Лабораторная работа №6

Арифметические операции в NASM

Борисенкова София Павловна

Содержание

Список иллюстраций

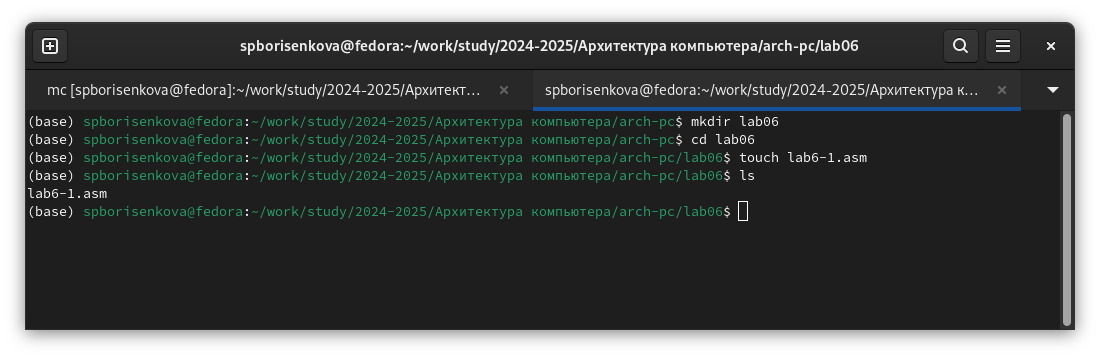
Список таблиц

# 1 Цель работы

Познакомиться с базовыми инструкциями языка Ассемблер, отвечающими за основные арифметические операции

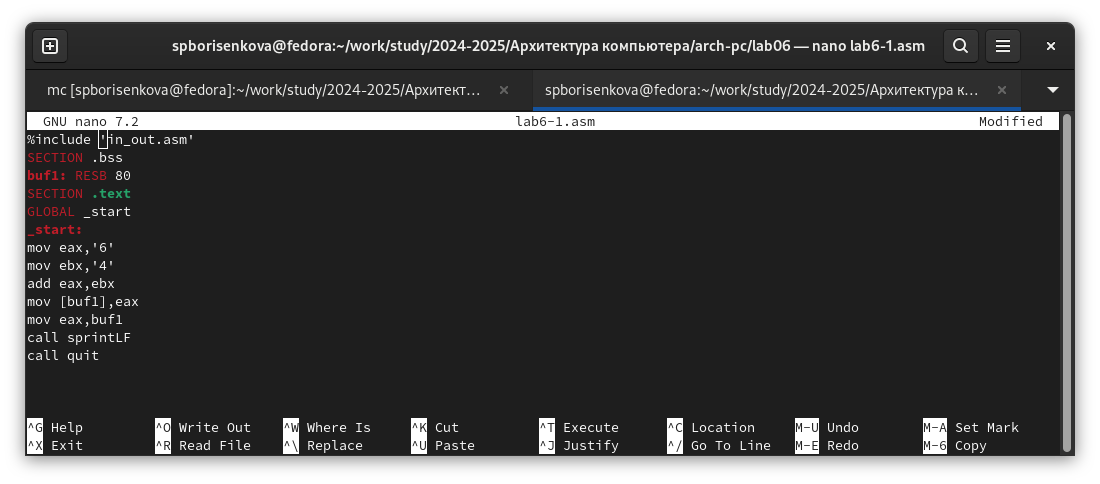
# 2 Выполнение лабораторной работы

Для начала выполнения лабораторной работы необходимо создать папку рабочего каталога и файл lab6-1.asm (Рис. 2.1):



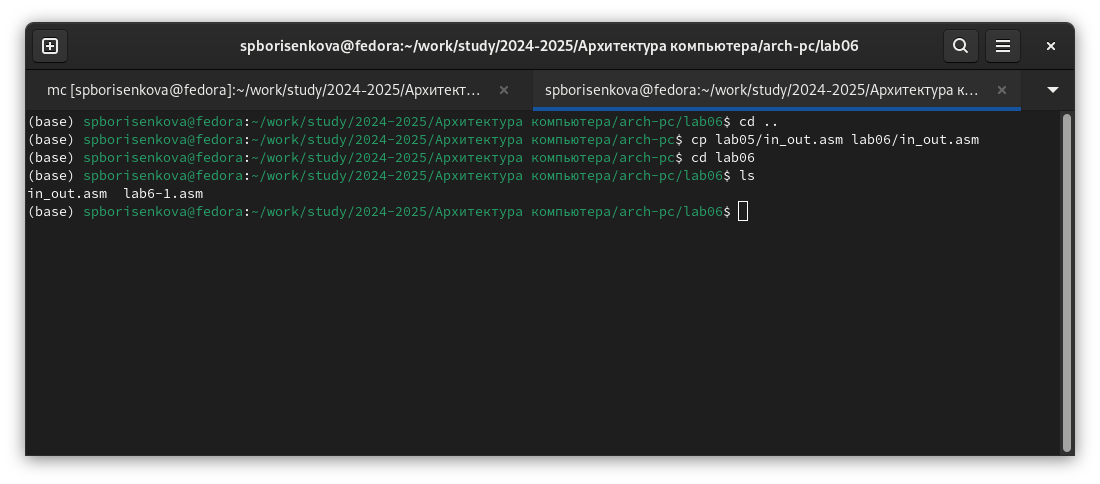
Создание рабочей директории и файла lab6-1.asm

Вставим в наш созданный файл код из листинга 6.1 (Рис. 2.2):



