Отчёт по лабораторной работе 8

Дисциплина: архитектура компьютера

Давлетова Мадина

Содержание

# 1 Цель работы

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

# 2 Задание

1. Изучение цмклов в ассемблере
2. Изучение стека в ассемблере
3. Изучение передачи аргументов в ассемблере
4. Выполнение заданий, рассмотрение примеров
5. Выполнение заданий для самостоятельной работы

# 3 Теоретическое введение

Стек — это структура данных, организованная по принципу LIFO («Last In — First Out» или «последним пришёл — первым ушёл»). Стек является частью архитектуры процессора и реализован на аппаратном уровне. Для работы со стеком в процессоре есть специальные регистры (ss, bp, sp) и команды.

Команда push размещает значение в стеке, т.е. помещает значение в ячейку памяти, на которую указывает регистр esp, после этого значение регистра esp увеличивается на 4.

Команда pop извлекает значение из стека, т.е. извлекает значение из ячейки памяти, на которую указывает регистр esp, после этого уменьшает значение регистра esp на 4. У этой команды также один операнд, который может быть регистром или переменной в памяти.

Для организации циклов существуют специальные инструкции. Для всех инструкций максимальное количество проходов задаётся в регистре ecx. Наиболее простой является инструкция loop. Она позволяет организовать безусловный цикл.

# 4 Выполнение лабораторной работы

## 4.1 Реализация циклов в NASM

Создаю каталог для выполнения лабораторной работы № 8 и файл с именем lab8-1.asm.

При использовании инструкции loop в NASM для реализации циклов, нужно помнить о том, что данная инструкция использует регистр ecx в качестве счетчика и на каждой итерации уменьшает его значение на единицу.

Рассмотрим пример программы, которая выводит значение регистра ecx. Написала текст программы из листинга 8.1 в файле lab8-1.asm. (рис. [[1](#fig:001)])

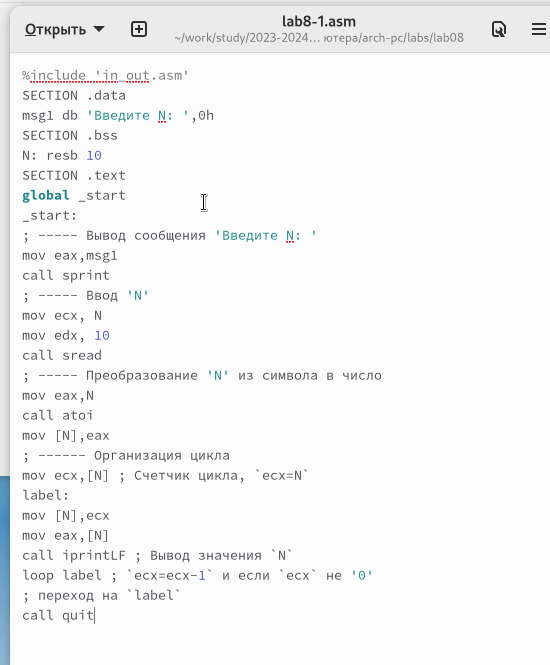


Figure 1: Изменение кода lab8-1.asm

Затем создаю исполняемый файл и проверю его работу.(рис. [[2](#fig:002)])

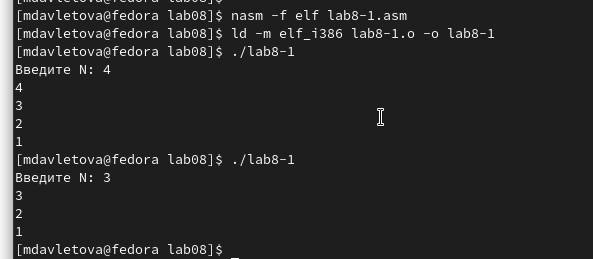


Figure 2: Компиляция текста программы lab8-1.asm

В данном примере демонстрируется, что использование регистра ecx в инструкции loop может привести к неправильной работе программы. В тексте программы вношу изменения, которые включают изменение значения регистра ecx внутри цикла.(рис. [[3](#fig:003)])

