Отчёт по лабораторной работе 8

Архитектура компьютера

Ридой Хайатуннаби

Содержание

[1 Цель работы 1](#_Toc158325821)

[2 Выполнение лабораторной работы 1](#_Toc158325822)

[2.1 Реализация циклов в NASM 1](#_Toc158325823)

[2.2 Задание для самостоятельной работы 10](#_Toc158325824)

[3 Выводы 12](#_Toc158325825)

# 1 Цель работы

Целью работы является приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки..

# 2 Выполнение лабораторной работы

## 2.1 Реализация циклов в NASM

Был организован каталог для выполнения лабораторного задания №8, в котором также был сформирован файл с наименованием lab8-1.asm.

Когда вы используете команду loop в NASM для создания циклических структур, важно учитывать, что она использует регистр ecx как счетчик, автоматически декрементируя его на один с каждым проходом цикла. Для наглядности рассмотрим пример кода, который демонстрирует значение регистра ecx.

В файл lab8-1.asm был введен код из примера 8.1. После этого была собрана исполняемая версия и проведена ее проверка.

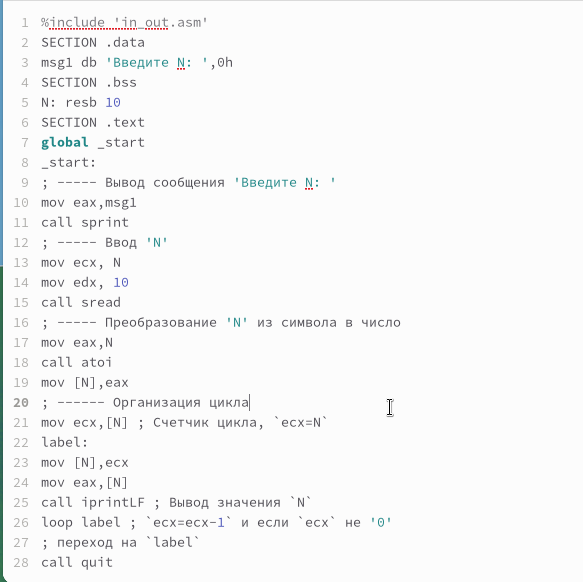


Figure 1: Код программы lab8-1.asm

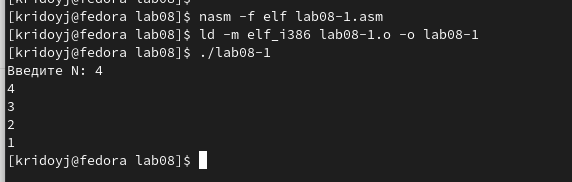


Figure 2: Компиляция и запуск программы lab8-1.asm

В данном случае видно, что использование регистра ecx в команде loop может стать причиной ошибочного поведения программы. Я изменил код, изменив обработку значения регистра ecx во время цикла.

Теперь программа входит в бесконечный цикл, если N нечетное, и выводит только нечетные числа, если N четное.

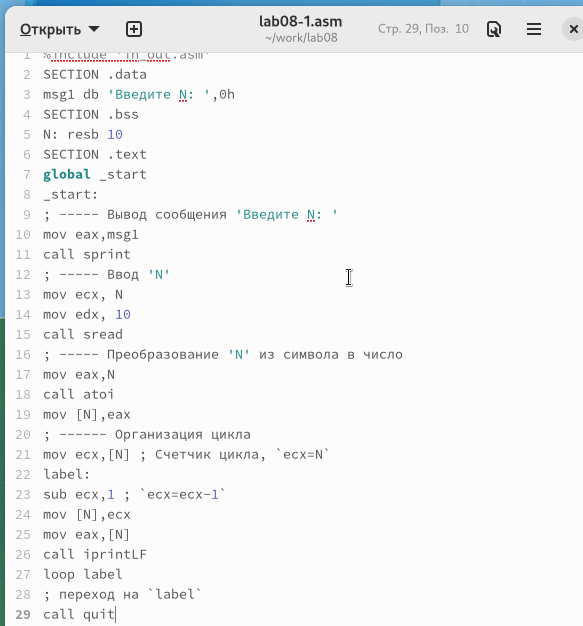


Figure 3: Код программы lab8-1.asm

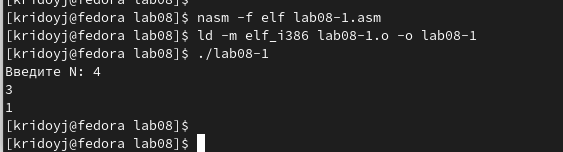


Figure 4: Компиляция и запуск программы lab8-1.asm

Для корректного использования регистра ecx в цикле и обеспечения правильной работы программы можно применить стек. Я модифицировал код, добавив инструкции push и pop, чтобы сохранить значение счетчика цикла loop в стеке.