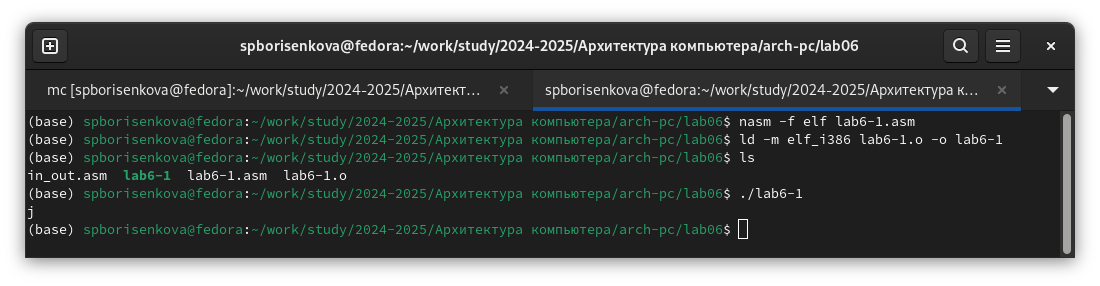
Запись кода из листинга в файл lab6-1.asm

Перед сборкой файла стоит учесть, что он использует сторонний файл in\_out.asm. Cкопируем его из каталога пятой лабораторной работы (Рис. 2.3):



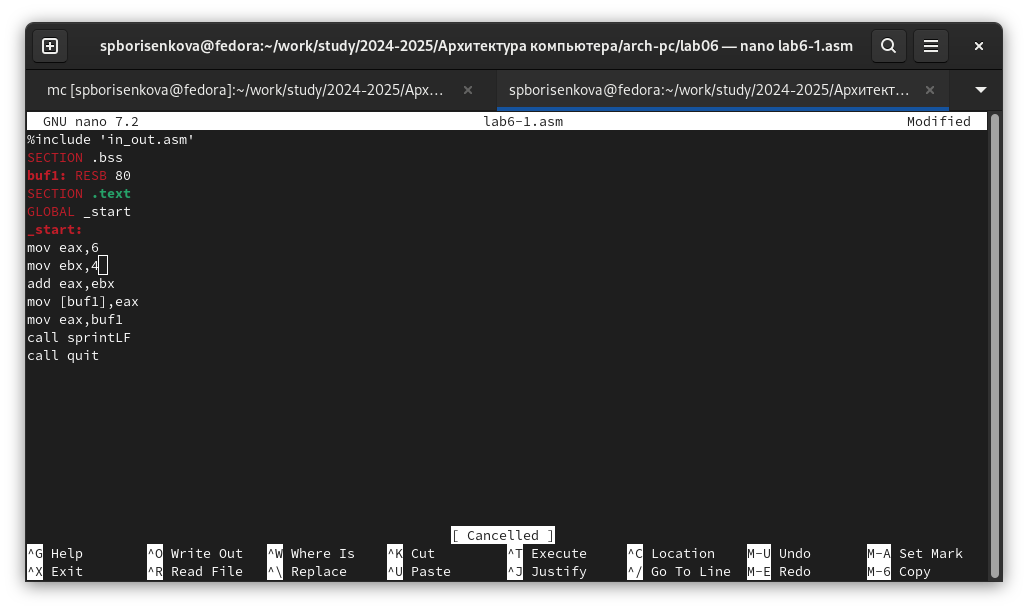
Копирование файла in\_out.asm в рабочую директорию

Теперь соберём наш файл в исполняемое приложение. Запустим его и посмотрим на результат (Рис. 2.4):



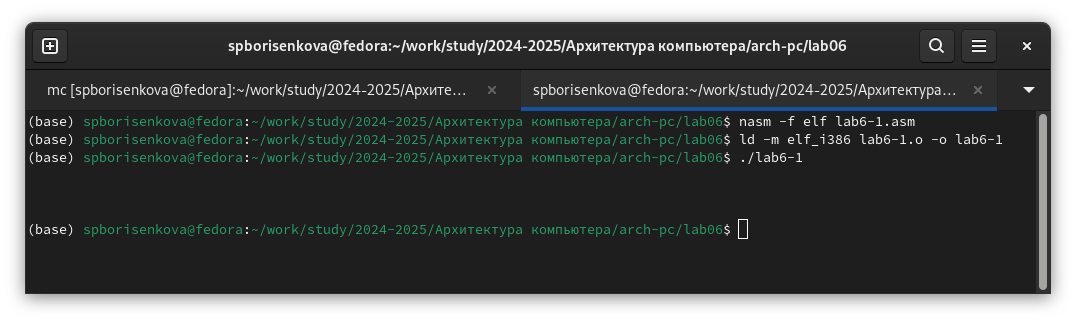
Запуск исполняемого файла и результат вывода

Программа выводит символ j, однако это неправильный вывод. Наша цель - сложить 6 и 4, и получить в выводе число 10. Изменим файл (Рис. 2.5):



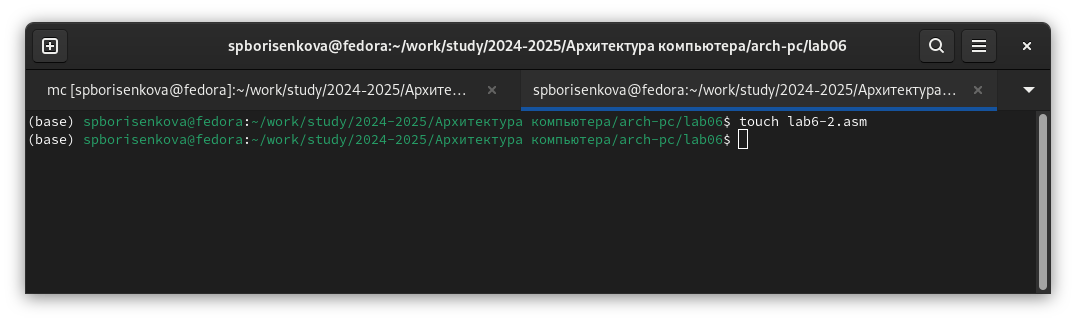
Редактирование файла

Мы убрали кавычки у цифр, и теперь складываем уже не символы 6 и 4, а числа. Теперь попробуем собрать исполняемый исполняемый файл также и запустим его (Рис. 2.6):



