Отчёт по лабораторной работе 8

Программирование цикла. Обработка аргументов командной строки.

Зиборова Вероника Николаевна НММбд-02-24

Содержание

# 1 Цель работы

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

# 2 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).

# 3 Выполнение лабораторной работы

## 3.1 Реализация циклов в NASM

Создала каталог для программ лабораторной работы № 8 и файл lab8-1.asm.

При реализации циклов в NASM с использованием инструкции loop необходимо помнить, что эта инструкция использует регистр ecx в качестве счетчика, уменьшая его значение на единицу с каждым шагом. В качестве примера я рассмотрела программу, которая выводит текущее значение регистра ecx.

Добавила в файл lab8-1.asm текст программы из листинга 8.1. Затем создала исполняемый файл и проверила его работу.

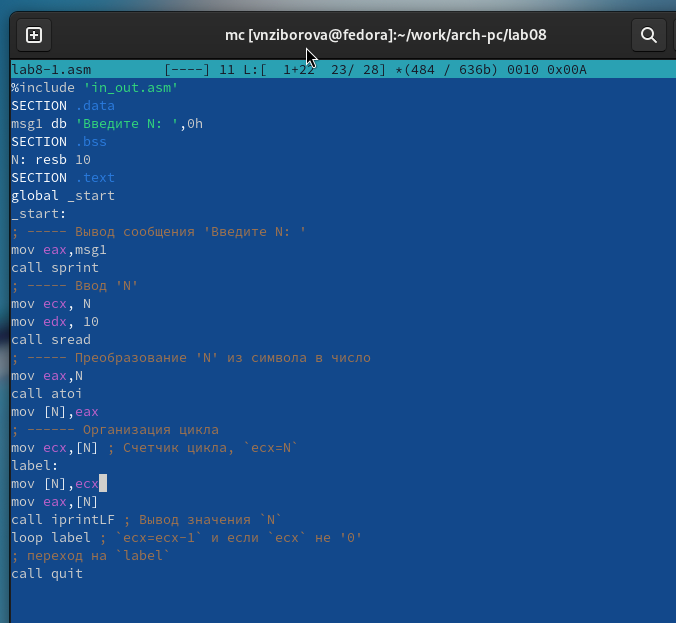


Рис. 1: Программа в файле lab8-1.asm

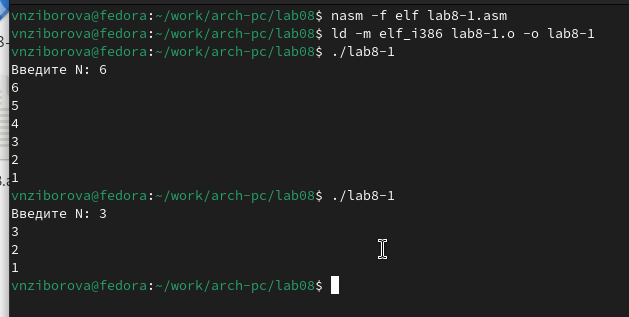


Рис. 2: Запуск программы lab8-1.asm

Этот пример показал, что изменение регистра ecx внутри цикла loop может привести к некорректной работе программы. Я изменила текст программы, добавив модификацию регистра ecx в цикле.

Полученная программа: - Запускает бесконечный цикл при нечетном значении N. - Выводит только нечетные числа, если N четное.

