Программирование цикла. Обработка аргументов командной строки

Лабораторная работа №8

Приходько Иван Иванович

Содержание

# 1 Цель работы

Научиться работать с циклами на языке Ассемблера, а также научиться обрабатывать аргументы командной строки

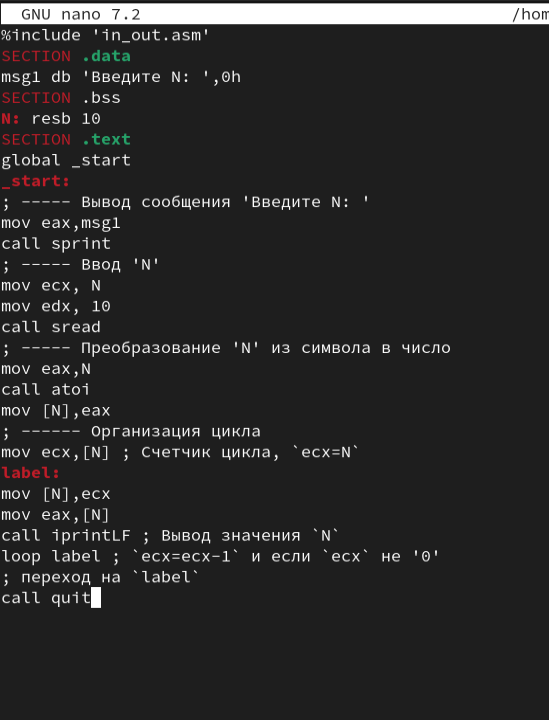
# 2 Выполнение лабораторной работы

Для начала выполнения лабораторной работы перейдем рабочую директорию и создадим файл lab8-1.asm (рис. 2.1).

Создание файла lab8-1.asm

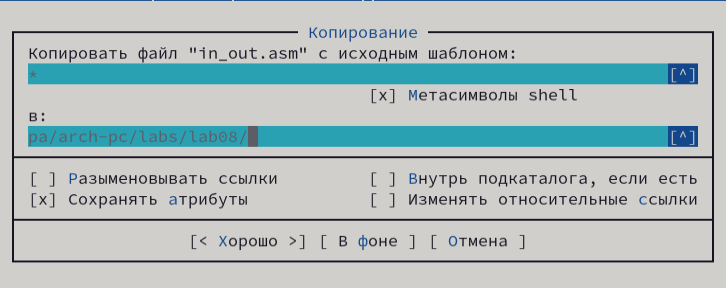
Создание файла lab8-1.asm

Теперь вставим код из листинга 8.1. Он должен запускать цикл и выводить каждую итерацию число, на единицу меньше предыдущего (рис. 2.2).



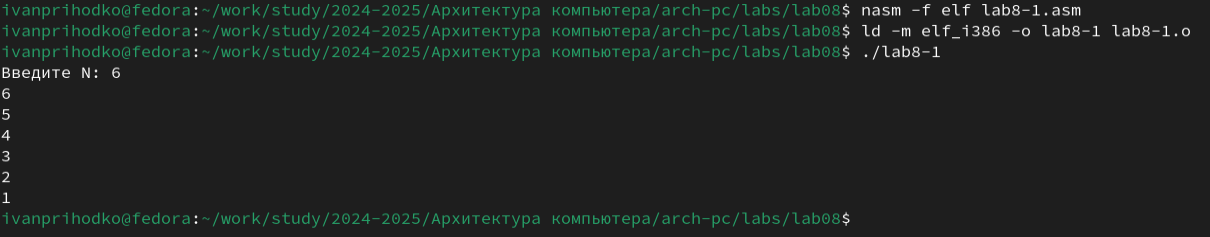
Вставка кода из листинга 8.1

Скопируем файл in\_out.asm из предыдущей работы (рис. 2.3).



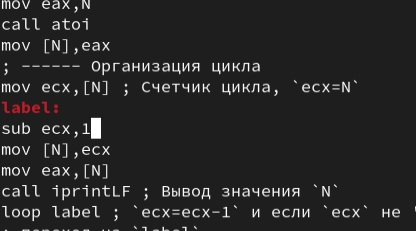
Копирование in\_out.asm

Теперь создадим и запустим файл (рис. 2.4).



