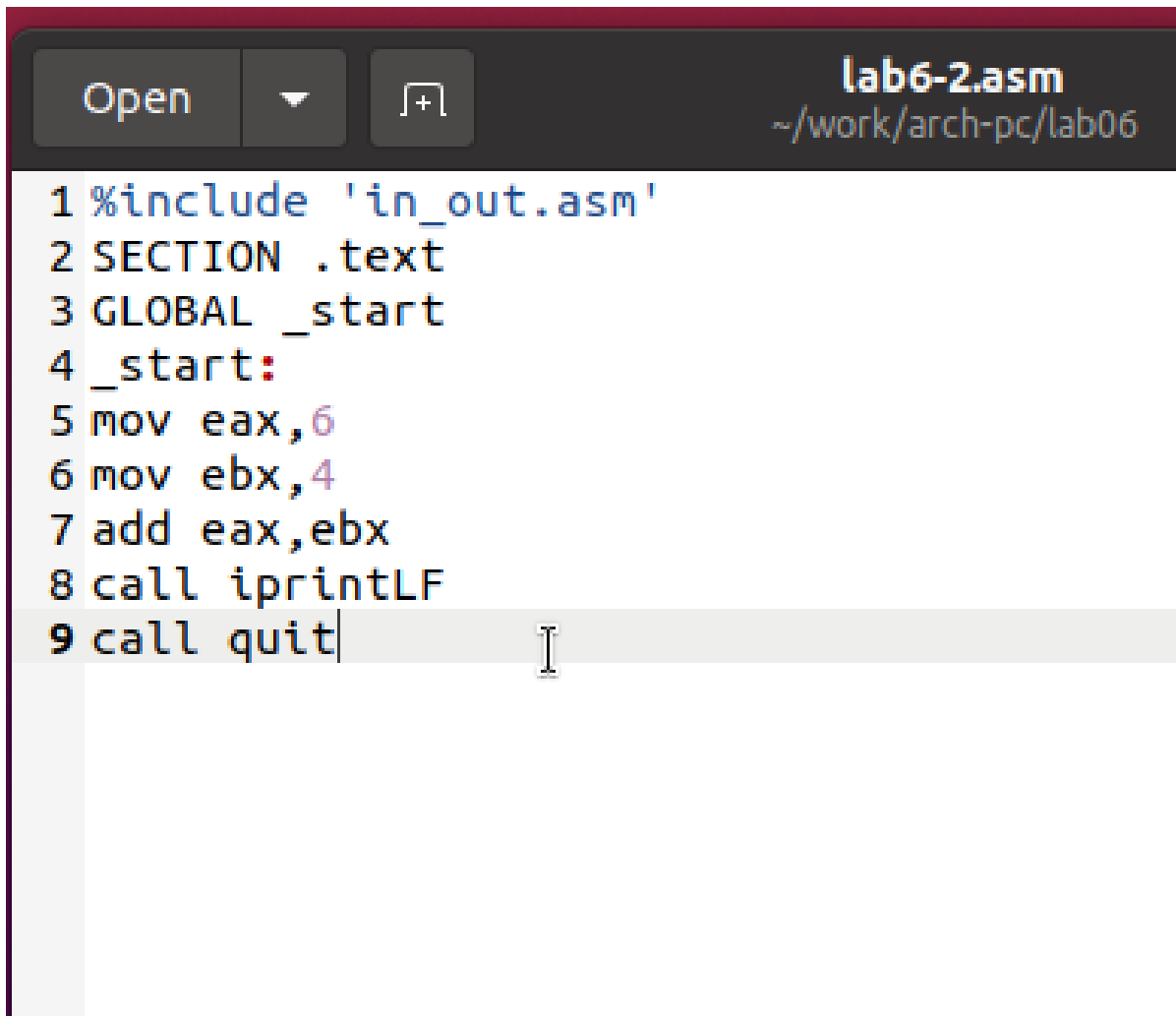


```
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-2.o -o lab6-2
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ./lab6-2
106
dovraniviev@VirtualBox:~/work/arch-pc/lab06$
```

Рис. 2.6: Запуск программы lab6-2.asm

В результате выполнения программы получаем значение 106. В этом случае, как и в первом примере, команда `add` суммирует коды символов '6' и '4' ($54+52=106$). Но в этот раз, в отличие от предыдущей программы, функция `iprintLF` позволяет отобразить число, а не символ, ASCII код которого соответствует этому числу.