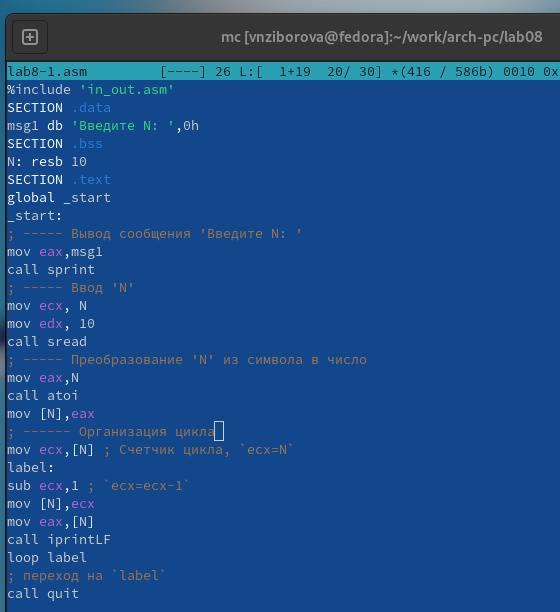


Рис. 3: Программа в файле lab8-1.asm

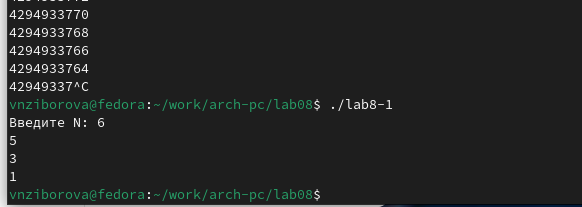


Рис. 4: Запуск программы lab8-1.asm

Для корректного использования регистра ecx в цикле, я добавила команды push и pop, чтобы временно сохранять значение регистра в стеке. Это позволило сохранить корректность работы программы. Внесла изменения в текст программы, создала исполняемый файл и проверила его работу.

Теперь программа: - Выводит числа от N-1 до 0. - Число проходов цикла соответствует значению N.

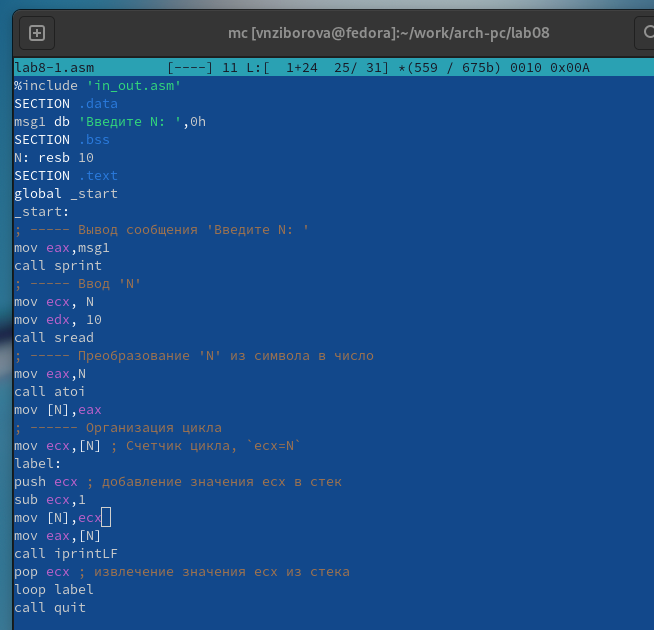


Рис. 5: Программа в файле lab8-1.asm

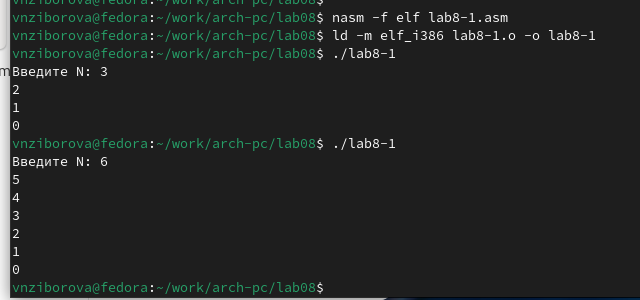


Рис. 6: Запуск программы lab8-1.asm

## 3.2 Обработка аргументов командной строки

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

Создала исполняемый файл и запустила его, указав аргументы. Программа обработала 5 аргументов, разделенных пробелами (слова или числа).