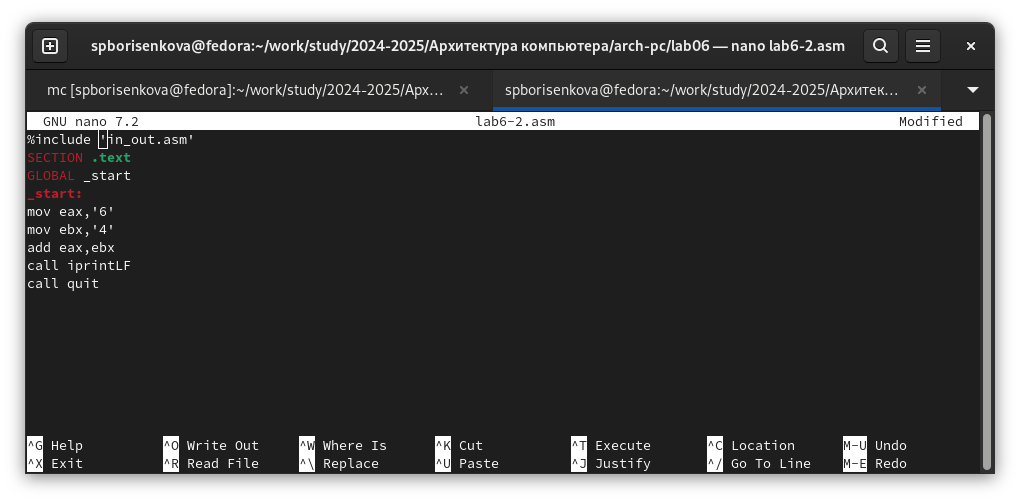
Запуск исполняемого файла и результат вывода

Мы видим, что ничего не вывелось. Но так ли это? Когда мы вызываем команду sprintLF, она выводит не число 10, а символ с номером 10. Посмотрим на таблицу ASCII и увидим, что символ под номером 10 это символ перевода строки. Именно поэтому мы его не видим, мы видим просто новую строку. Теперь создадим второй файл под названием lab6-2.asm (Рис. 2.7):



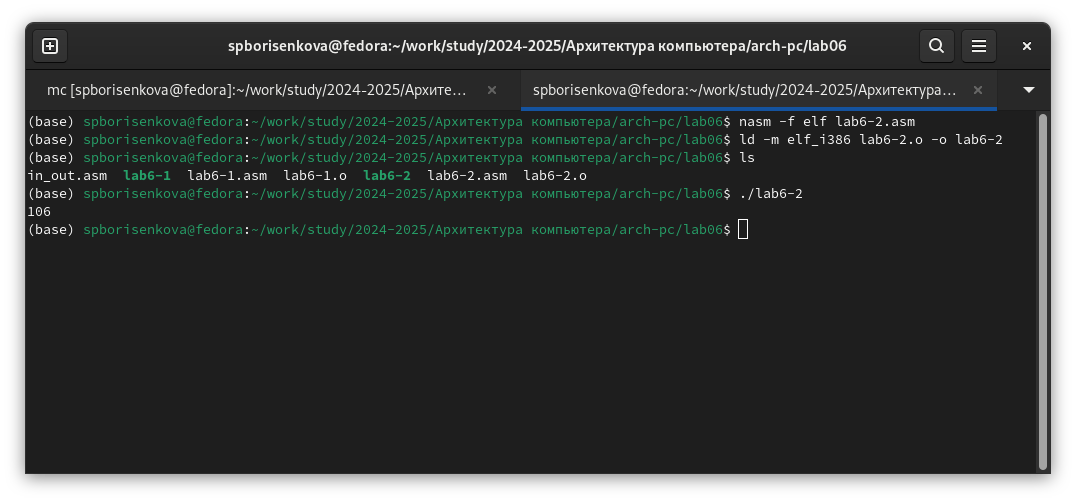
Создание файла lab6-2.asm

Вставим в него код из листинга 6.2 (Рис. 2.8):



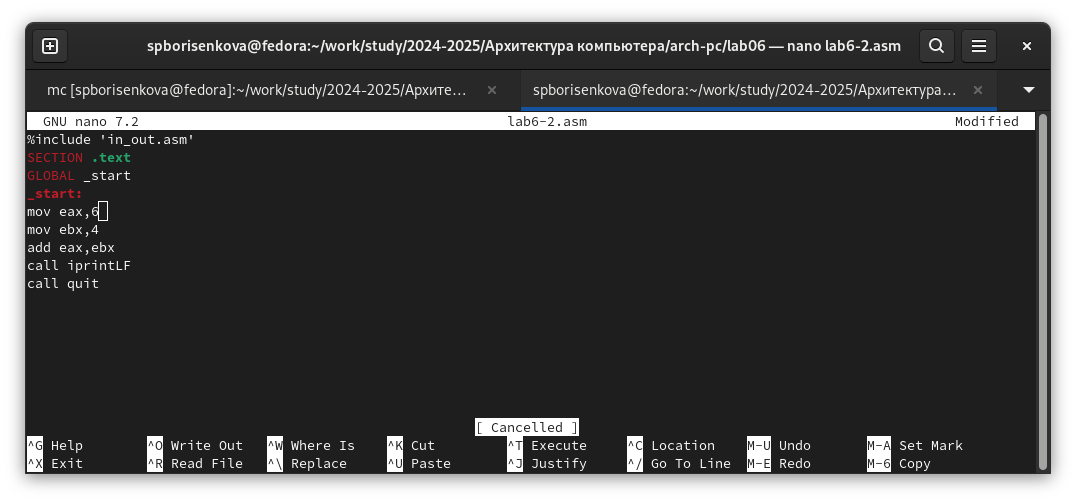
Запись кода в файл lab6-2.asm

Основное отличие заключается в том, что вместо sprintLF используется iprintLF. Соберём файл и запустим его, чтобы посмотреть, как изменится вывод (Рис. 2.9):