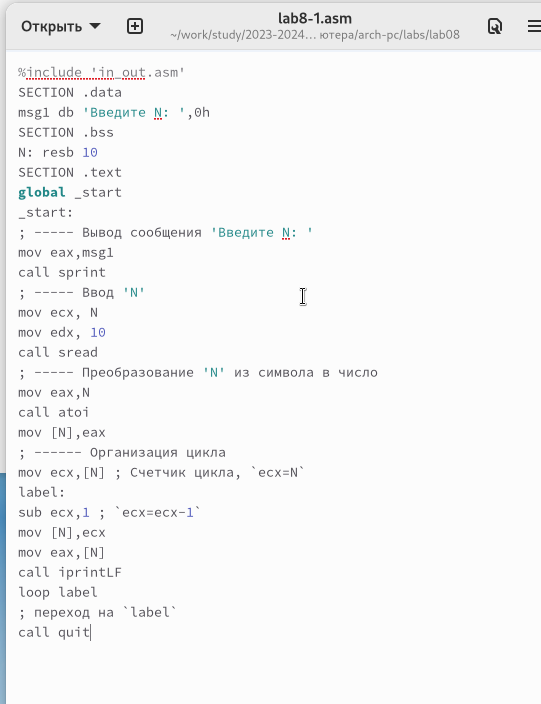


Figure 3: Изменение кода lab8-1.asm

Программа запускает бесконечный цикл при нечетном значении N и выводит только нечетные числа при четном значении N. (рис. [[4](#fig:004)])

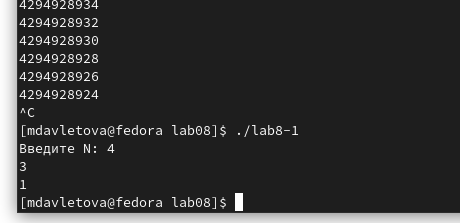


Figure 4: Компиляция текста программы lab8-1.asm

Для того чтобы использовать регистр ecx в цикле и обеспечить корректность работы программы, применяется стек. Вношу изменения в текст программы, добавив команды push и pop для сохранения значения счётчика цикла loop в стеке.(рис. [[5](#fig:005)])

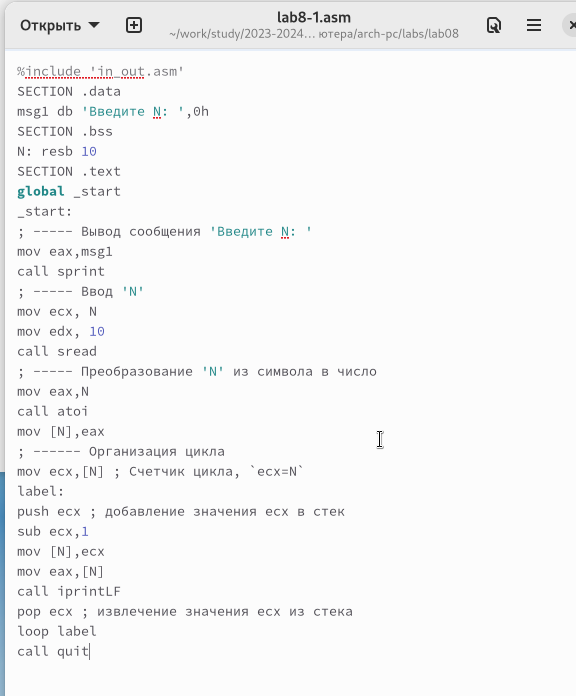


Figure 5: Изменение кода lab8-1.asm

Был создан исполняемый файл и проверена его работа. Программа выводит числа от N-1 до 0, где количество проходов цикла соответствует значению N.(рис. [[6](#fig:006)])