Была сформирована исполняемая версия и осуществлена ее проверка. Программа отображает числа от N-1 до 0, где число итераций соответствует величине N.

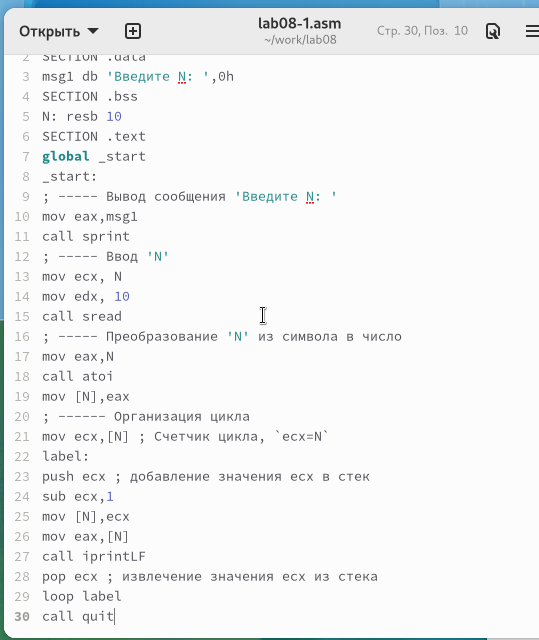


Figure 5: Код программы lab8-1.asm

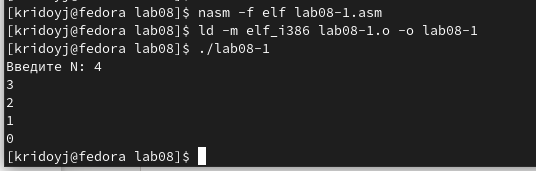


Figure 6: Компиляция и запуск программы lab8-1.asm

Я создал файл с именем lab8-2.asm в папке ~/work/arch-pc/lab08 и занес в него код, взятый из примера 8.2.

После этого я собрал исполняемый файл из исходного кода и запустил его с параметрами. В итоге программа успешно обработала пять переданных ей параметров. Под параметрами понимаются элементы, разделяемые пробелами, которые могут быть текстом или числами.

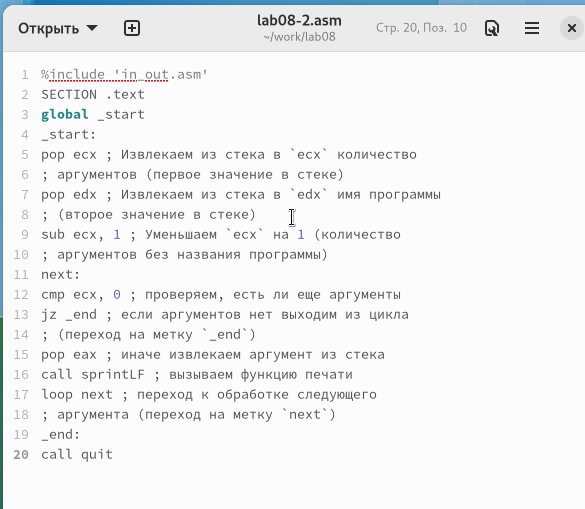


Figure 7: Код программы lab8-2.asm

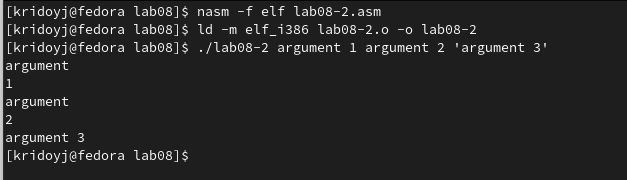


Figure 8: Компиляция и запуск программы lab8-2.asm

Теперь давайте рассмотрим другой пример программы, задачей которой является вывод на экран суммы чисел, передаваемых в неё в качестве параметров.

