ОТЧЕТ по лабораторной работе №8

дисциплина: Архитектура компьютера

Студент: Идрисов Д.А.

Содержание

# 1 Цель работы

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

# 2 Задание

1. Изучение стека и цикла в ассемблере
2. Изучение примеров программ со стеком
3. Изучение примеров программ с аргументами
4. 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.

Для стека существует две основные операции:

* добавление элемента в вершину стека (push);
* извлечение элемента из вершины стека (pop).

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. Данная команда имеет один операнд — значение, которое необходимо поместить в стек.

Существует ещё две команды для добавления значений в стек. Это команда pusha, которая помещает в стек содержимое всех регистров общего назначения в следующем порядке: ах, сх, dx, bх, sp, bp, si, di. А также команда pushf, которая служит для перемещения в стек содержимого регистра флагов. Обе эти команды не имеют операндов.

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.

Аналогично команде записи в стек существует команда popa, которая восстанавливает из стека все регистры общего назначения, и команда popf для перемещения значений из вершины стека в регистр флагов.

# 4 Выполнение лабораторной работы

## 4.1 Реализация циклов в NASM

Создал каталог для программам лабораторной работы № 8 и файл lab8-1.asm

При использовании инструкции loop в NASM для реализации циклов, необходимо учитывать, что она использует регистр ecx в качестве счетчика и на каждом шаге уменьшает его значение на единицу. Для наглядности рассмотрим программу, которая выводит значение регистра ecx.

Написал в файл lab8-1.asm текст программы из листинга 8.1. Создал исполняемый файл и проверил его работу.

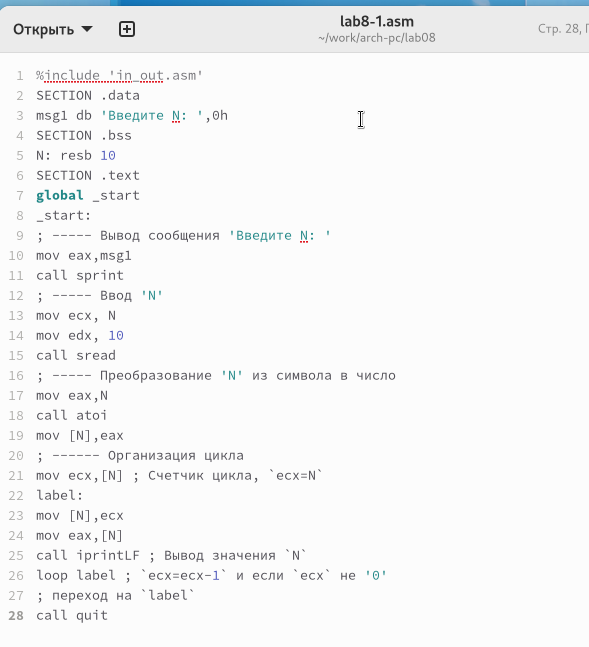


Figure 1: Изменение кода

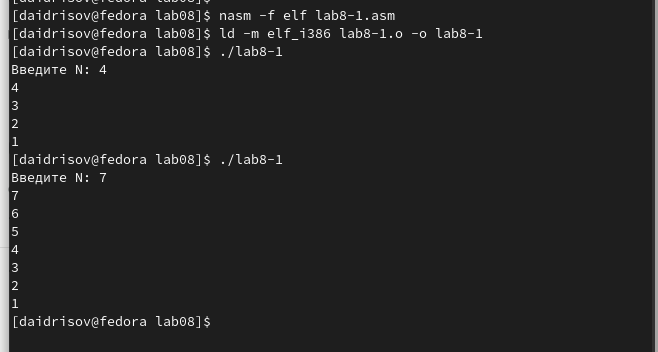


Figure 2: Запуск программы

Однако, в данном примере становится очевидно, что использование регистра ecx в теле цикла loop может привести к некорректной работе программы. Чтобы исправить это, мы можем использовать стек для сохранения значения счетчика цикла loop. Программа запускает бесконечный цикл при нечетном N и выводит только нечетные числа при четном N.