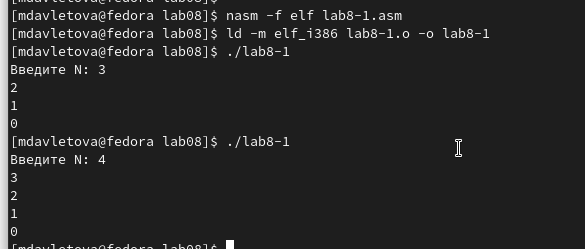


Figure 6: Компиляция текста программы lab8-1.asm

## 4.2 Обработка аргументов командной строки

Я создала файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и ввела в него текст программы из листинга 8.2. (рис. [[7](#fig:007)])

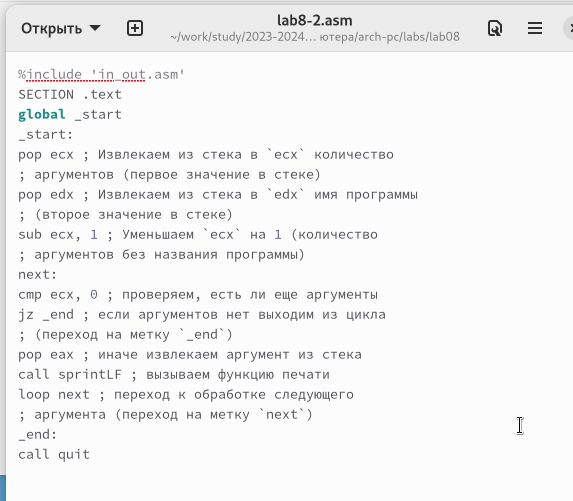


Figure 7: Изменение кода lab8-2.asm

Затем я создала исполняемый файл и запустила его, указав аргументы. Программа успешно обработала 5 аргументов. Аргументами считаются слова/числа, разделенные пробелом. (рис. [[8](#fig:008)]

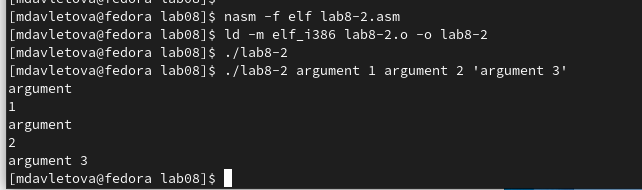


Figure 8: Компиляция текста программы lab8-2.asm

Теперь рассмотрим ещё один пример программы, которая выводит сумму чисел, переданных в программу как аргументы командной строки. (рис. [[9](#fig:009)]) (рис. [[10](#fig:010)])

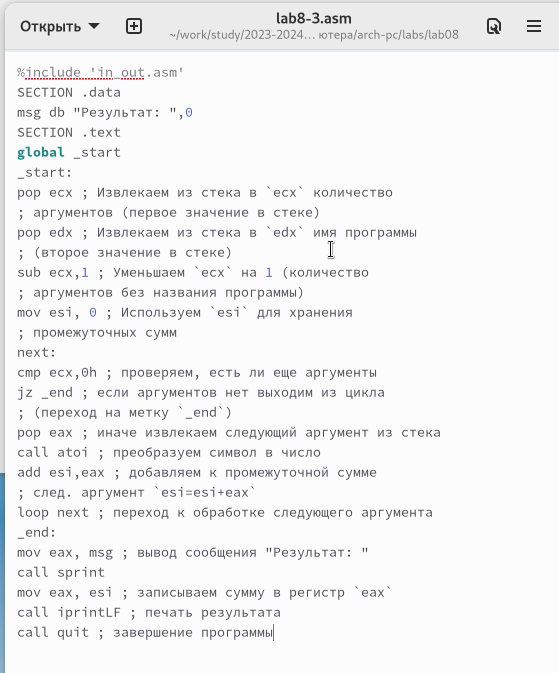


Figure 9: Изменение кода lab8-3.asm

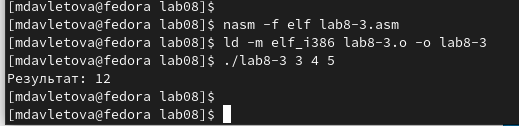


Figure 10: Компиляция текста программы lab8-3.asm

Я изменила текст программы из листинга 8.3 для вычисления произведения аргументов командной строки. (рис. [[11](#fig:011)]) (рис. [[12](#fig:012)])