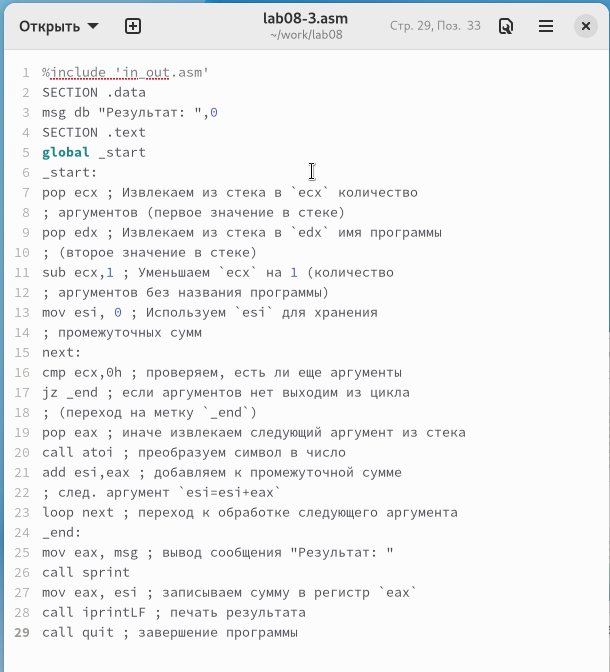


Figure 9: Код программы lab8-3.asm

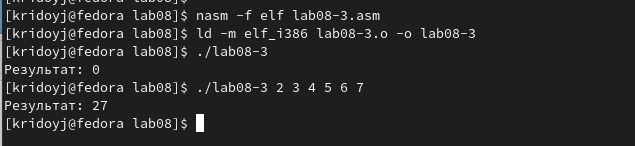


Figure 10: Компиляция и запуск программы lab8-3.asm

Я внес изменения в код из примера 8.3 таким образом, чтобы программа теперь вычисляла произведение значений, переданных через командную строку.

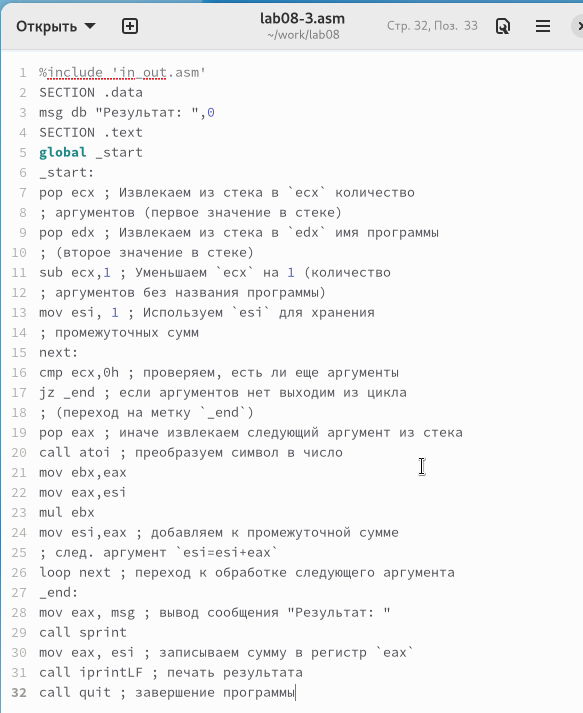


Figure 11: Код программы lab8-3.asm

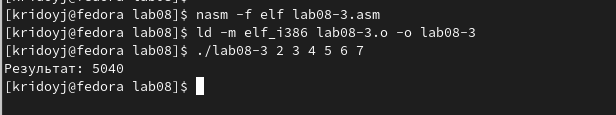


Figure 12: Компиляция и запуск программы lab8-3.asm

## 2.2 Задание для самостоятельной работы

Напишите программу, которая находит сумму значений функции для , т.е. программа должна выводить значение . Значения передаются как аргументы. Вид функции выбрать из таблицы 8.1 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу на нескольких наборах .

Мой вариант 16:

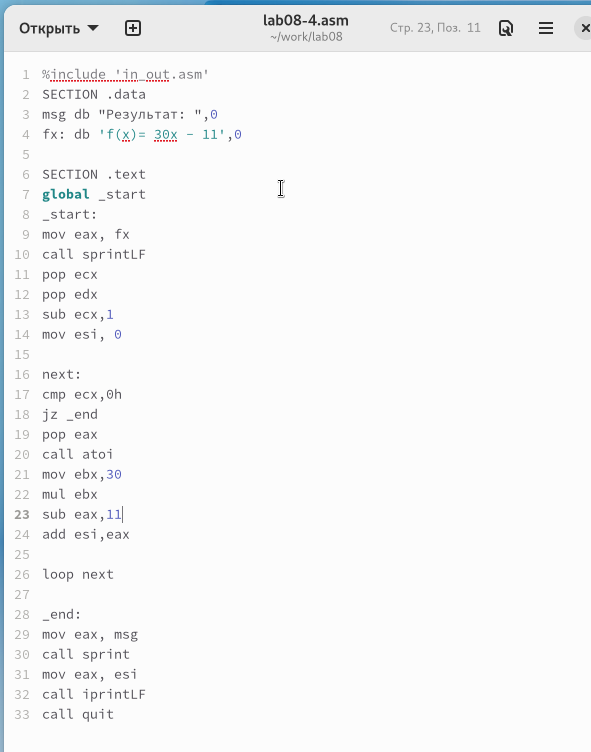


Figure 13: Код программы lab8-4.asm

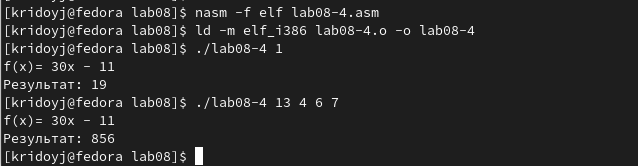


Figure 14: Компиляция и запуск программы lab8-4.asm

# 3 Выводы

Освоили работы со стеком, циклом и аргументами на ассемблере nasm.