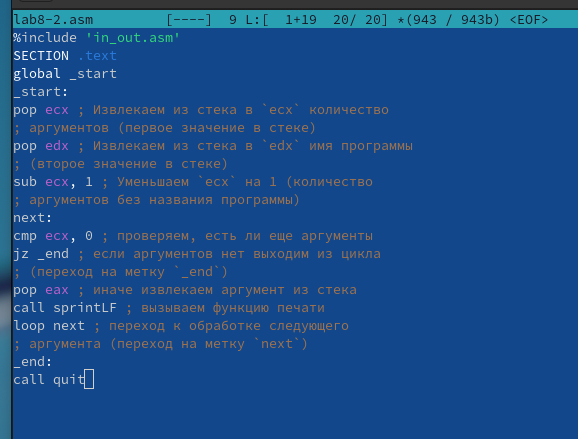


Рис. 7: Программа в файле lab8-2.asm

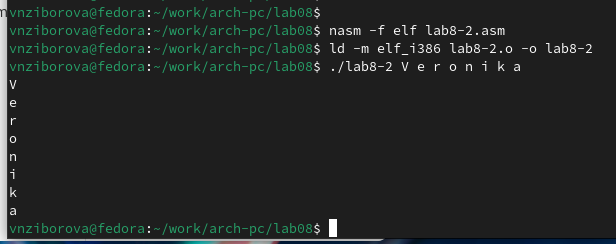


Рис. 8: Запуск программы lab8-2.asm

Рассмотрела пример программы, которая выводит сумму чисел, переданных в качестве аргументов командной строки.

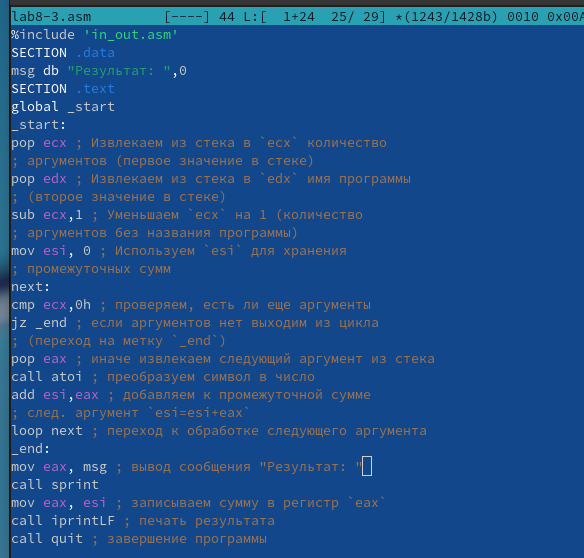


Рис. 9: Программа в файле lab8-3.asm

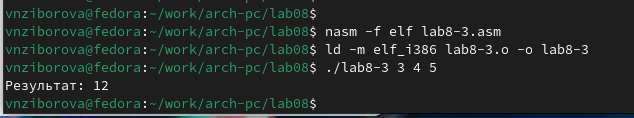


Рис. 10: Запуск программы lab8-3.asm

Изменила текст программы из листинга 8.3, чтобы вычислять произведение аргументов.

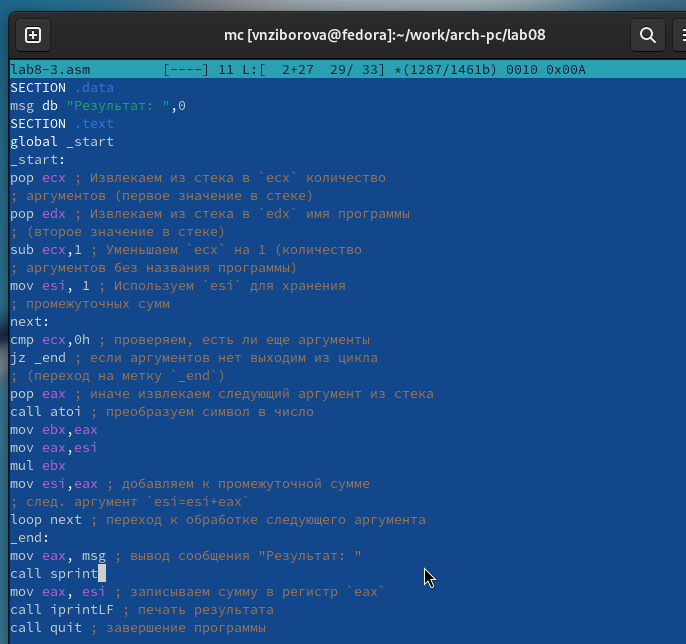


Рис. 11: Программа в файле lab8-3.asm

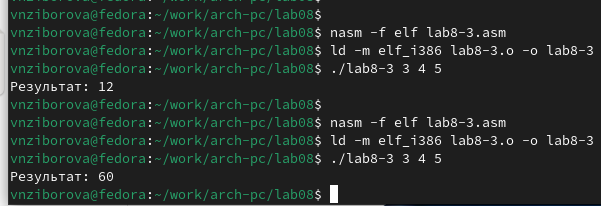


Рис. 12: Запуск программы lab8-3.asm

## 3.3 Задание для самостоятельной работы

Написала программу для нахождения суммы значений функции для . Программа должна выводить значение .