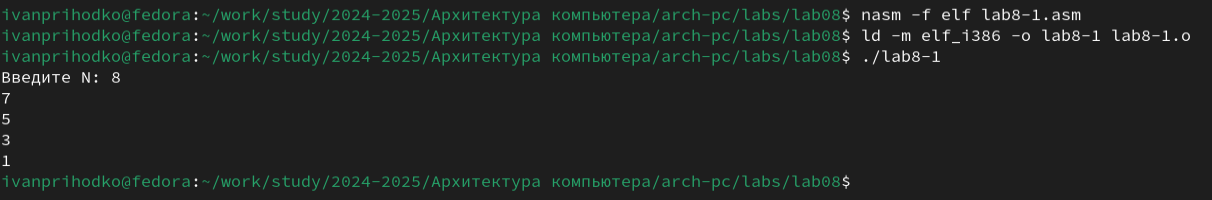
Запуск lab8-1.asm

Теперь попробуем изменить код, чтобы в цикле также отнималась единица у регистра ecx (рис. 2.5).



Редактирование lab8-1.asm

Теперь соберем и запустим файл (рис. 2.6).



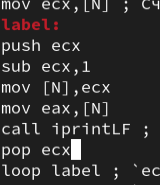
Повторный запуск lab8-1.asm



Повторный запуск lab8-1.asm

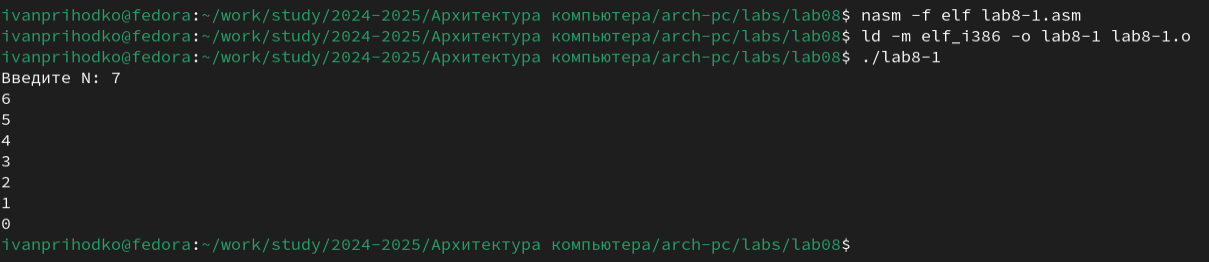
Как видим, цикл выполняется бесконечное количество раз. Это связано с тем, что цикл останавливается в тот момент, когда при проверке ecx равен 0, но он каждое выполнение цикла уменьшается на 2, из-за чего, в случае нечётного числа, никогда не достигнет нуля. Если на вход подать чётное число, цикл прогонится N/2 раз, выводя числа от N-1 до 1.

Теперь попробуем изменить программу так, чтобы она сохраняла значение регистра ecx в стек (рис. 2.8).



Редактирование файла lab8-1.asm

Теперь соберем и запустим файл (рис. 2.9).



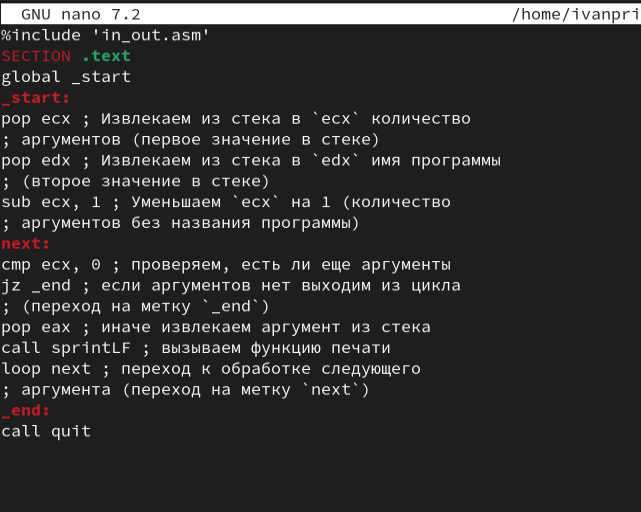
Повторный запуск lab8-1.asm

Теперь, программа выводит все числа от N-1 до нуля. Таким образом, число прогонов цикла равно числу N. Создадим второй файл (рис. 2.10).

