Отчёт по лабораторной работе №6

Выполнил студент НКАбд-02-25

Арина Андреевна Дрекина

Содержание

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Порядок выполнения лабораторной работы

3 Символьные и численные данные в NASM

Сначала я создала каталог для программ лабораторной работы №6, а после перешла в него. (Рисунок 3.1 и Рисунок 3.2)

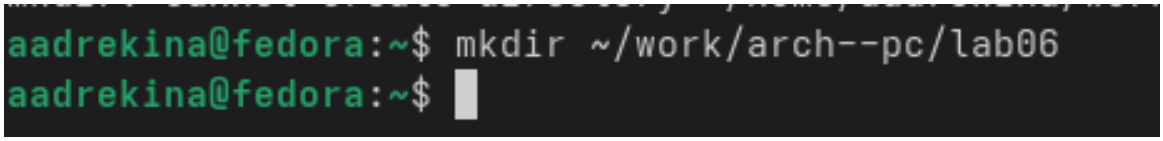

Рисунок 3.1: Создание каталога для лабораторной работы.

Рисунок 3.2: Переход в созданный каталог.

После этого я создала текстовый файл «lab6-1.asm» (Рисунок 3.3)

Рисунок 3.3: Создание текстового файла.

Затем я ввела текст из Листинга 6.1 (Рисунок 3.4)

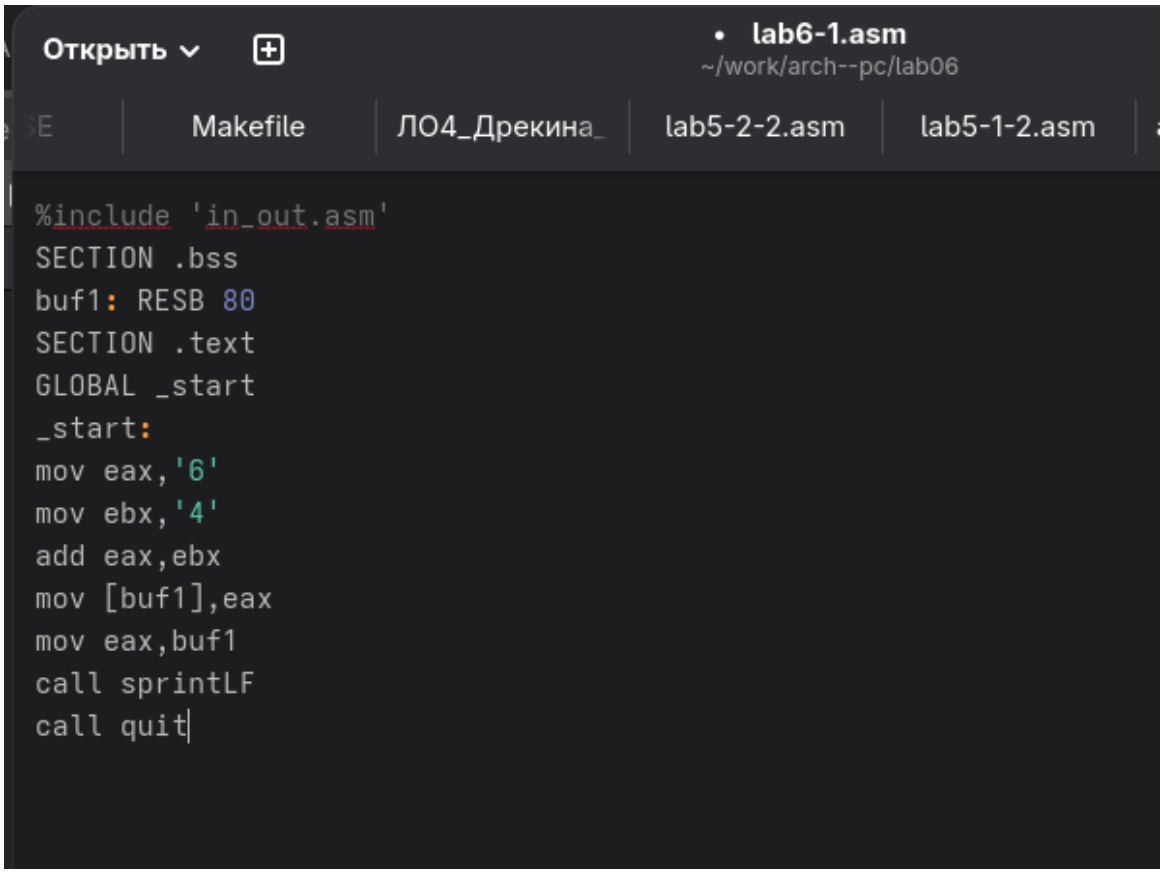

Рисунок 3.4: Ввод текста из Листинга 6.1.

Листинг 6.1:

```
%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
```

Далее я сделала полученный файл исполняемым, а перед тем как запустить его, в папку lab06 я поместила копию файла «in_out.asm». (Рисунок 3.5)

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

Рисунок 3.5: Создание исполняемого файла и его запуск.

Можно заметить, что программа вывела не число 10, а букву j, все потому что код символа 6 равен 00110110 в двоичном представлении, а код символа 4 - 00110100. Команда «add eax,abx» запишет в регистр eax сумму кодов - 01101010, что является кодом символа j.