5. Подобно предшествующему случаю, произведем замену символов на числовые значения.



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

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

Рис. 2.7: Программа lab6-2.asm

Функция `iprintLF` дает возможность отобразить число, при этом операнды были числами, а не ASCII кодами символов. Таким образом, результатом является число 10.

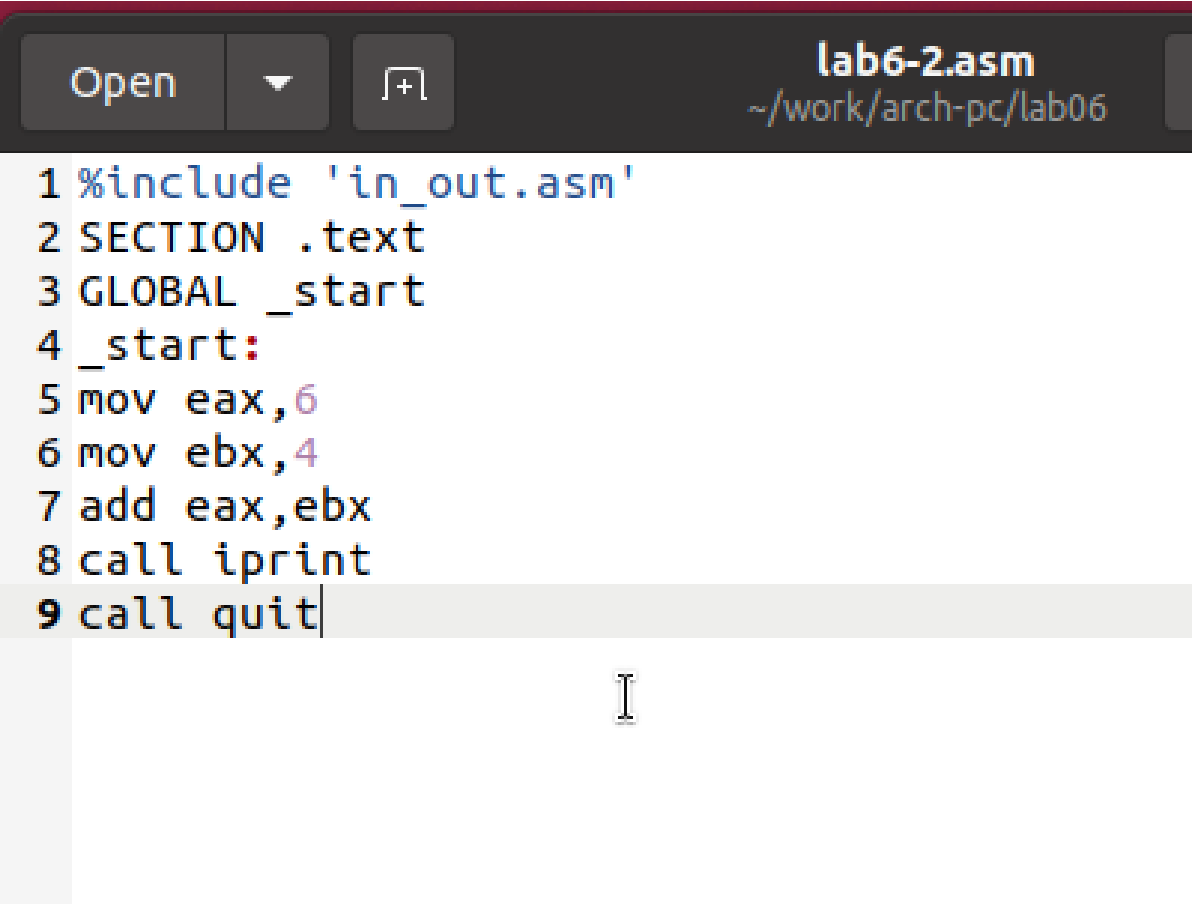
```

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