Значения передаются в программу как аргументы. Для функции выбрала вариант 6: .

Создала исполняемый файл и проверила его работу на нескольких наборах значений .

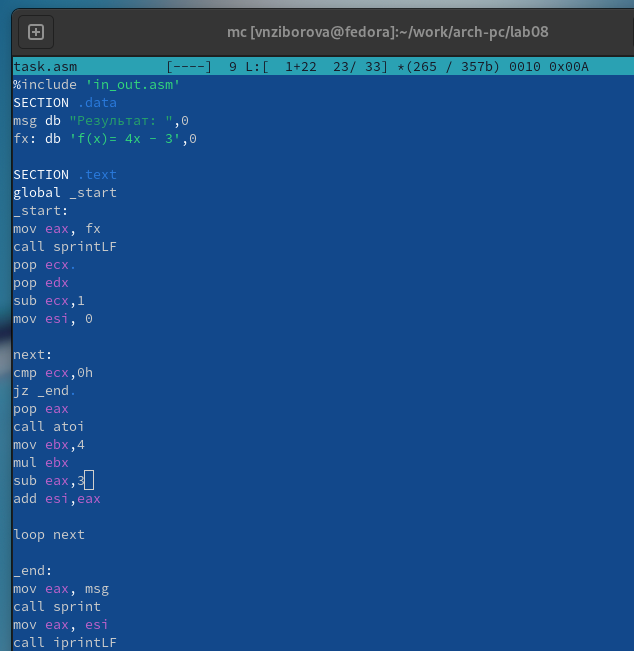


Рис. 13: Программа в файле task.asm

Для проверки запустила программу сначала с одним аргументом: - При , . - При , .

Затем передала несколько аргументов и получила сумму значений функции.

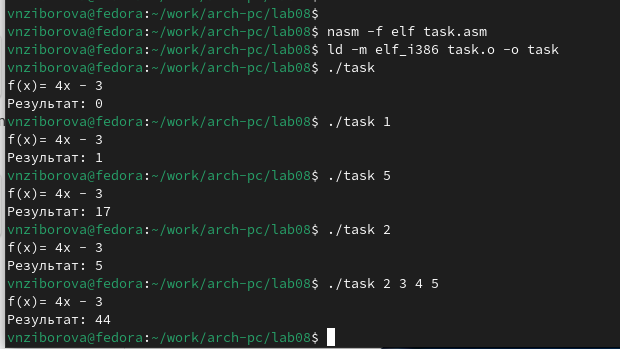


Рис. 14: Запуск программы task.asm

# 4 Выводы

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

# 5 Ответы на вопросы

1. **Опишите работу команды loop.**
   * Команда loop используется для организации циклов в NASM. Она уменьшает значение регистра ECX (или CX для 16-битных программ) на 1 и проверяет, не стало ли оно равным нулю. Если значение регистра не равно нулю, осуществляется переход по указанной метке. Если равно — выполнение программы продолжается с текущей позиции.
2. **Как организовать цикл с помощью команд условных переходов, не прибегая к специальным командам управления циклами?**
   * Можно использовать прямую арифметическую модификацию регистра и условные переходы. Например:
     1. Инициализировать регистр-счетчик (например, mov ecx, 10).
     2. Внутри цикла уменьшать значение регистра (dec ecx).
     3. Проверять с помощью условной команды (jnz, jz) и организовывать переход на начало цикла.
3. **Дайте определение понятия «стек».**
   * Стек — это структура данных, организованная по принципу «последним пришел — первым вышел» (LIFO, **Last In, First Out**). В контексте NASM стек используется для временного хранения данных, параметров функций, адресов возврата и других данных. Доступ к стеку осуществляется через команды push (помещение данных в стек) и pop (извлечение данных из стека).
4. **Как осуществляется порядок выборки содержащихся в стеке данных?**
   * Данные извлекаются из стека в обратном порядке их помещения. Это соответствует принципу LIFO:
     1. Последний добавленный элемент извлекается первым.
     2. При каждом извлечении указатель стека (регистр ESP для 32-битных программ или SP для 16-битных) увеличивается.
     3. Для доступа к данным используются команды pop (снять элемент) или прямой доступ через указатель стека.