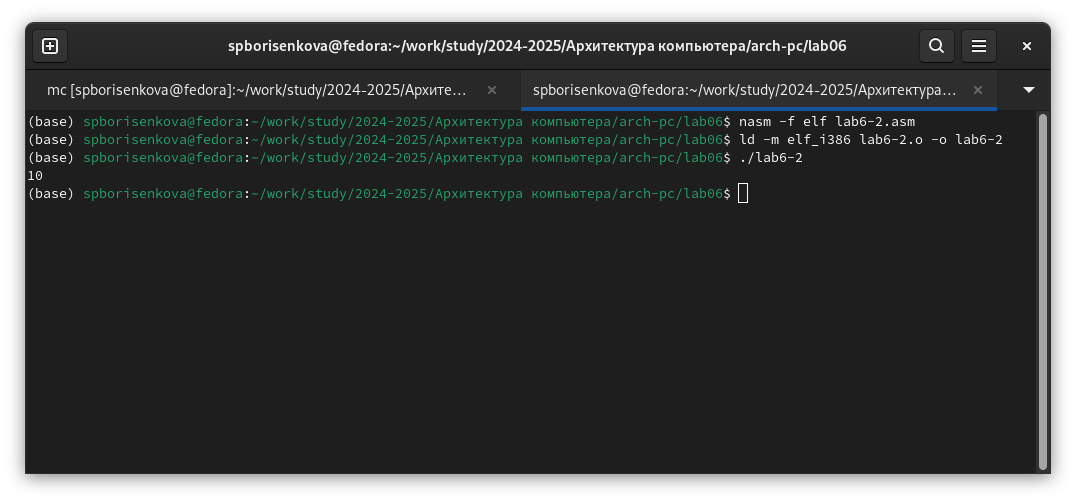
Запуск исполняемого файла и результат вывода

Мы видим число 106. Так как цифры в коде указаны в кавычках, мы складываем их коды (54 и 52 в сумме дают 106). Теперь программа способна вывести число, а не символ ASCII с соответствующим номером. Теперь, если мы уберём кавычки у цифр, программа должна вывести 10. Убедимся в этом, сделав соответствующие изменения в коде (Рис. 2.10):



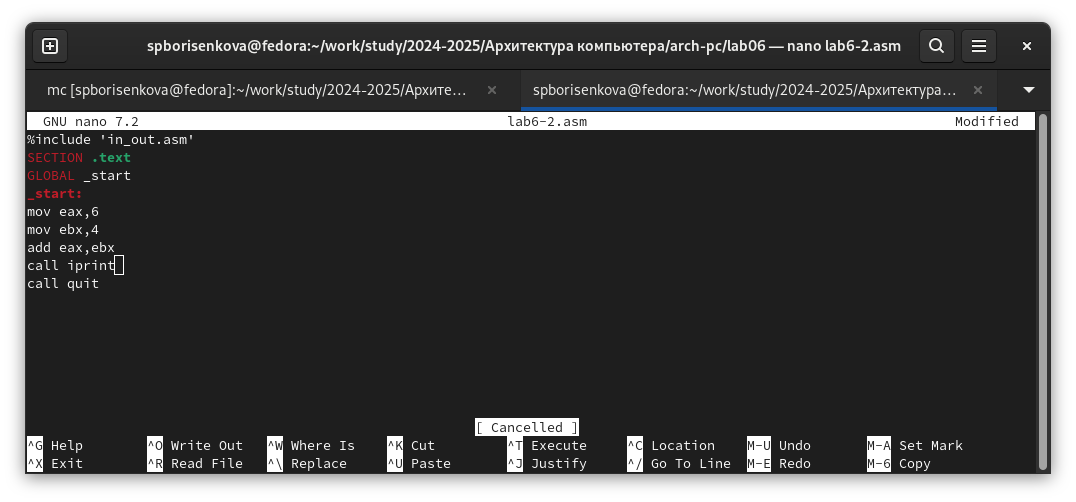
Изменение файла lab6-2.asm

Соберём программу и запустим её (Рис. 2.11):



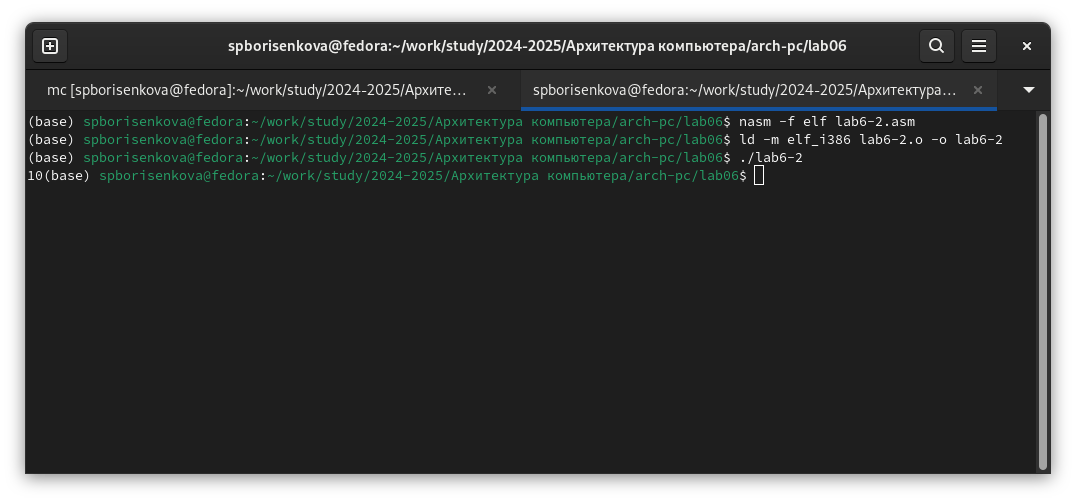
Сборка исполняемого файла и результат работы программы

Как видим, программа действительно вывела число 10. Кроме операции iprintLF в файле in\_out.asm есть операция iprint. Посмотрим, чем они отличаются. Заменим в коде iprintLF на iprint (Рис. 2.12):