```

Рис. 2.8: Запуск программы lab6-2.asm

Замена функции `iprintLF` на `iprint` приводит к изменению вывода: отсутствует перевод строки.



```

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

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

```

Рис. 2.9: Программа lab6-2.asm

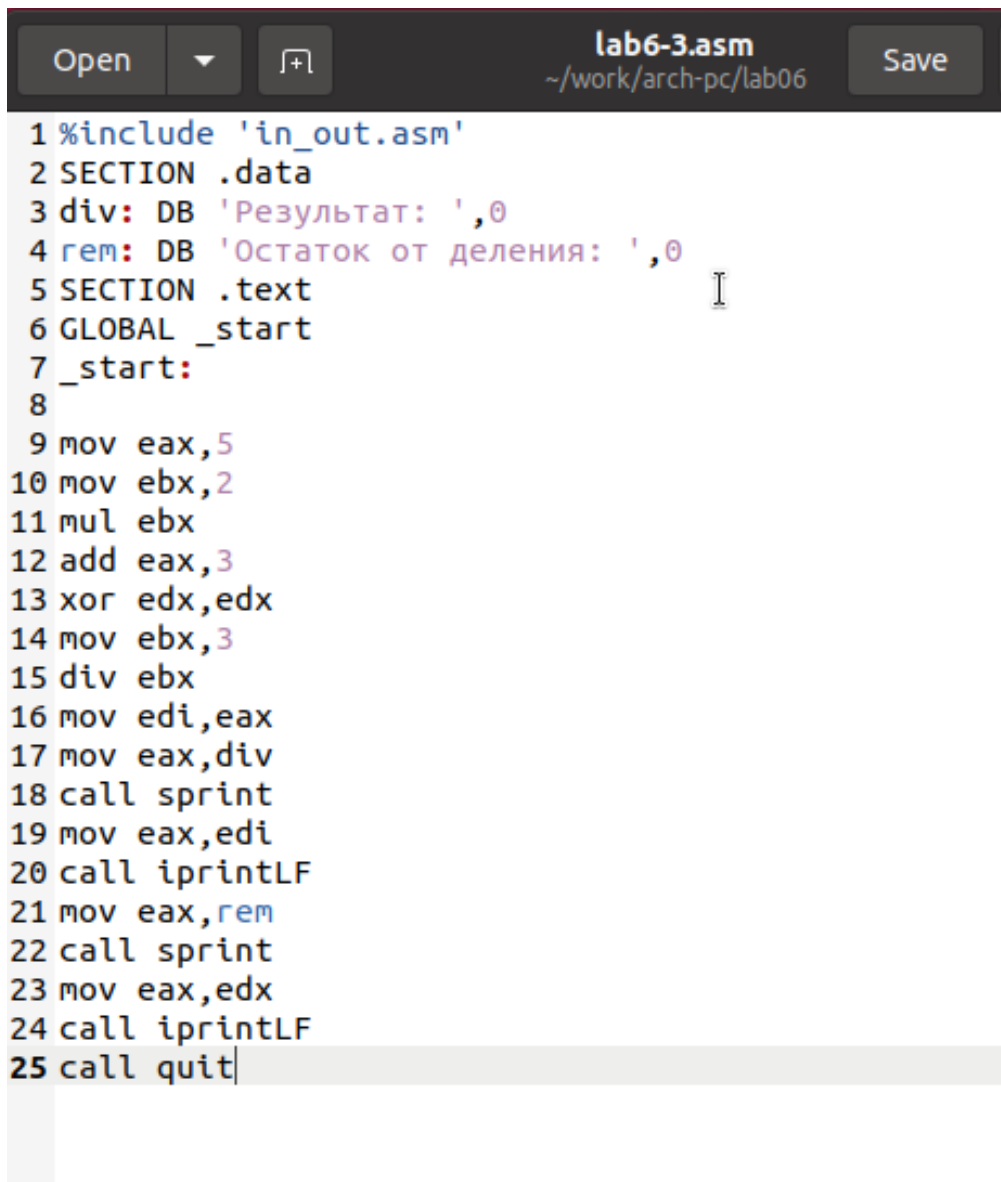
```
dovraniviev@VirtualBox:~/work/arch-pc/lab06$  
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm  