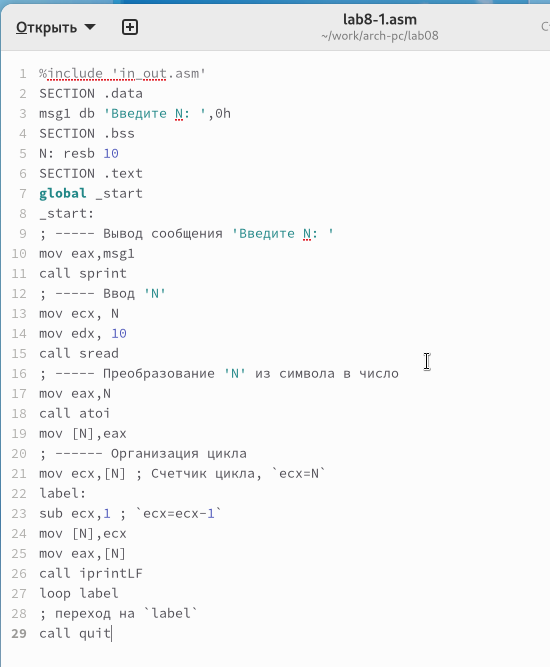


Figure 3: Изменение кода

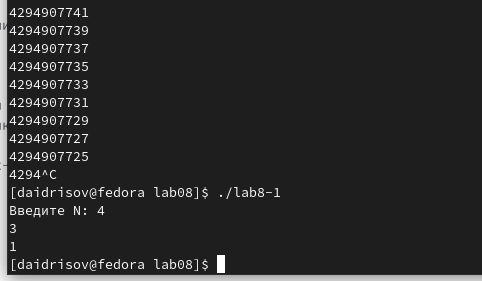


Figure 4: Запуск программы

Внесем изменения в текст программы, добавив команды push и pop, чтобы сохранить и извлечь значение счетчика из стека соответственно. После этого создадим исполняемый файл и проверим его работу. Таким образом, мы обеспечим корректность работы программы.

Создал исполняемый файл и проверьте его работу. Программа выводит числа от N-1 до 0, число проходов цикла соответсвует N.

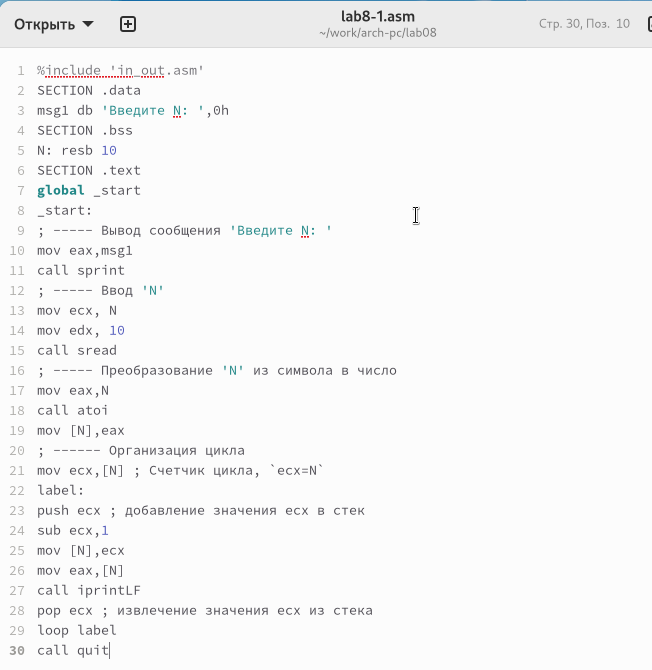


Figure 5: Изменение кода

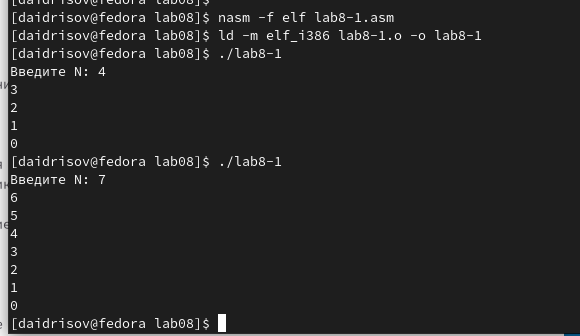


Figure 6: Запуск программы

## 4.2 Обработка аргументов командной строки

Создал файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и ввел в него текст программы из листинга 8.2.

Создал исполняемый файл и запустил его, указав аргументы. Программа обработала 5 аргументов. Аргументами считаются слова/числа, разделенные пробелом.

![Изменение кода(image/07.png){ #fig:007 width=70%, height=70% }

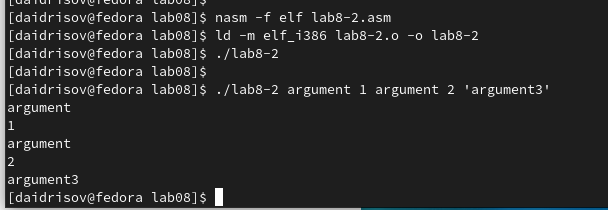


Figure 7: Запуск программы

Рассмотрим еще один пример программы которая выводит сумму чисел, которые передаются в программу как аргументы.