Далее я изменила текст программы: вместо символов записала регистры чисел.(Рисунок 3.6)

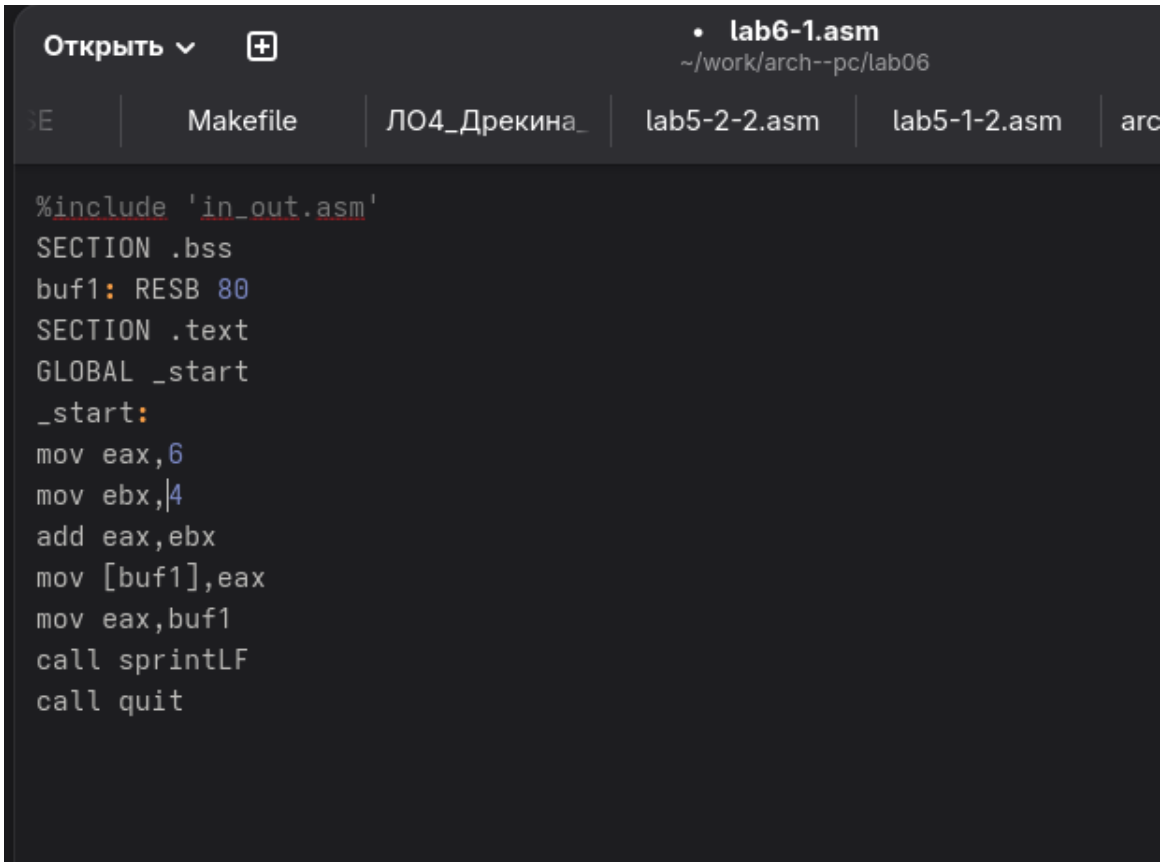

Рисунок 3.6: Измененный текст программы.

Затем я сделала этот файл исполняемым и запустила его. (Рисунок 3.7)

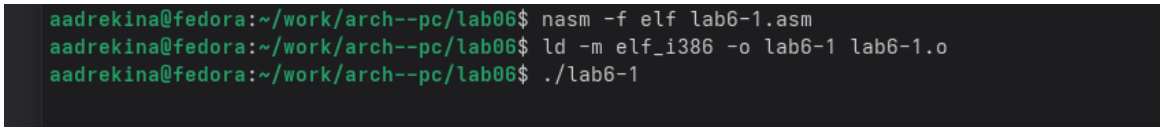

Рисунок 3.7: Создание исполняемого файла и его запуск.

Можно заметить, что число 10 мы так и не получили, однако теперь нам не вывелась и буква j, символ при выводе на экран не отображается.

После этого я создала еще один файл «lab6-2.asm». (Рисунок 3.8)

Рисунок 3.8: Создание нового текстового файла.