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-2.o -o lab6-2  
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ./lab6-2  
10dovraniviev@VirtualBox:~/work/arch-pc/lab06$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

Рис. 2.10: Запуск программы lab6-2.asm

6. В качестве иллюстрации выполнения арифметических операций в NASM представим программу, которая вычисляет арифметическое выражение

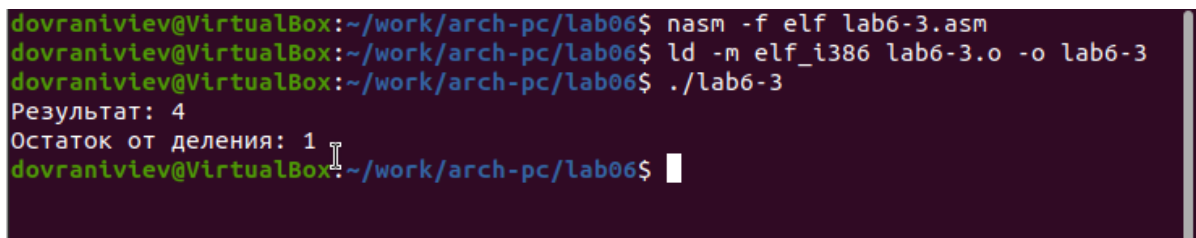
$$f(x) = (5 * 2 + 3) / 3$$

.



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

Рис. 2.11: Программа lab6-3.asm



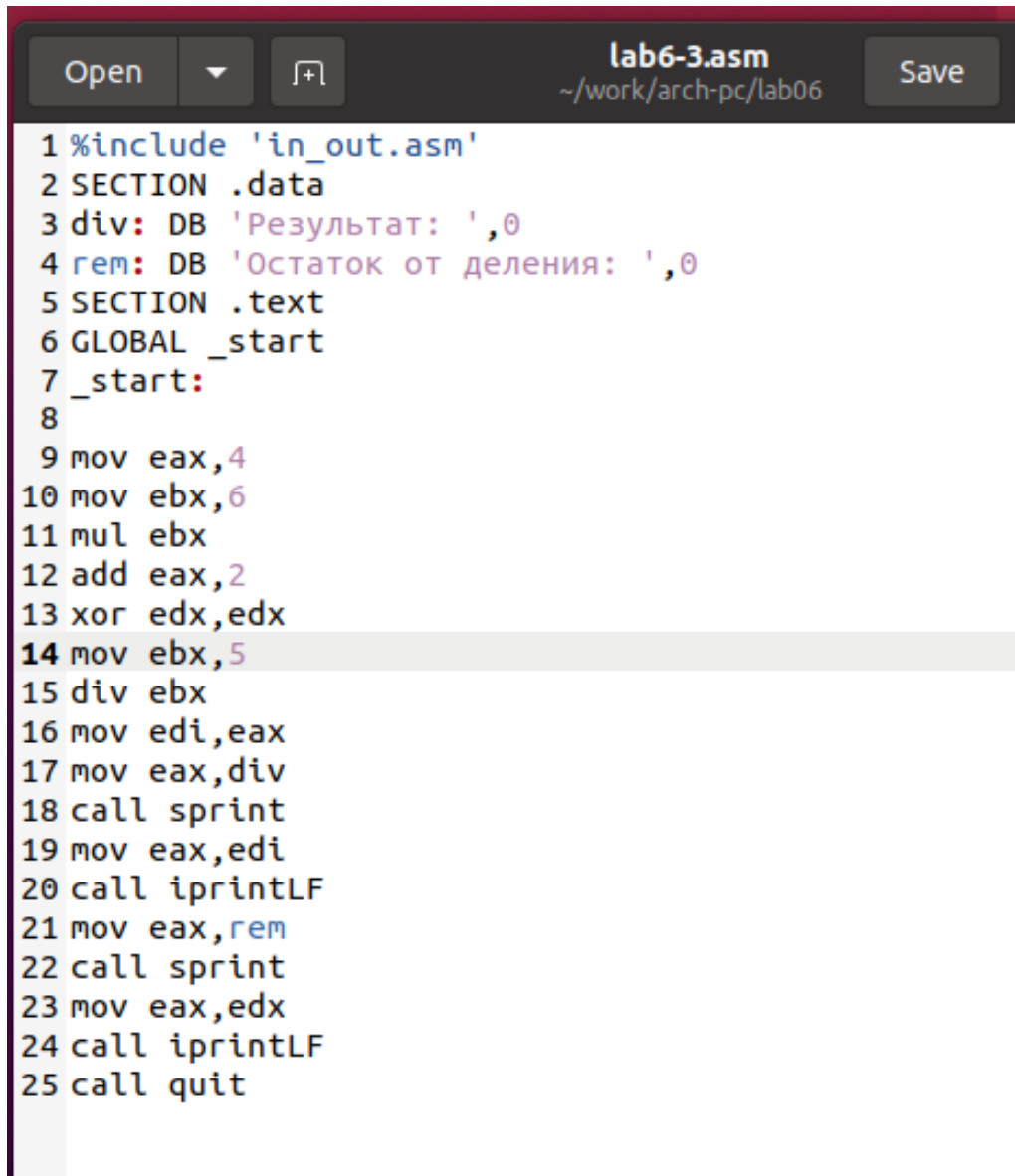
```
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-3.o -o lab6-3
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
dovraniviev@VirtualBox:~/work/arch-pc/lab06$
```

Рис. 2.12: Запуск программы lab6-3.asm

