Отчёт по лабораторной работе 8

Архитектура компьютеров и операционные системы

Горелашвили Лия Михайловна НКАбд-03-23

Содержание

# 1 Цель работы

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

# 2 Задание

1. Изучение циклов и стека в ассемблере
2. Изучение процесса передачи аргументов командной строки
3. Рассмотрение примеров с циклами и стеком
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.

Основной функцией стека является функция сохранения адресов возврата и передачи аргументов при вызове процедур. Кроме того, в нём выделяется память для локальных переменных и могут временно храниться значения регистров.

Стек имеет вершину, адрес последнего добавленного элемента, который хранится в регистре esp (указатель стека). Противоположный конец стека называется дном. Значение, помещённое в стек последним, извлекается первым. При помещении значения в стек указатель стека уменьшается, а при извлечении — увеличивается.

Для стека существует две основные операции:

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

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. Иструкция loop выполняется в два этапа. Сначала из регистра ecx вычитается единица и его значение сравнивается с нулём. Если регистр не равен нулю, то выполняется переход к указанной метке. Иначе переход не выполняется и управление передаётся команде, которая следует сразу после команды loop.

# 4 Выполнение лабораторной работы

## 4.1 Реализация циклов в NASM

Я создаю папку для выполнения лабораторной работы № 8 и файл с именем lab8-1.asm.

Стоит отметить, что при использовании команды loop в NASM для реализации циклов, необходимо помнить, что эта команда использует регистр ecx в роли четчика и на каждом шаге уменьшает его значение на единицу.

Посмотрим на пример программы, которая выводит значение регистра ecx. В файл lab8-1.asm я внесла текст программы из листинга 8.1.

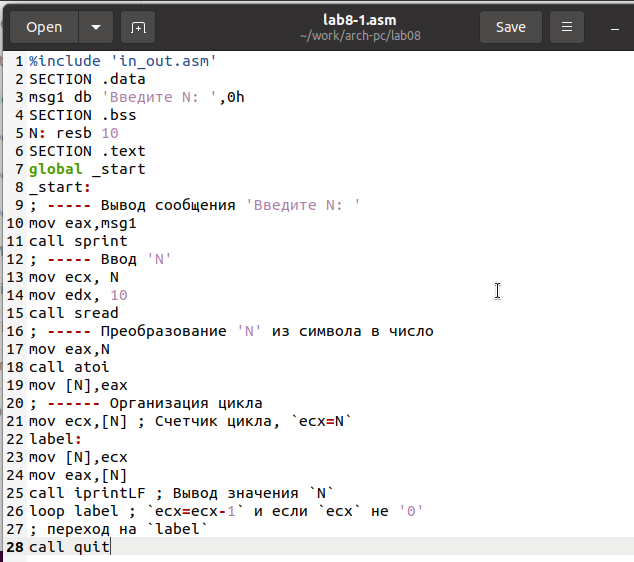


Figure 1: Редактирование файла lab8-1.asm

Далее, я создаю исполняемый файл и проверяю его функционирование.

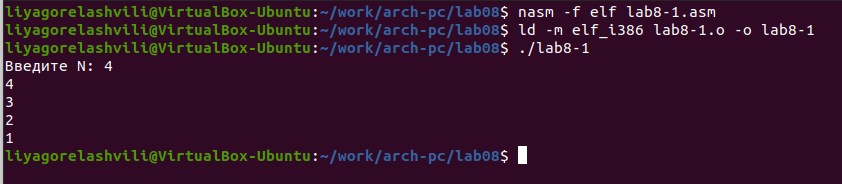


Figure 2: Тестирование программы lab8-1.asm

В этом примере показано, что использование регистра ecx в команде loop может привести к неправильному выполнению программы. В текст программы я вношу изменения, которые включают в себя изменение значения регистра ecx внутри цикла.