Затем в созданный файл я написала текст из Листинга 6.2. (Рисунок 3.9)

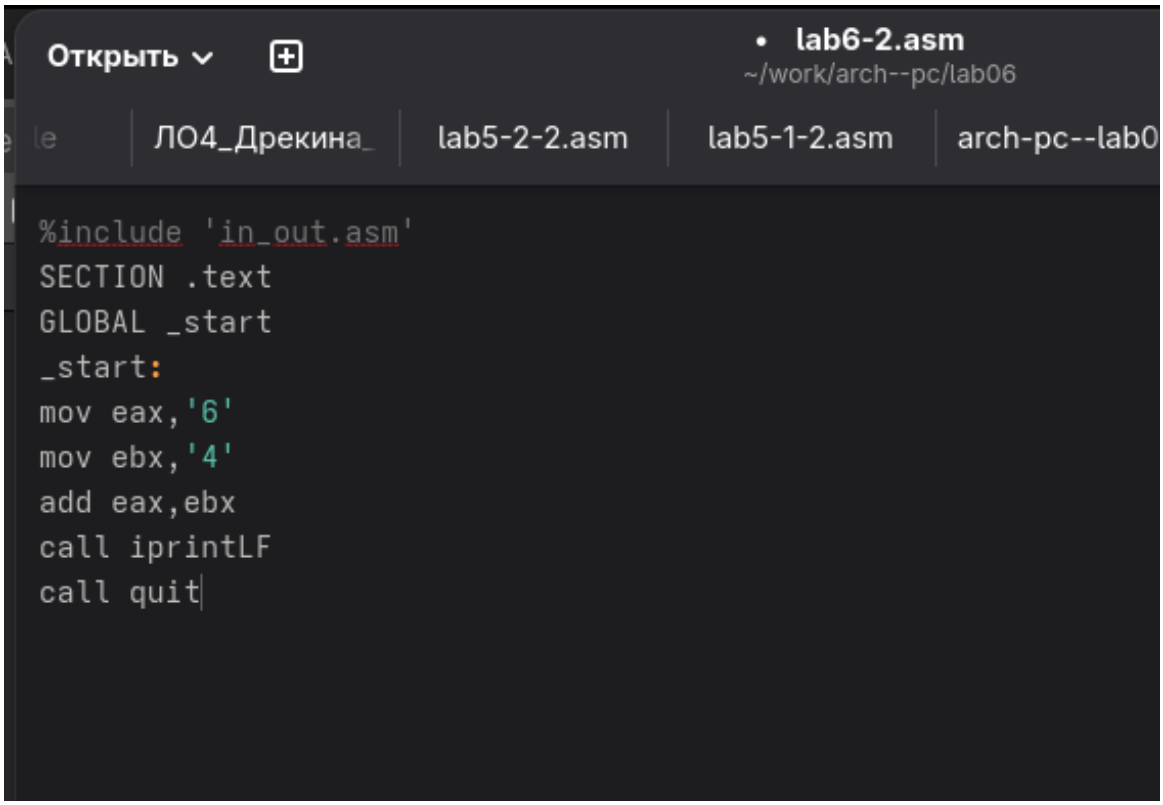

Рисунок 3.9: Добавление текста в созданный файл.

Листинг 6.2:

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
```

Далее я создала исполняемый файл и запустила его. (Рисунок 3.10)

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

Рисунок 3.10: Создание исполняемого файла и его запуск.

Теперь вывелось число 106. В этом случае команда add складывает коды символов „6“ и „4“. Однако, в отличие от первой проограммы, функция iprintLF позволяет вывести число, а не символ, кодом которого является число.

В этой программе я тоже изменила символы на числа. (Рисунок 3.11)

Открыть

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

ЛО4_Дрекина_ | lab5-2-2.asm | lab5-1-2.asm | arch-pc--lab

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Рисунок 3.11: Изменения в тексте программы.

Далее я сделала файл исполняемым и запустила его. (Рисунок 3.12)

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

Рисунок 3.12: Создание исполняемого файла и его запуск.

После изменений программы вывелось число 10, к которому мы шли с самого начала. Далее я поменяла iprintLF на iprint. (Рисунок 3.13)

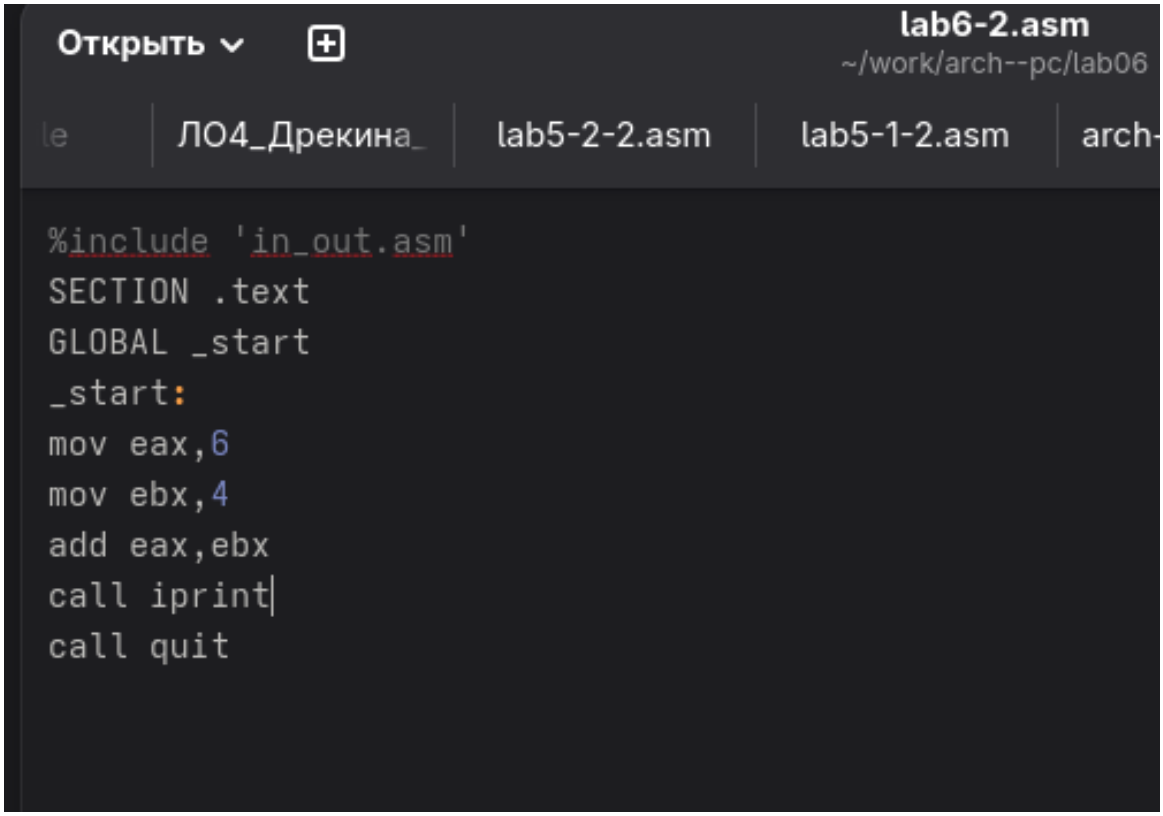

Рисунок 3.13: Замена iprintLF на iprint.

Затем сделала исполняемый файл и запустила программу. (Рисунок 3.14)

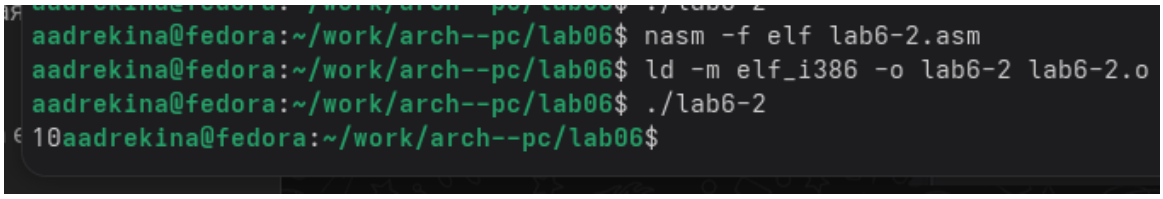

Рисунок 3.14: Создание исполняемого файла и запуск.

Отличие заключается в том, что в iprintLF курсор переходит на новую строку, а вот в iprint курсор остается на этой же строке.

4 Выполнение арифметических операций в NASM.

Я создала файл lab3-asm, для программы, которая будет выполнять арифметические операции в NASM. (Рисунок 4.1)

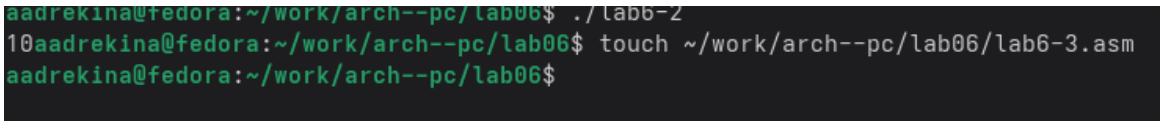

Рисунок 4.1: Создание файла для дальнейшей работы.

Далее я изучила текст Листинга 6.3 и ввела его в lab6-3.asm. (Рисунок 4.2)

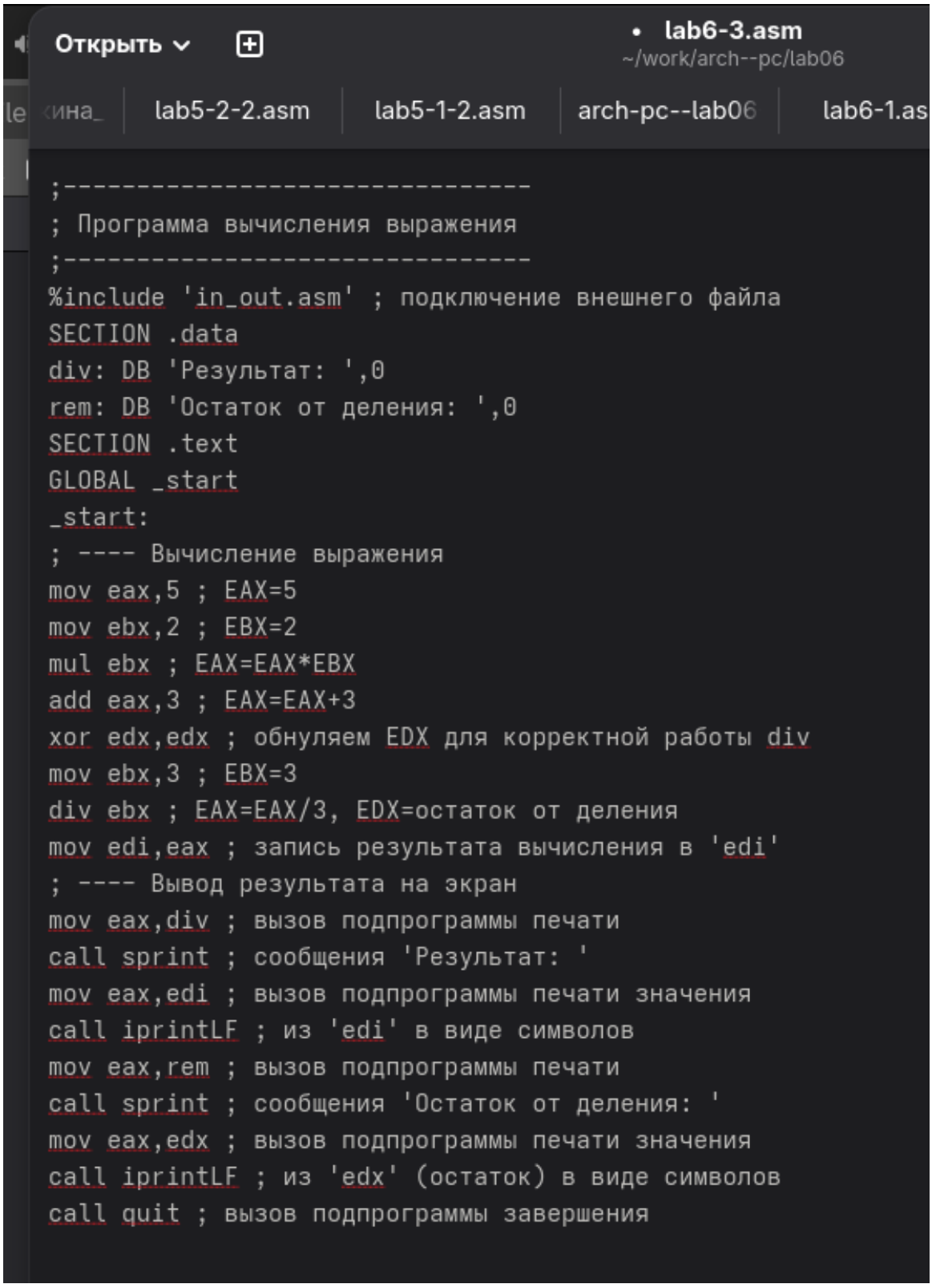

Рисунок 4.2: Ввод текста программы.

Листинг 6.3:

```
;-----------------------------
; Программа вычисления выражения
;-----------------------------
%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
```

Затем я сделала исполняемый файл и запустила его. (Рисунок 4.3)

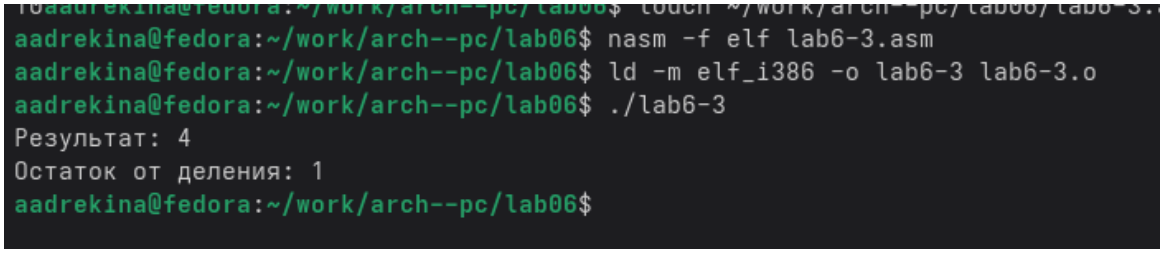

Рисунок 4.3: Создание исполняемого файла и запуск.

После запуска на экран вывелся - «Результат: 4», а на следующей строке - «Остаток от деления: 1». Что совпадает с выводом из лекции, значит все сделано правильно.

Далее я изменила текст программы: теперь у меня вычисляется выражение f(x) = (4 * 6+2)/5, а не f(x) =(5 * 2+3)/3. (Рисунок 4.4)

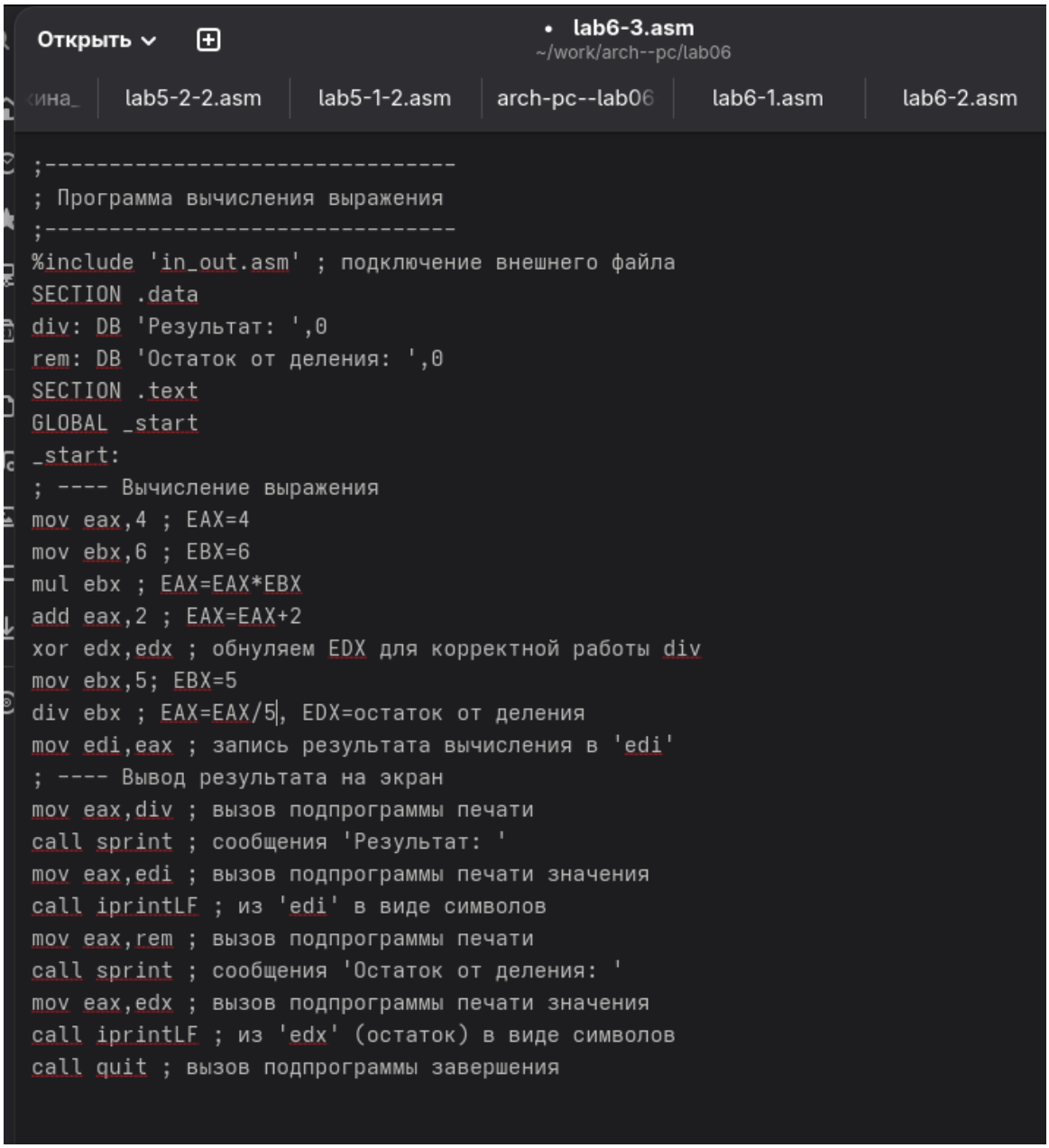

Рисунок 4.4: Изменения в тексте программы.

После того как я изменила текст я создала исполняемый файл и запустила его. (Рисунок 4.5)

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

Рисунок 4.5: Исполняемый файл и запуск программы.

В терминале вывелось - «Результат: 5», а на следующей строке - «Остаток от деления: 1». Я проверила на калькуляторе, ответ у меня получился такой же, значит я сделала все правильно.

Теперь сделаем еще один пример, программа будет вычислять вариант задания по номеру студенческого билета. В самом начале программа будет запрашивать на ввод номер студенческого билета, затем будет вычислять по формуле (Sn mod 20) + 1, где Sn – номер студенческого билета (В данном случае a mod b – это остаток от деления a на b), а после выводить результат.

Я создала еще один текстовый файл с названием «variant.asm». (Рисунок 4.6)