Адаптировал код программы для расчета формулы

$$f(x) = (4 * 6 + 2) / 5$$

. Скомпилировал исполняемый файл и осуществил его тестирование.



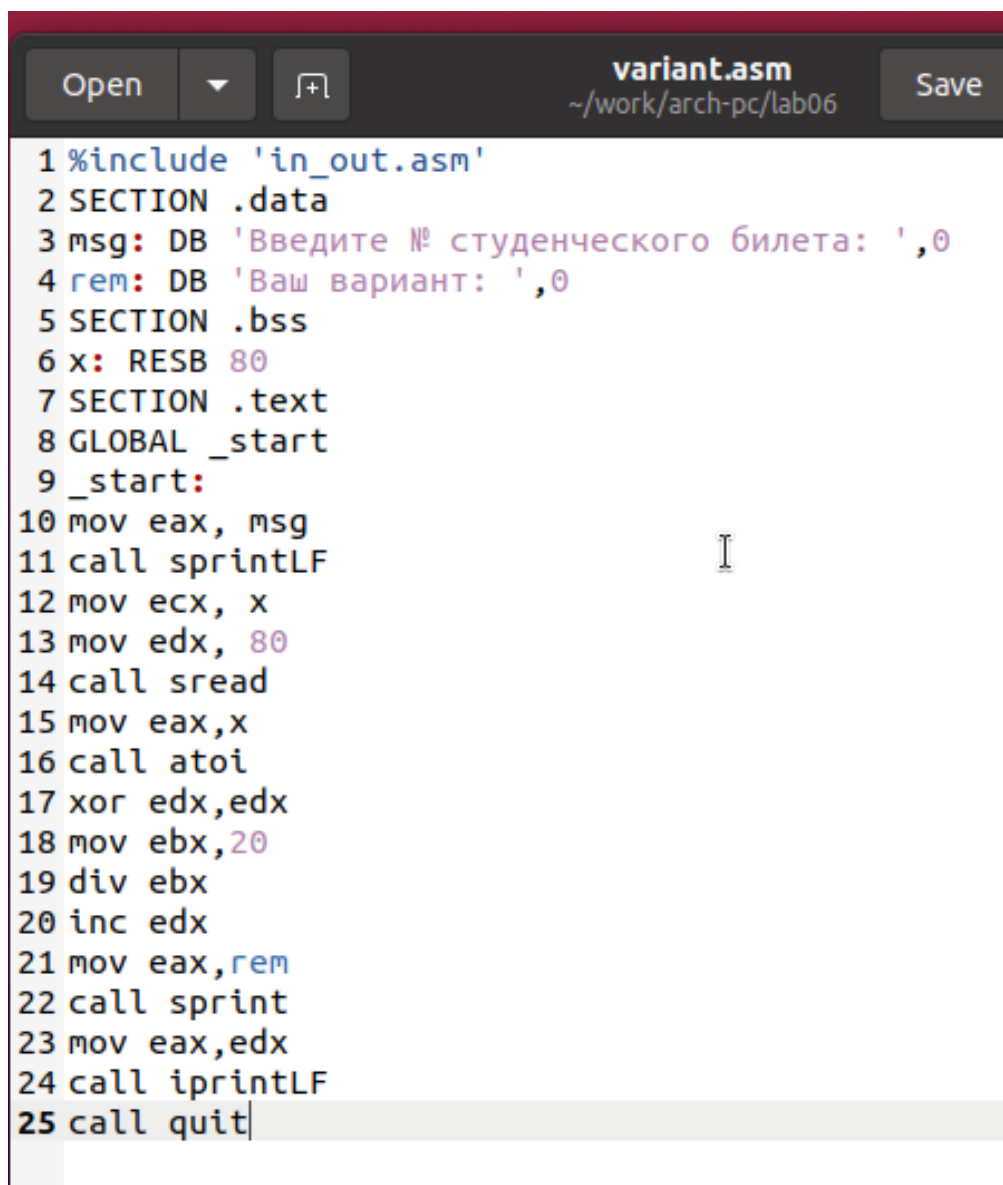
```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8
9 mov eax,4
10 mov ebx,6
11 mul ebx
12 add eax,2
13 xor edx,edx
14 mov ebx,5
15 div ebx
16 mov edi,eax
17 mov eax,div
18 call sprint
19 mov eax,edi
20 call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