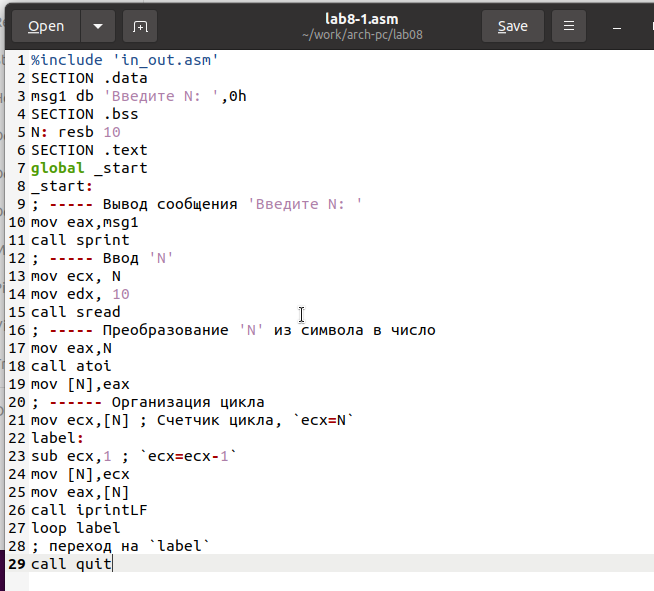


Figure 3: Редактирование файла lab8-1.asm

Программа запускает бесконечный цикл при нечетном значении N и выводит только нечетные числа при четном значении N.

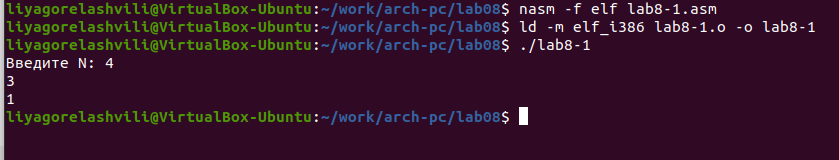


Figure 4: Тестирование программы lab8-1.asm

Чтобы использовать регистр ecx в цикле и обеспечить правильную работу программы, используется стек. Я внесла изменения в текст программы, добавив команды push и pop для сохранения значения счётчика цикла loop в стеке.

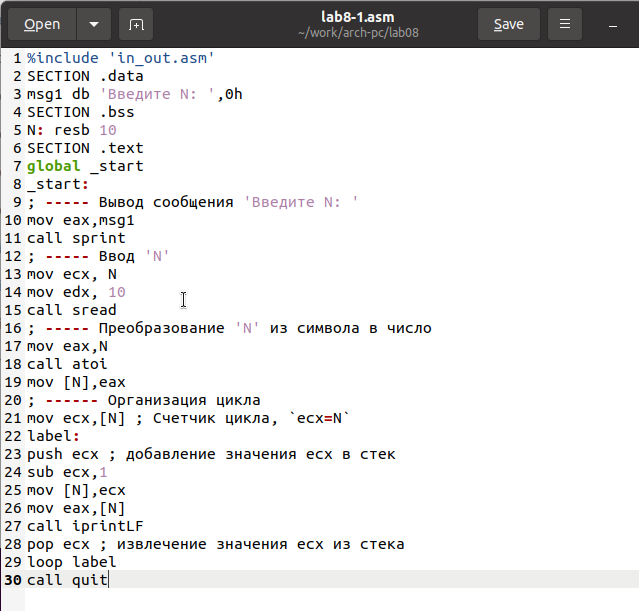


Figure 5: Редактирование файла lab8-1.asm

Затем был создан исполняемый файл и проверена его работа. Программа выводит числа от N-1 до 0, где количество проходов цикла соответствует значению N.

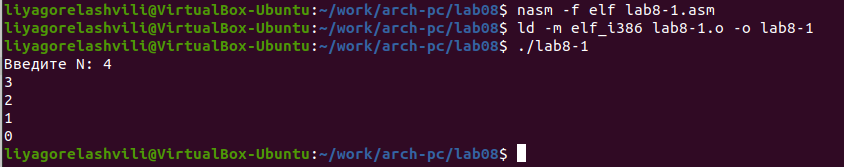


Figure 6: Тестирование программы lab8-1.asm

## 4.2 Обработка аргументов командной строки

Я изучила файл lab8-2.asm, в который внесла код программы из листинга 8.2.