Создание lab8-2.asm

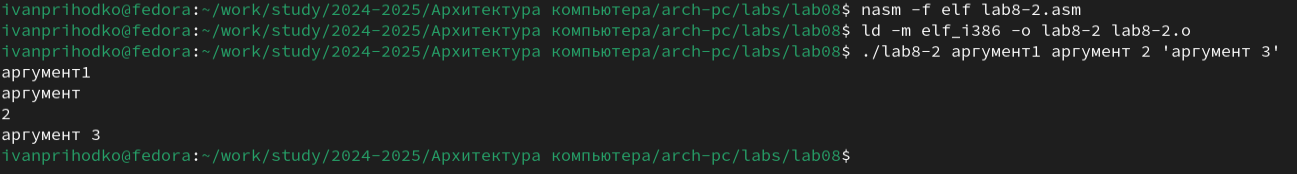
Создание lab8-2.asm

Вставим код из листинга 8.2 (рис. 2.11).



Редактирование lab8-2.asm

Теперь соберем и запустим файл (рис. 2.12).



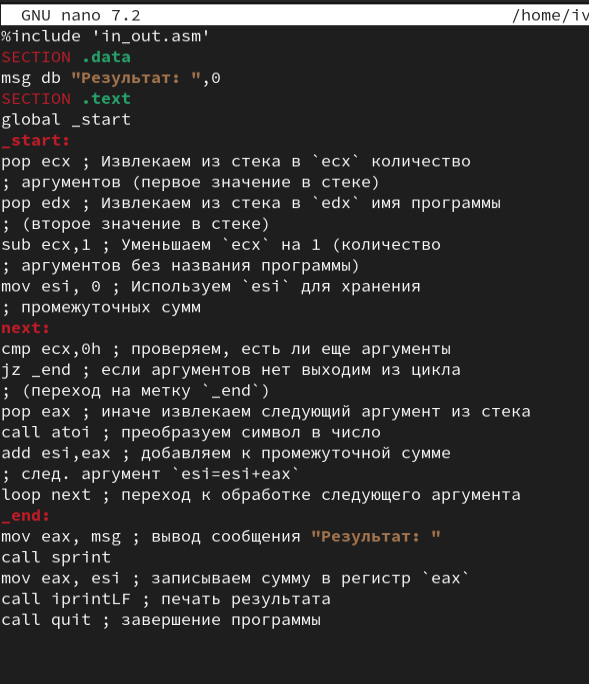
Запуск lab8-2.asm

Теперь создадим lab8-3.asm (рис. 2.13).

Создание lab8-3.asm

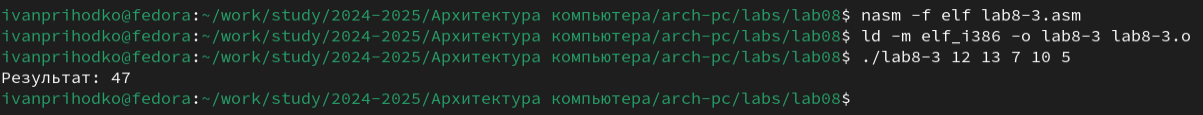
Создание lab8-3.asm

Вставим в него код из листинга 8.3 (рис. 2.14).



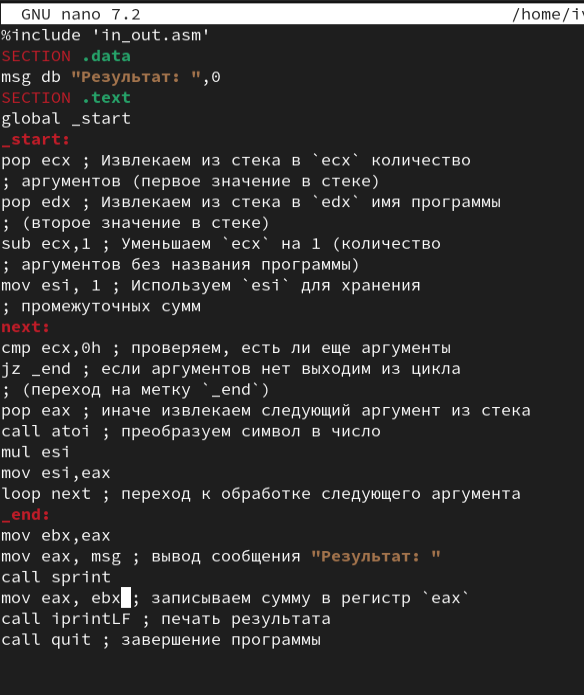
Редактирование lab8-3.asm

Соберем и запустим файл (рис. 2.15).