Рис. 2.13: Программа lab6-3.asm

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

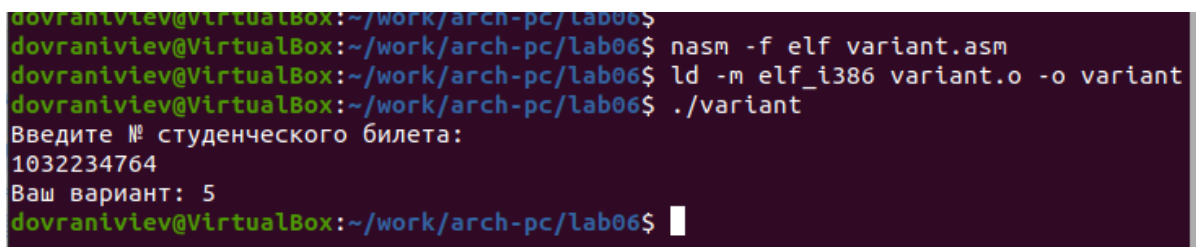
Рис. 2.14: Запуск программы lab6-3.asm

7. В качестве альтернативного примера рассмотрим программу, которая выполняет расчет заданного варианта исходя из номера студенческого билета.



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

Рис. 2.15: Программа variant.asm



```
dovraniviev@VirtualBox:~/work/arch-pc/lab06$
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf variant.asm
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 variant.o -o variant
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032234764
Ваш вариант: 5
dovraniviev@VirtualBox:~/work/arch-pc/lab06$
```

Рис. 2.16: Запуск программы variant.asm

ответы на вопросы

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

Значение с текстом ‘Ваш вариант:’ загружается в регистр `eax` с помощью команды `mov eax,rem`.

Для вывода указанной строки на экран используется команда `call sprint`, которая инициирует соответствующую функцию.

2. Для чего используются следующие инструкции?

```
mov ecx, x  
mov edx, 80  
call sread
```

Значение переменной `X` копируется в регистр `ecx` с помощью команды `mov ecx, x`.

Команда `mov edx, 80` задаёт число 80 в регистре `edx`.

Вызов функции чтения с консоли осуществляется командой `call sread`.

3. Для чего используется инструкция “`call atoi`”?

Вызов “`call atoi`” инициирует функцию, конвертирующую строковые данные в целочисленный формат.

4. Какие строки листинга отвечают за вычисления варианта?

Обнуление регистра `edx` достигается командой `xor edx,edx`.

Установка числа 20 в регистр `ebx` осуществляется командой `mov ebx,20`.

Деление числа студенческого билета на 20 производится командой `div ebx`.

Последующее увеличение содержимого регистра `edx` на единицу выполняется командой `inc edx`.

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

Регистр `edx` используется для хранения остатка от деления при выполнении команды `div ebx`.

6. Для чего используется инструкция “`inc edx`”?

Команда `inc edx` служит для инкрементирования содержимого регистра `edx` на один. Это необходимо для корректировки результата вычисления варианта, учитывая прибавление единицы к остатку от деления.

7. Какие строки листинга отвечают за вывод на экран результата вычислений?

Для передачи результата расчётов в регистр `eax` используется команда `mov eax,edx`.

Для вывода результата на экран применяется команда `call iprintLF`, активирующая соответствующую функцию.