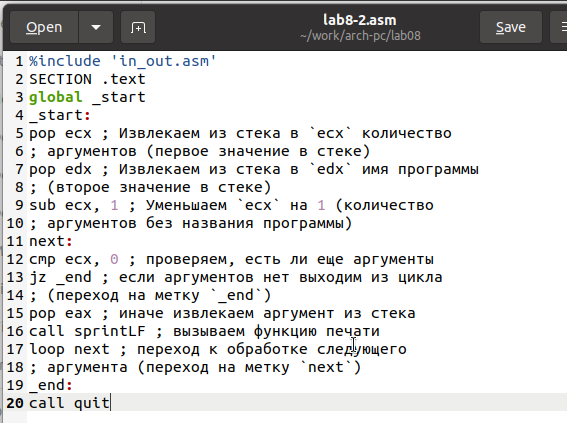


Figure 7: Редактирование файла lab8-2.asm

После этого был создан исполняемый файл, который я запустила с указанными аргументами. Программа эффективно обработала пять аргументов, которые представляют собой слова или числа, разделенные пробелами.

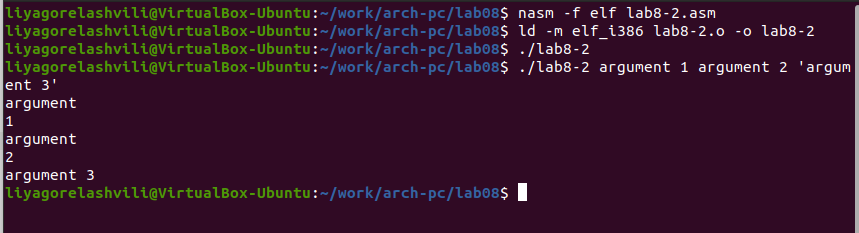


Figure 8: Тестирование программы lab8-2.asm

Давайте рассмотрим еще один пример программы. Эта программа выводит общую сумму чисел, которые были переданы в программу в качестве аргументов командной строки.

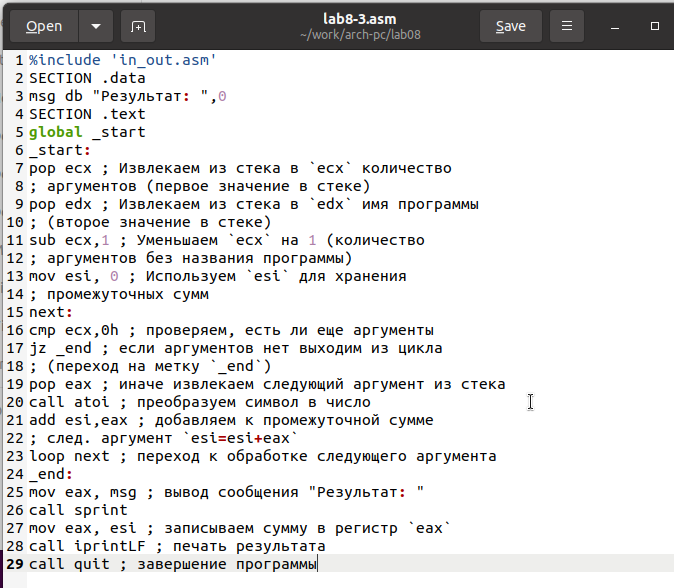


Figure 9: Редактирование файла lab8-3.asm

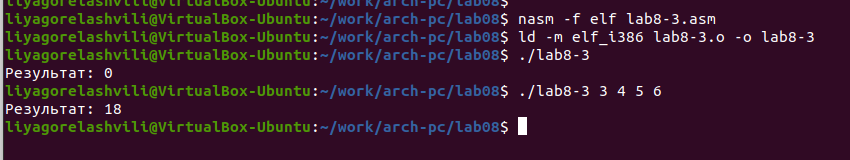


Figure 10: Тестирование программы lab8-3.asm

Я внесла изменения в код программы из листинга 8.3 с целью расчета произведения аргументов командной строки.