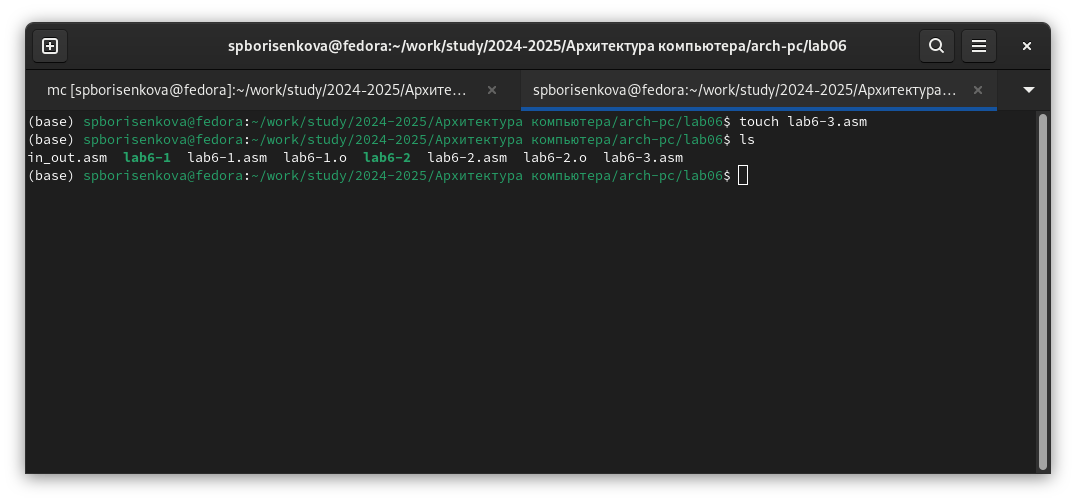
Редактирование файла lab6-2.asm

Попробуем собрать программу и запустить её (Рис. 2.13):



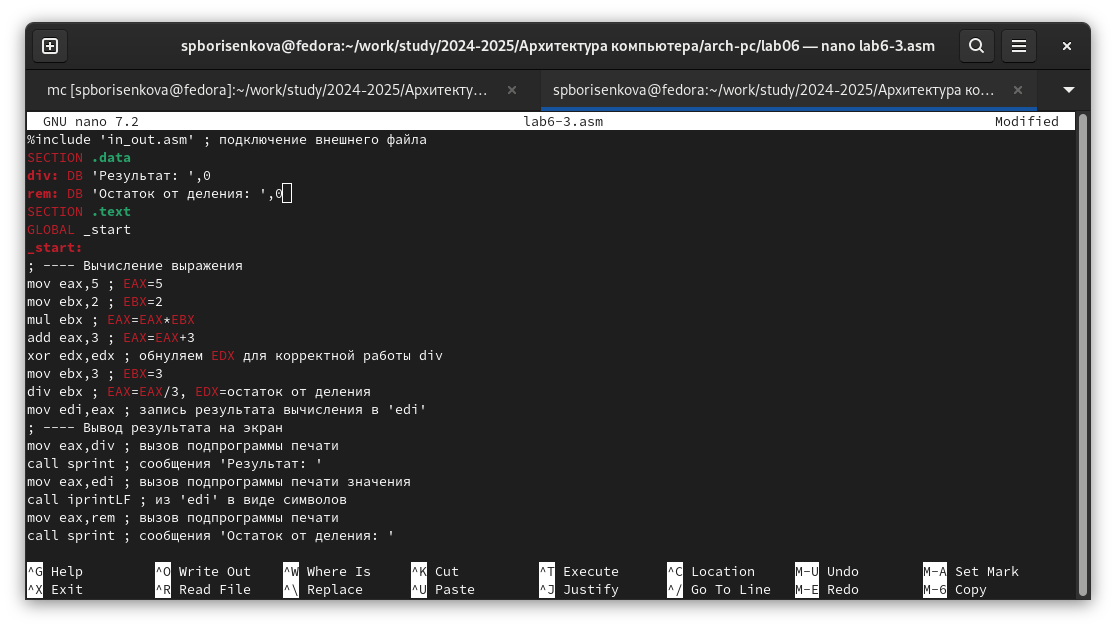
Сборка и результат работы отредактированного файла

Как видим, операция iprint не переносит на следующую строку, в отличие от iprintLF. Теперь создадим третий файл (Рис. 2.14):