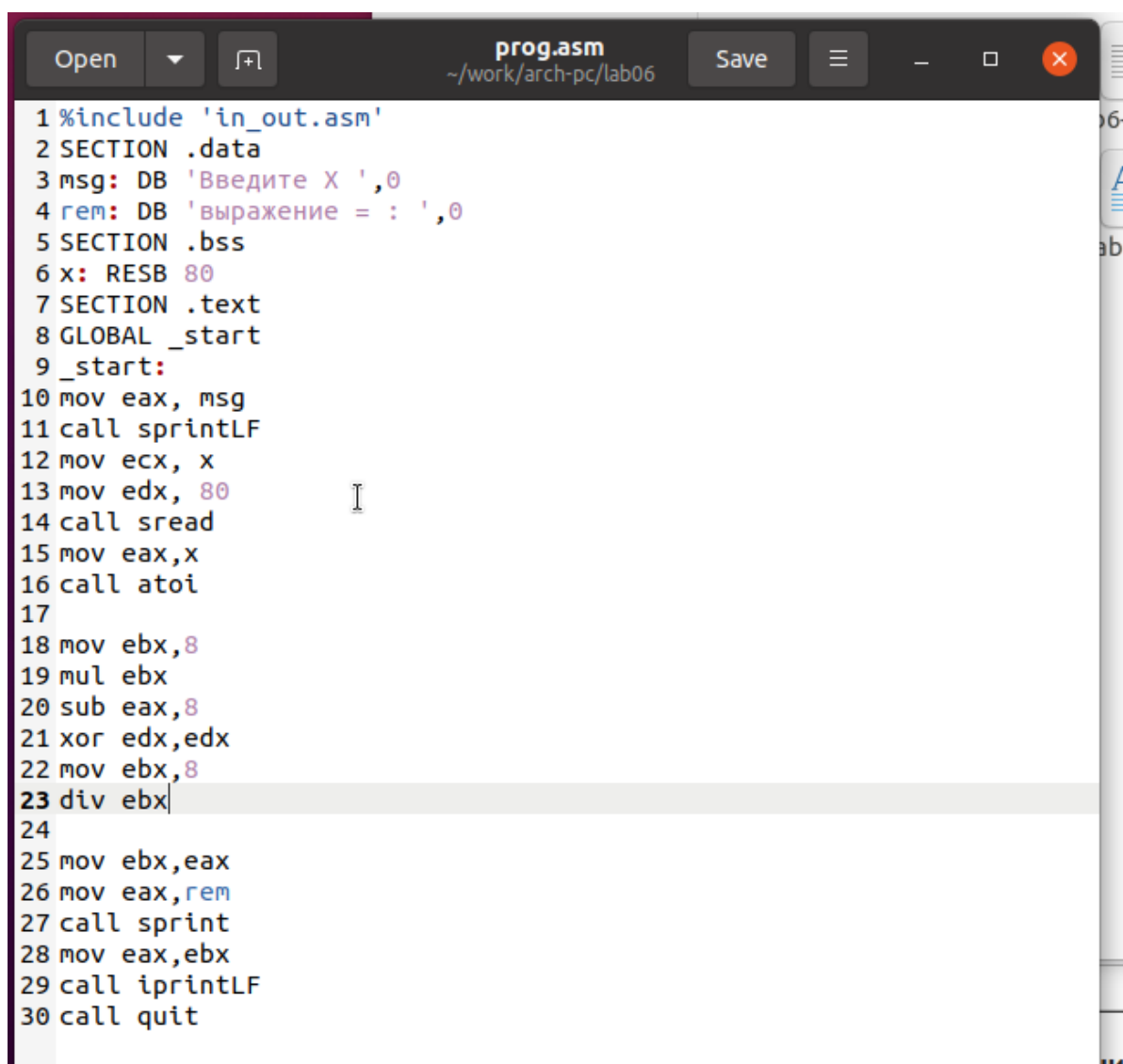
8. Написать программу вычисления выражения $y = f(x)$. Программа должна выводить выражение для вычисления, выводить запрос на ввод значения x , вычислять заданное выражение в зависимости от введенного x , выводить результат вычислений. Вид функции $f(x)$ выбрать из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создайте исполняемый файл и проверьте его работу для значений x_1 и x_2 из 6.3.

Получили вариант 5 -

$$(9x - 8)/8$$

для

$$x = 8, x = 64$$



```
1 %include 'in_out.asm'
2 SECTION .data
3 msg: DB 'Введите X ',0
4 rem: DB 'выражение = : ',0
5 SECTION .bss
6 x: RESB 80
7 SECTION .text
8 GLOBAL _start
9 _start:
10 mov eax, msg
11 call sprintLF
12 mov ecx, x
13 mov edx, 80
14 call sread
15 mov eax, x
16 call atoi
17
18 mov ebx, 8
19 mul ebx
20 sub eax, 8
21 xor edx, edx
22 mov ebx, 8
23 div ebx
24
25 mov ebx, eax
26 mov eax, rem
27 call sprint
28 mov eax, ebx
29 call iprintLF
30 call quit
```

Рис. 2.17: Программа prog.asm

```
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf prog.asm
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 prog.o -o prog
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ./prog
Введите X
8
выражение = : 7
dovraniviev@VirtualBox:~/work/arch-pc/lab06$ ./prog
Введите X
64
выражение = : 63
dovraniviev@VirtualBox:~/work/arch-pc/lab06$
```

Рис. 2.18: Запуск программы prog.asm

3 Выводы

Изучили работу с арифметическими операциями.