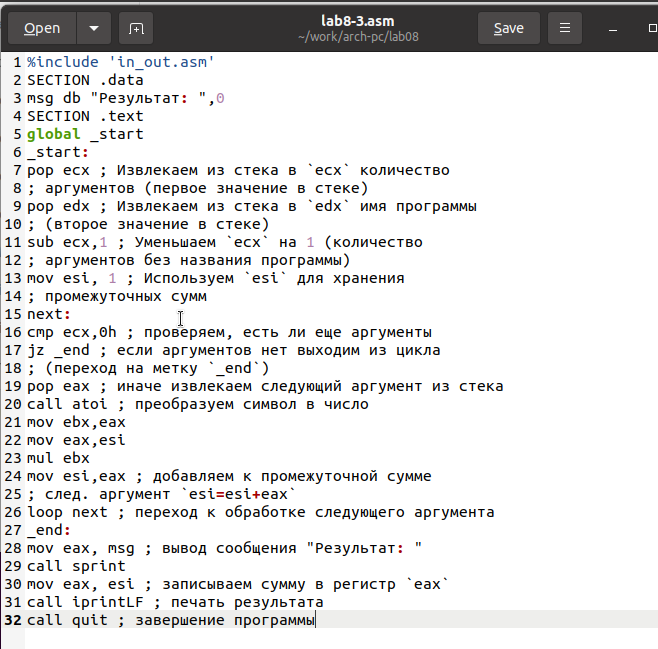


Figure 11: Редактирование файла lab8-3.asm

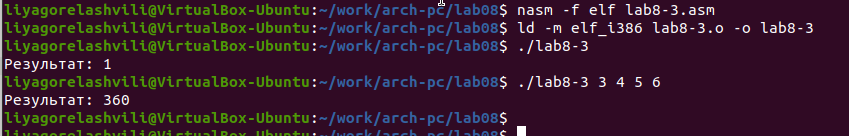


Figure 12: Тестирование программы lab8-3.asm

## 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 .

Мой вариант 8:

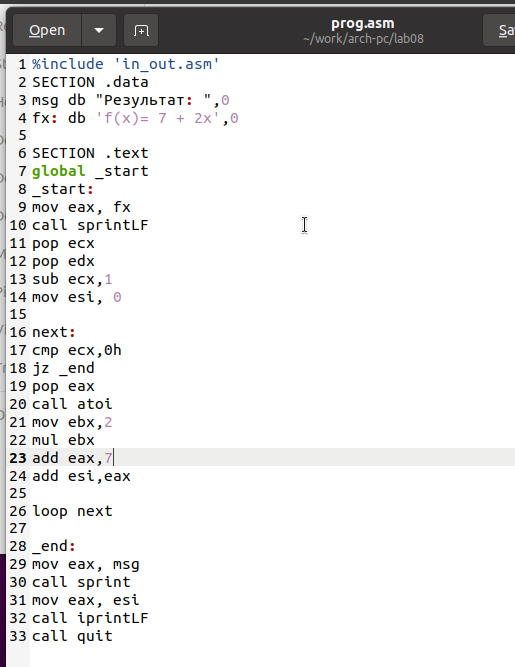


Figure 13: Редактирование файла prog.asm

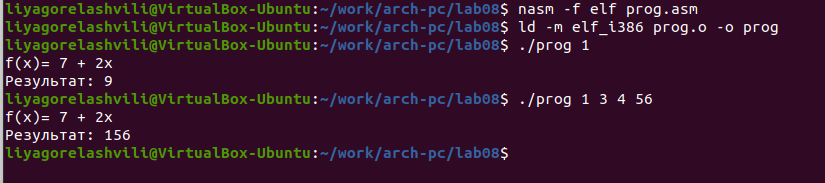


Figure 14: Тестирование программы prog.asm

# 5 Выводы

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