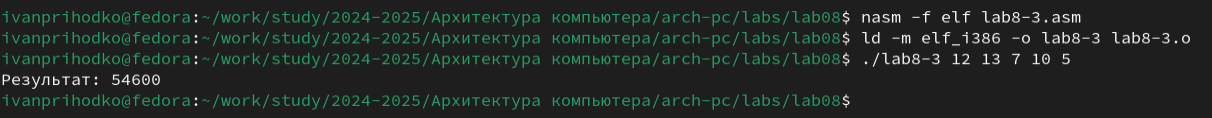
Запуск lab8-3.asm

Изменим файл так, чтобы она находила не сумму, а произведение всех аргументов (рис. 2.16).



Редактирование lab8-3.asm

Соберем и запустим файл (рис. 2.17).



Повторный запуск lab8-3.asm

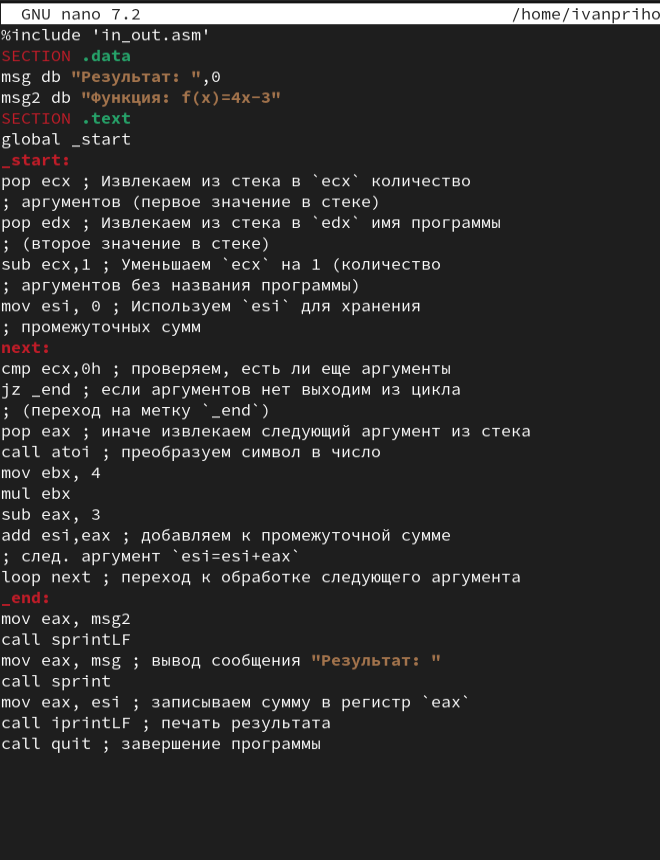
# 3 Выполнение лабораторной работы

Создадим файл для лабораторной работы (рис. 3.1).

Запуск task1v6.asm

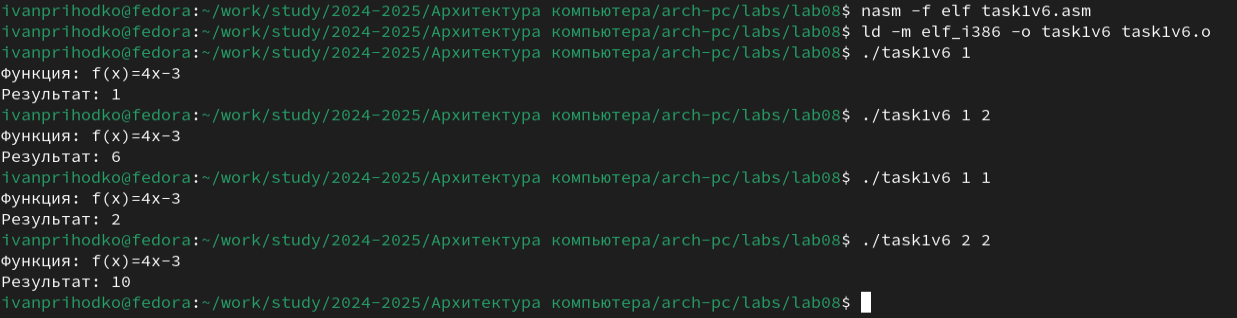
Запуск task1v6.asm

В рамках самостоятельной работы необходимо сделать задание под вариантом 6. Необходимо сложить результаты выполнения функции f(x)=4x-3 для всех введённых аргументов (рис. 3.2).



Код файла самостоятельной работы

Соберем и запустим файл (рис. 3.3).



Сборка и запуск task1v6.asm

# 4 Выводы

В результате выполнения лабораторной работы были получены навыки работы с циклами и обработкой аргументов из командной строки. Были написаны программы, использующие все вышеописанные аспекты.