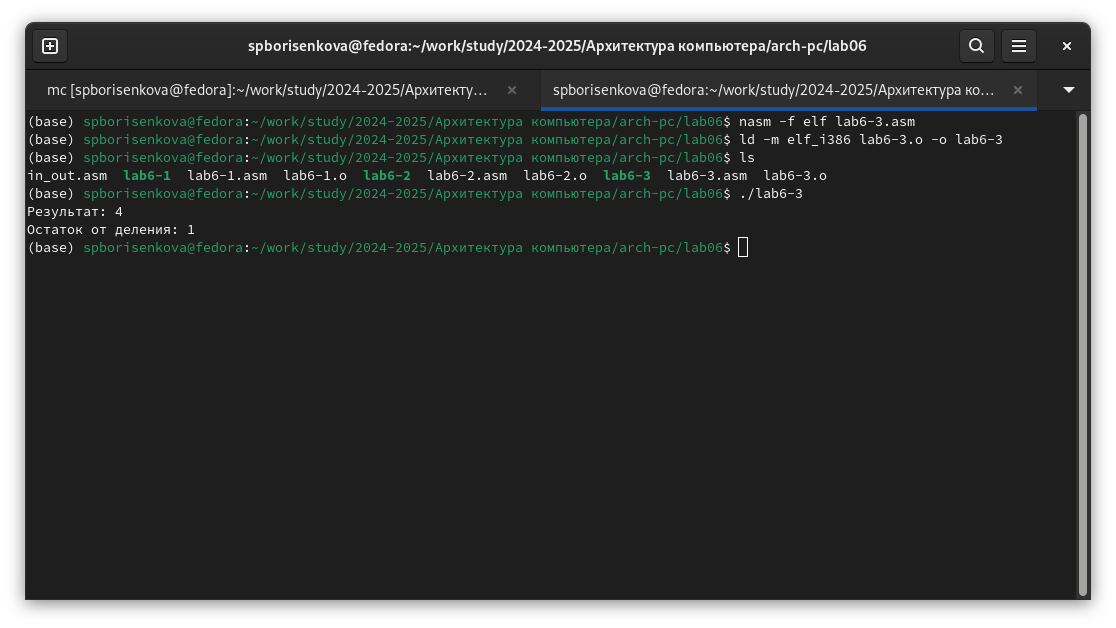
Создание файла lab6-3.asm

Он должен выводить значение функции (5\*2+3)/3. Для этого вставим код из файла листинга 6.3 (Рис. 2.15):



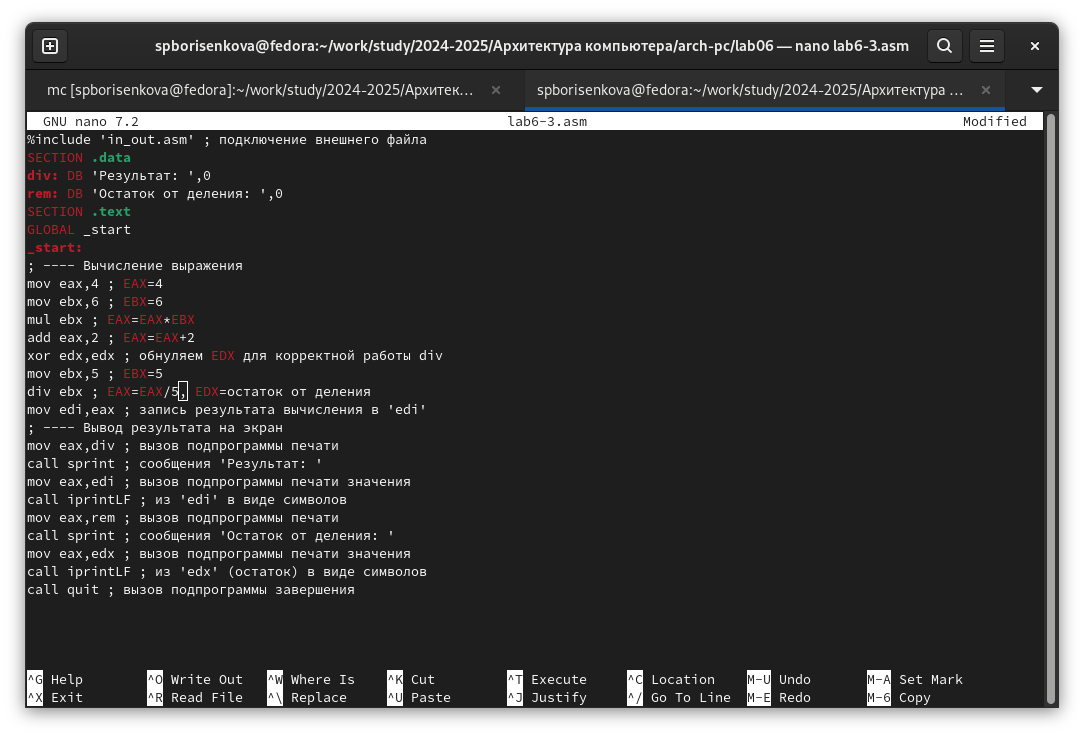
Вставка кода из листинга в файл

Попробуем запустить эту программу (Рис. 2.16):



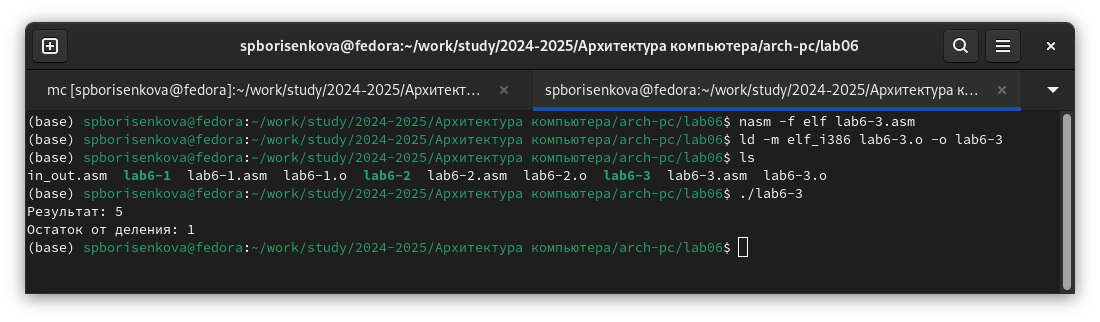
Сборка файла lab6-3.asm и результат его работы

Полученный результат совпадает с результатом, указанным в лабораторной работе. Теперь изменим файл так, чтобы он вычислял значение выражения (4\*6+2)/5. Для этого поменяем числа в коде следующим образом(Рис. 2.17):



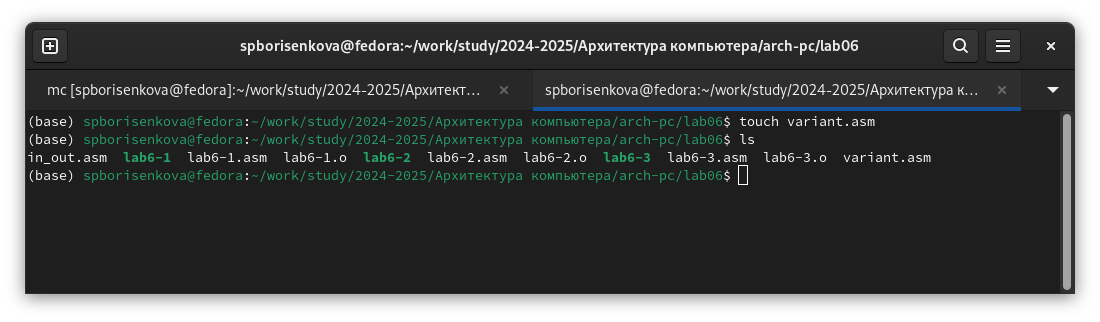
Редактирование файла lab6-3.asm

Соберём программу и запустим её (Рис. 2.18):