```
aadrekina@fedora:~$ touch ~/work/arch--pc/lab06/variant.asm
aadrekina@fedora:~$
```

Рисунок 4.6: Создание текстового файла variant.asm.

В созданный текстовый файл я добавила текст программы из Листинга 6.4, но перед этим внимательно его изучила. (Рисунок 4.7)

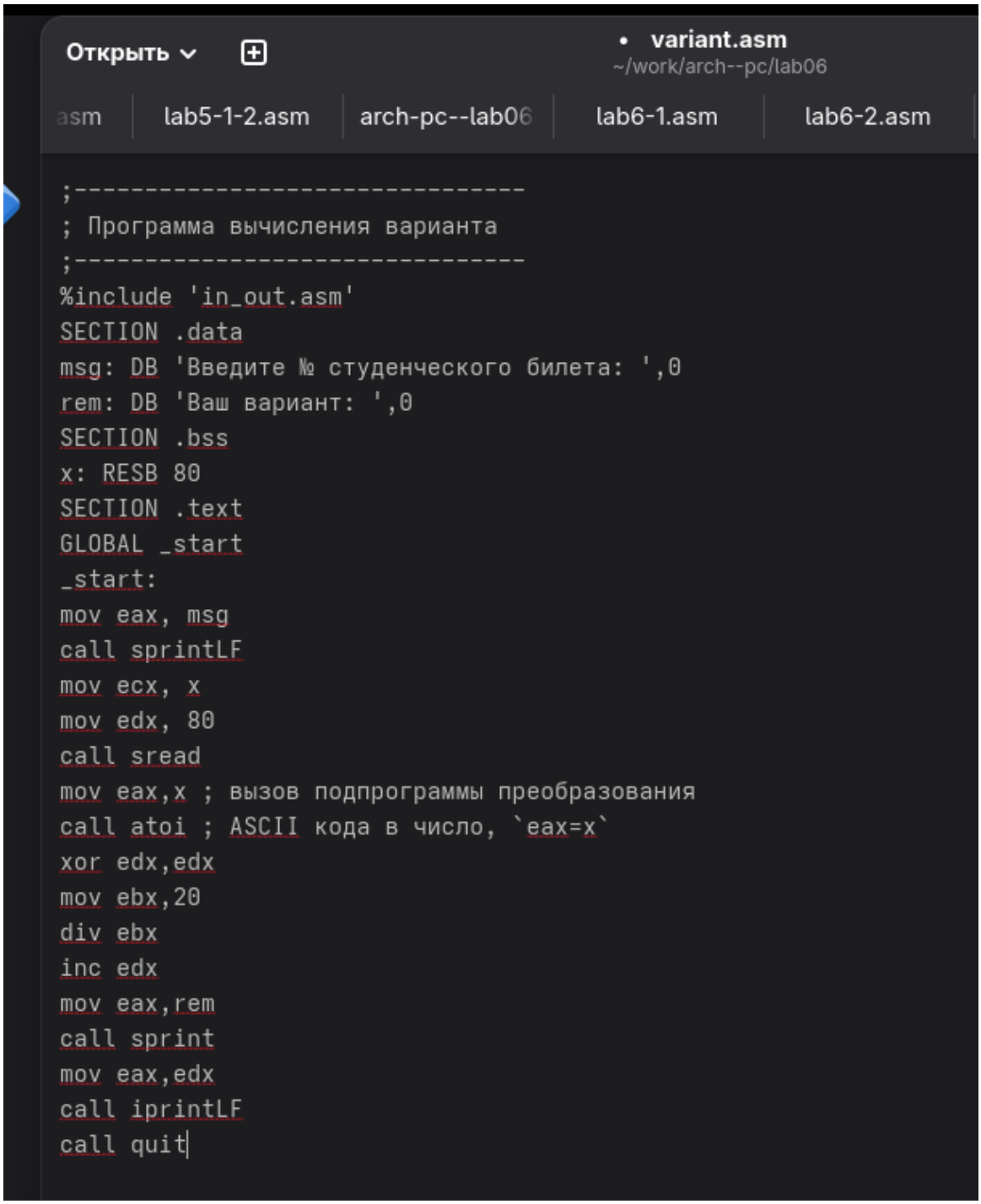

Рисунок 4.7: Перенос текста программы в файл variant.asm.

Листинг 6.4:

```
;-----------------------------
; Программа вычисления варианта
;-----------------------------
%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 sprintLF
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 sprint
mov eax,edx
call iprintLF
call quit
```

Затем я создала исполняемый файл и запустила его. (Рисунок 4.8)

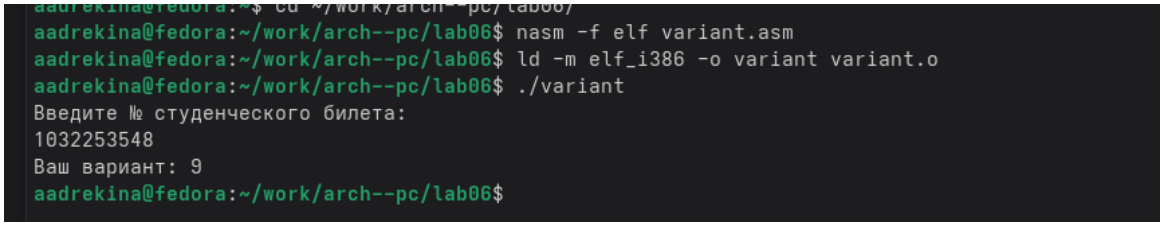

Рисунок 4.8: Создание исполняемого файла и его запуск.

В качестве вводных данных я ввела свой студенческий билет, мне вывелся результат «9». Я проверила вручную, нашла остаток деления своего студенческого билета на 20 - это 8, а после прибавила 1, получилось 9. Это означает, что программа корректна.

1.В Листинге 6.4 за вывод на экран сообщения «Ваш вариант:» отвечают строки:

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

2.Эти инструкции:

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

Используются для ввода студенческого билета, а затем введенные данные сохраняются в переменную x.

3.Строка «call atoi» используется для преобразований строки в число.

4. Строки

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

отвечают за вычисление варианта.

5. Остаток от деления при выполнении инструкции «dix ebx» записывается в регистр «EDX» .

6. Инструкция «inc edx» используется для увеличения значения в регистре EDX на 1.

7. Строки, отвечающие за вывод на экран результата вычислений:

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

5 Задание для самостоятельной работы.

Я создала еще один текстовый файл, для самостоятельной работы и назвала его «lab6-dz.asm». (Рисунок 5.1)

Рисунок 5.1: Создание нового файла.

Затем я создала программу, которая будет вычислять выражение вида y=10 + (31*x − 5). Программа должна выводить выражение для вычисления, выводить запрос на ввод значений x и после этого вычислять выражение и выводить ответ на экран. (Рисунок 5.2)

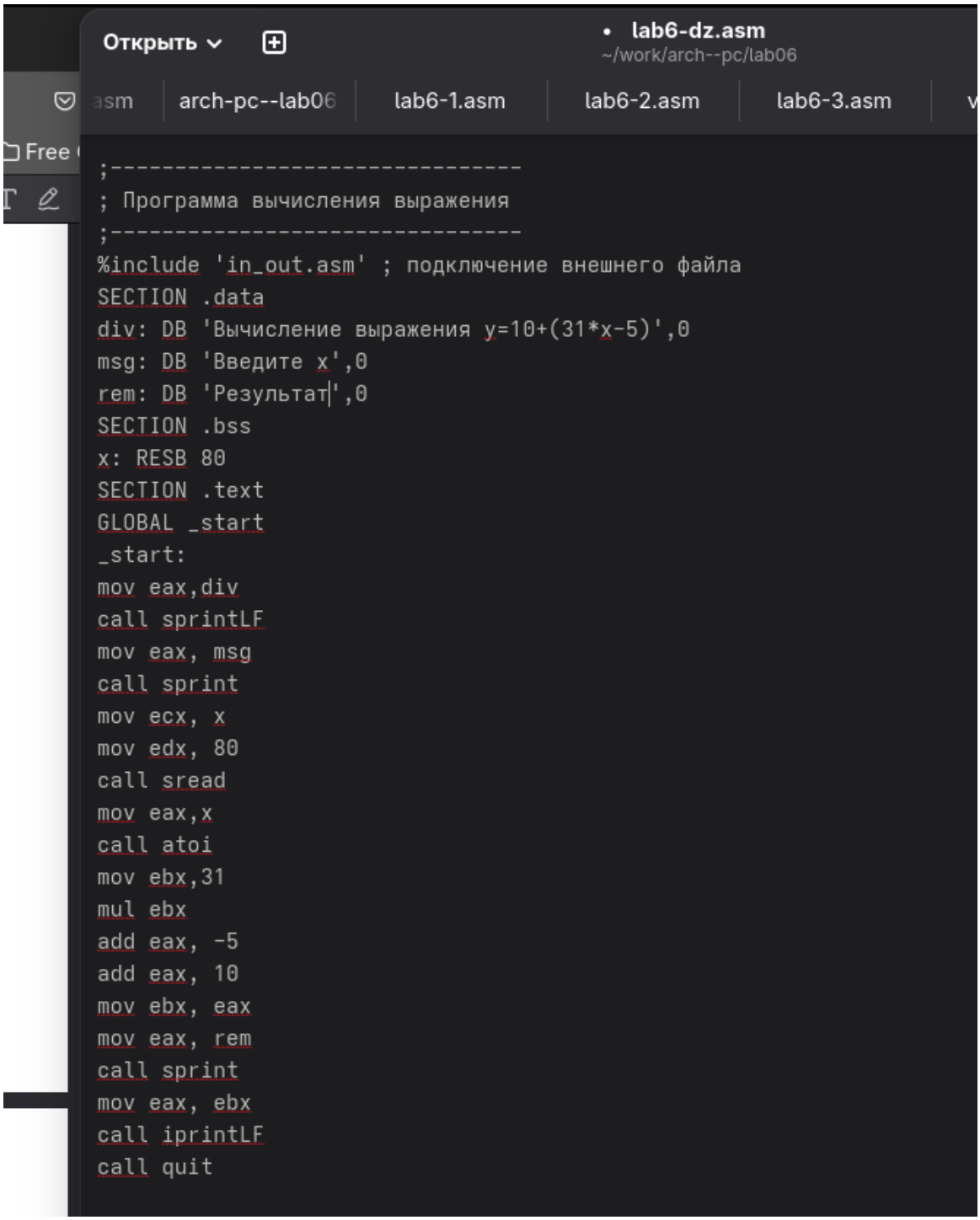

Рисунок 5.2: Написание текста программы.

Текст программы:

```
;---------------------------------
; Программа вычисления выражения
;---------------------------------
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Вычисление выражения y=10+(31*x-5)',0
msg: DB 'Введите x',0
rem: DB 'Результат',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,div
call sprintLF
mov eax, msg
call sprint
mov ecx, x
mov edx, 80
call sread
mov eax,x
call atoi
mov ebx,31
mul ebx
add eax, -5
add eax, 10
mov ebx, eax
mov eax, rem
```

```
call sprint
mov eax, ebx
call iprintLF
call quit
```

Далее я сделала этот файл исполняемым и запустила его, вводные данные я написала те, которые были в лекции. (Рисунок 5.3)

```
aadrekina@fedora:~/work/arch--pc/lab06$ nasm -f elf lab6-dz.asm
aadrekina@fedora:~/work/arch--pc/lab06$ ld -m elf_i386 -o lab6-dz lab6-dz.o
aadrekina@fedora:~/work/arch--pc/lab06$ ./lab6-dz
Вычисление выражения y=10+(31*x-5)
Введите x3 1
Результат98
aadrekina@fedora:~/work/arch--pc/lab06$
```

Рисунок 5.3: Создание исполняемого файла и запуск.

На экран вывелось число 98, я перепроверила все вручную, путем подстановки заданных x. Ответы совпали, это означает, что программа написана верно.

6 Вывод.

В ходе лабораторной работы я освоила арифметические инструкции языка ассемблера NASM и применила полученные знания на практике.