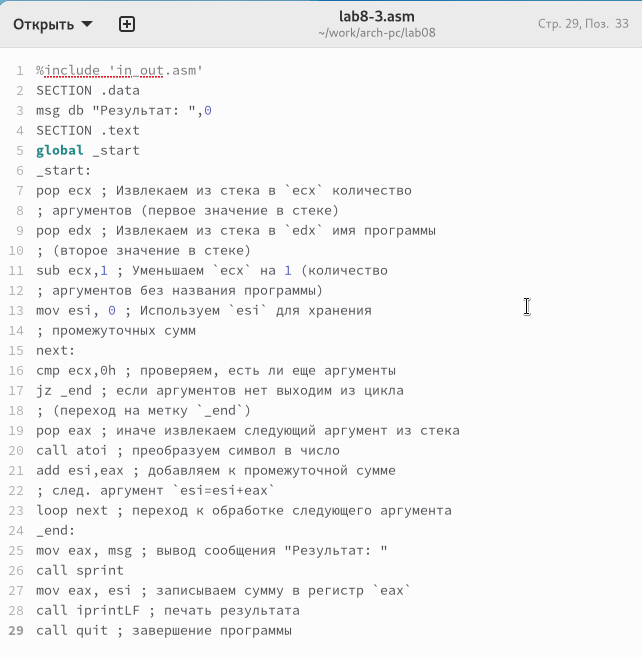


Figure 8: Изменение кода

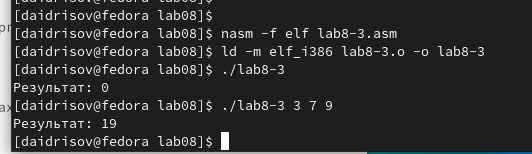


Figure 9: Запуск программы

Изменл текст программы из листинга 8.3 для вычисления произведения аргументов командной строки.

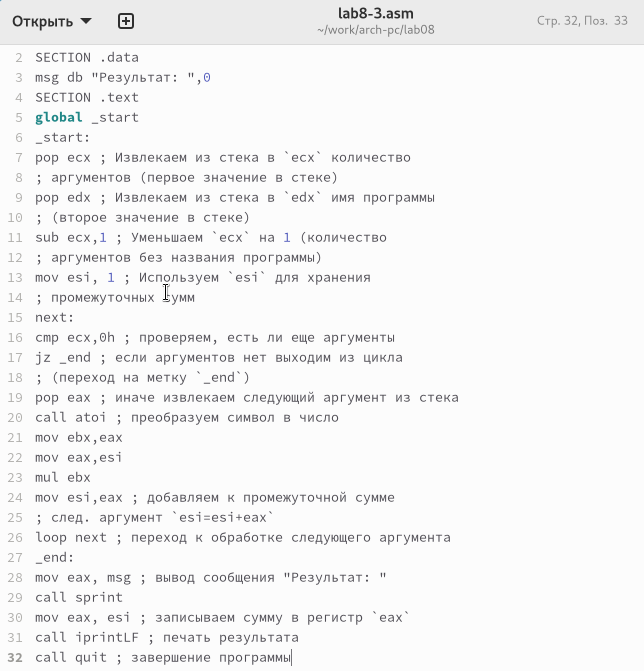


Figure 10: Изменение кода

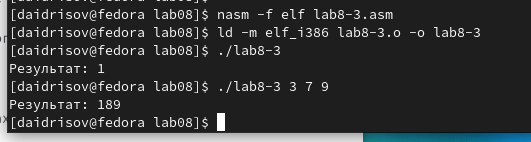


Figure 11: Запуск программы

## 4.3 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 .

для варианта 17

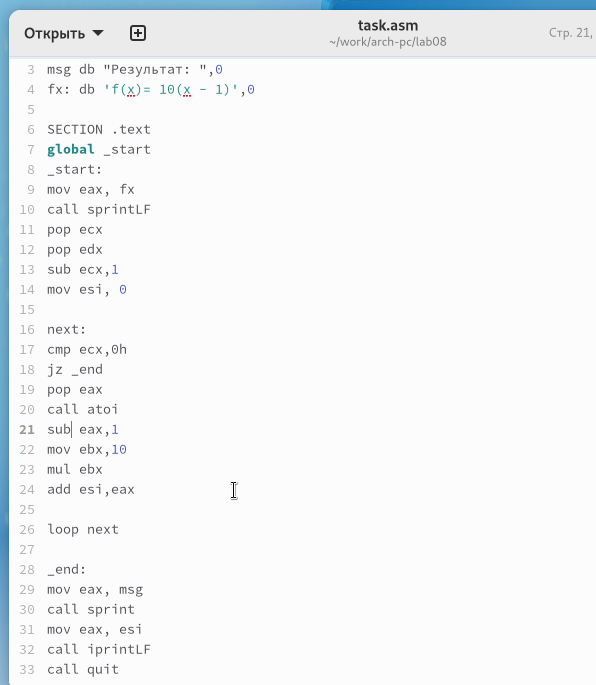


Figure 12: Изменение кода

Для проверки я запустил сначала с одним аргументом. Так, при подстановке

Затем подал несколько аргументов и получил сумму значений функции.

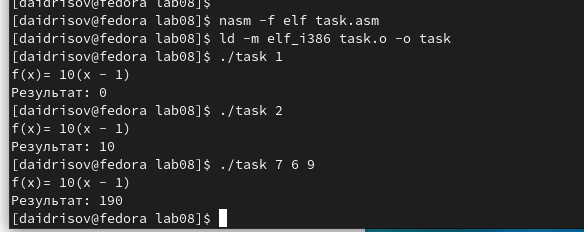


Figure 13: Запуск программы

# 5 Выводы

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