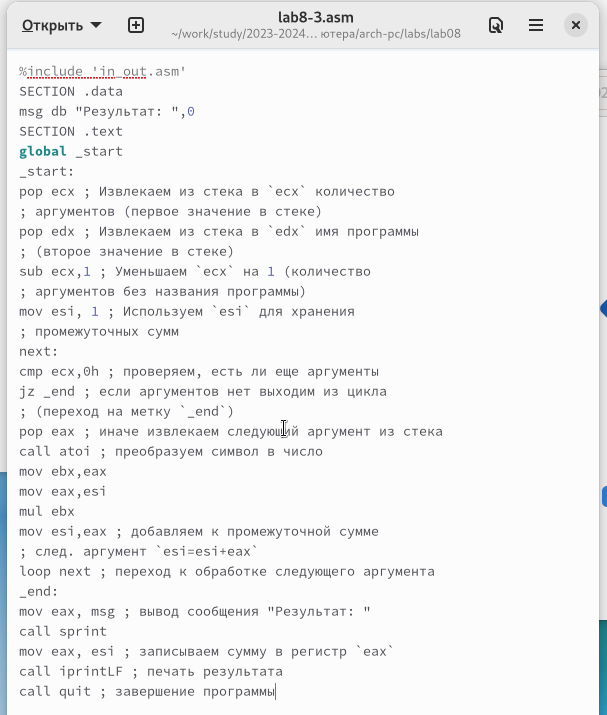


Figure 11: Изменение кода lab8-3.asm

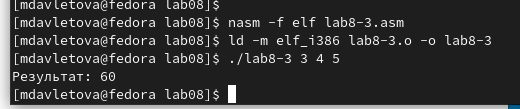


Figure 12: Компиляция текста программы lab8-3.asm

## 4.3 Выполнение заданий 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 .(рис. [[13](#fig:013)]) (рис. [[14](#fig:014)])

Мой вариант 7:

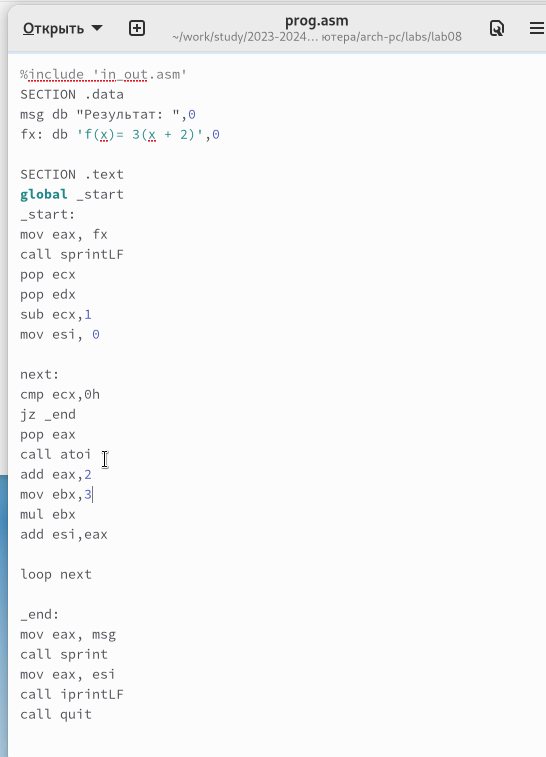


Figure 13: Изменение кода prog.asm

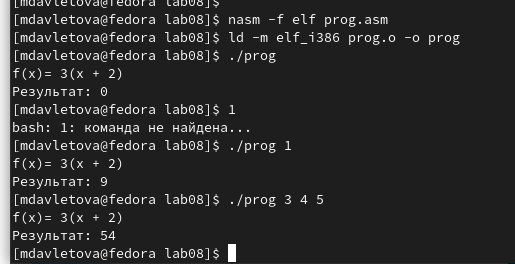


Figure 14: Компиляция текста программы prog.asm

# 5 Выводы

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