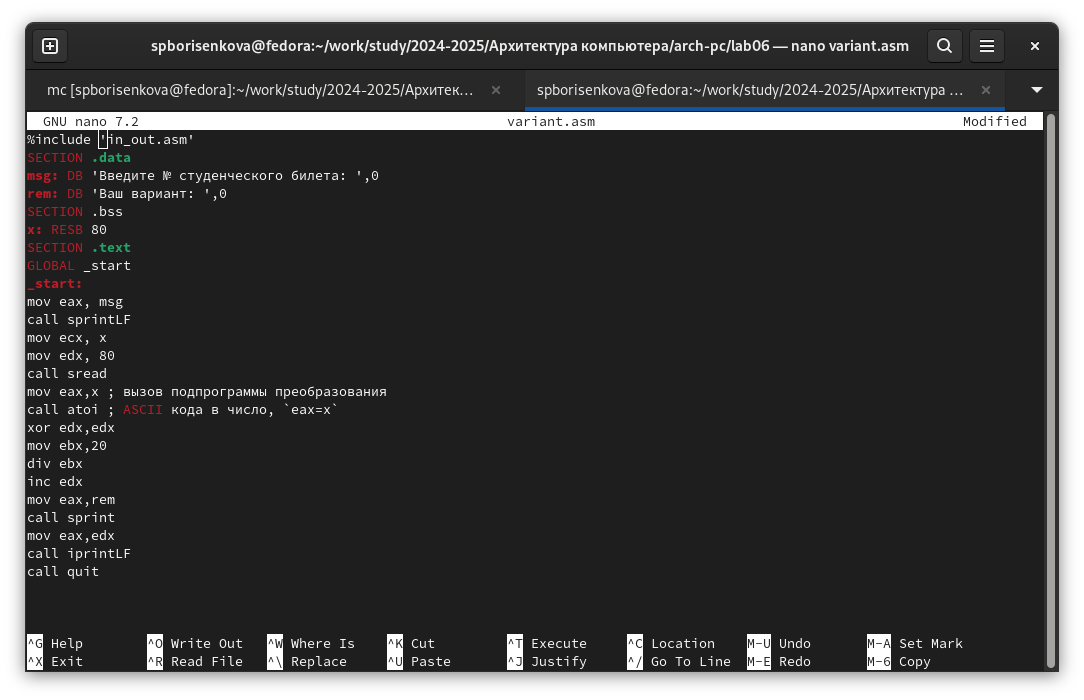
Сборка файла lab6-3.asm и результат его работы

Вывод корректный. Теперь создадим файл variant.asm для вычисления варианта самостоятельной работы (Рис. 2.19):



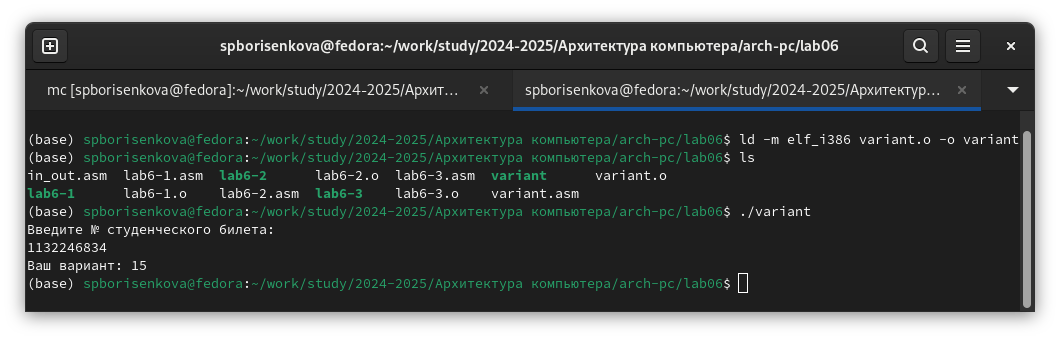
Создание файла variant.asm для вычисления варианта для самостоятельной работы

Вставим в файл код из листинга 6.4 (Рис. 2.20):



Вставка кода в файл variant.asm

Соберём и запустим программу, указав номер студенческого билета. В моём случае это 1132246834 (Рис. 2.21):



Сборка и запуск программы variant

Мой вариант - 15.

Ответы на вопросы:

1. За это отвечает 25-ая строчка call sprint, перед которой идёт строка mov eax,rem, которая перемещает строку с фразой в регистр eax, из которого мы считываем данные для вывода

2. Эти строки используются для того, чтобы записать данные в переменную x

3. Для преобразования ASCII кода в число

4. Напрямую за вычисление отвечают следующие строки:

div ebx

inc edx

5. В регистр edx

6. Для увеличения значения регистра edx на единицу

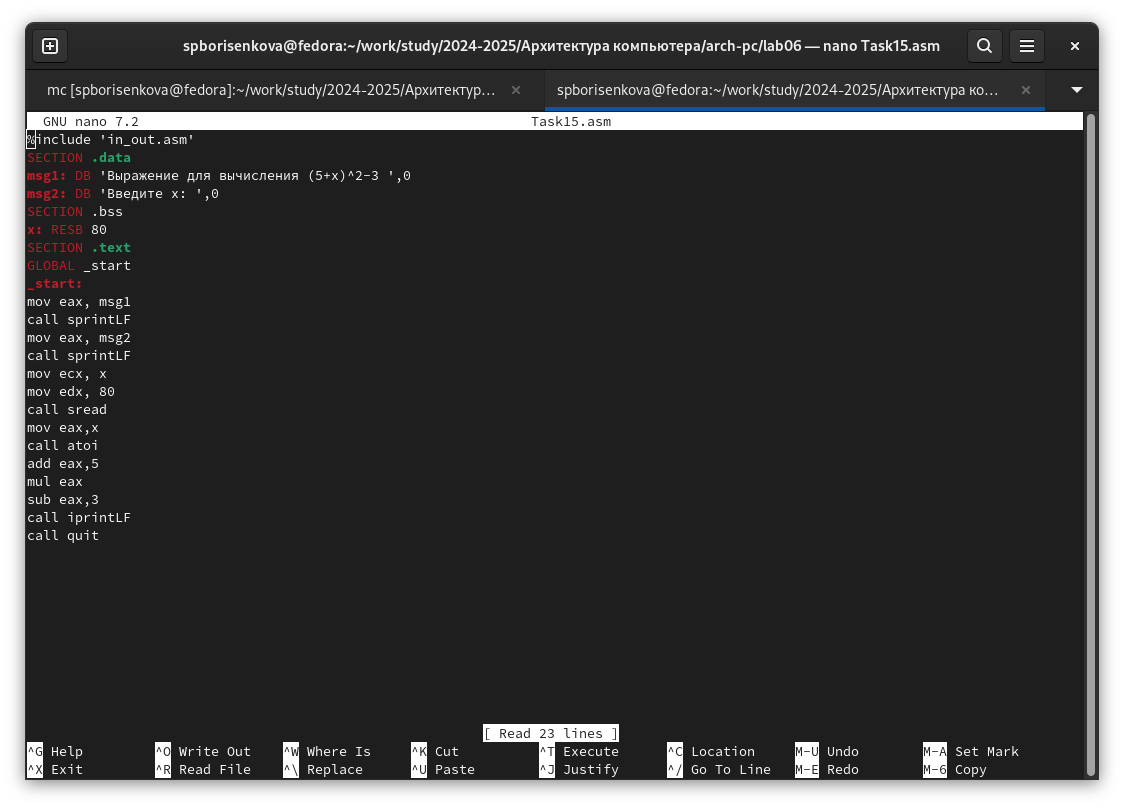
7. За это отвечают строки:

mov eax,edx

call iprintLF

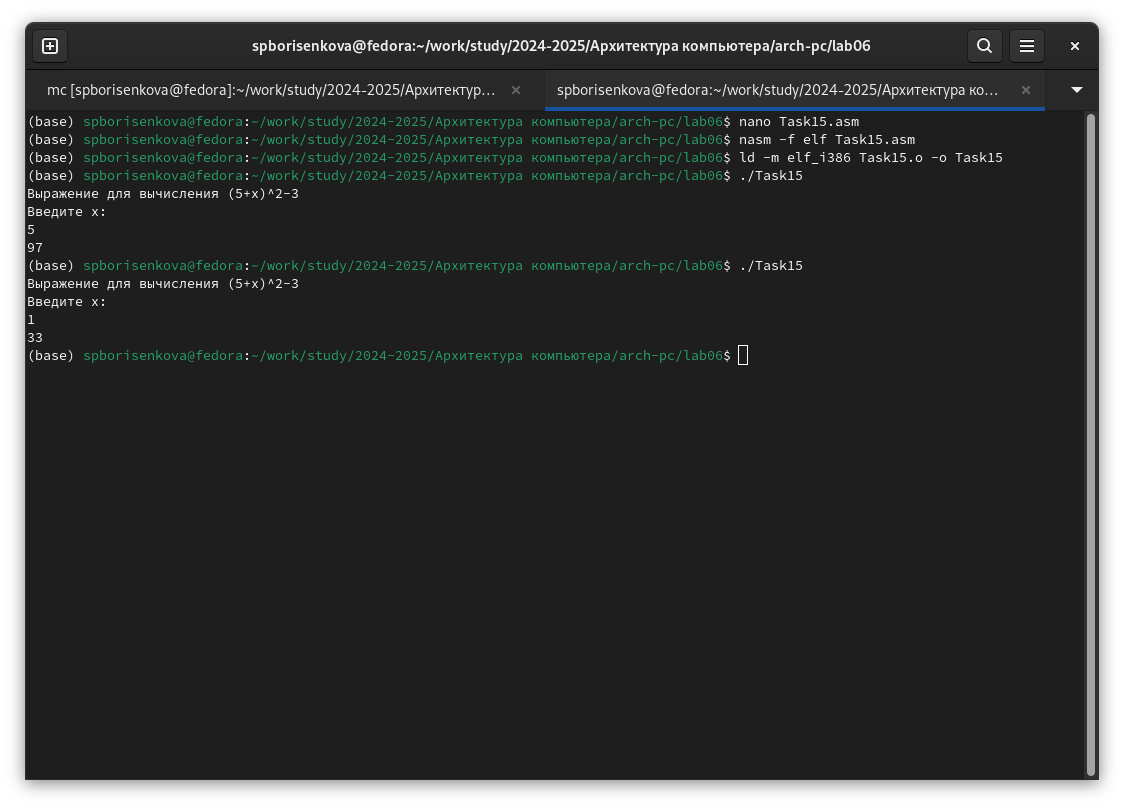
# 3 Выполнение задания для самостоятельной работы

Теперь в качестве самостоятельной работы напишем код программы для вычисления выражения в варианте 15: (5+x)^2-3 . В предварительно созданном файле Task15.asm напишем следующий код (Рис. 3.1):



Код требуемой программы

И запустим его, указав в качестве x предложенные в лабораторной работе значения (Рис. 3.2):



Запуск программы и проверка её корректной работы

# 4 Выводы

В результате выполнения лабораторной работы было получено представление о том, какие арифметические операции есть в языке ассемблера, и как они работают. Были написаны программы, использующие в себе операции сложения, вычитания, умножения и деления.