Отчёт по лабораторной работе №8

дисциплина: Архитектура вычислительных систем

Мосолов Александр Денисович

Содержание

# 1 Цель работы

Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

# 2 Выполнение лабораторной работы

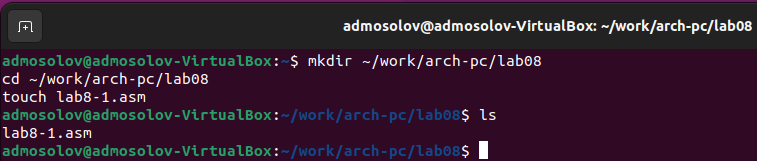
## 2.1 Реализация циклов в NASM

Создаём каталог для программ лабораторной работы № 8, переходим в него и создаём файл *lab8-1.asm*. С помощью команд:

mkdir ~/work/arch-pc/lab08

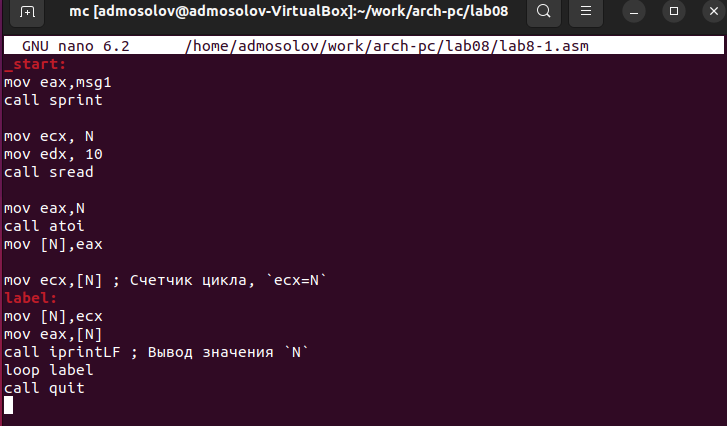
cd ~/work/arch-pc/lab08

touch lab8-1.asm



Создаем каталог для работы

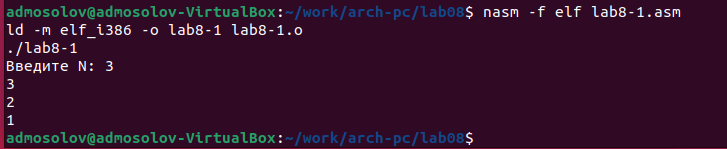
При реализации циклов в *NASM* с использованием инструкции *loop* необходимо помнить о том, что эта инструкция использует регистр *ecx* в качестве счетчика и на каждом шаге уменьшает его значение на единицу. В качестве примера рассмотрим программу, которая выводит значение регистра *ecx*. Внимательно изучим текст программы *(Листинг 8.1)*.



Текст программы в соответствии с листингом 8.1

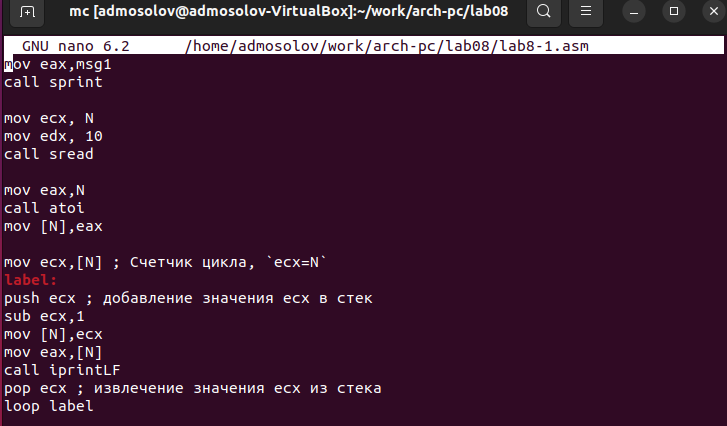
Для того, чтобы программа транслировалась без ошибок перенесем файл *in\_out.asm* в *~/work/arch-pc/lab08*.

Создадим исполняемый файл и запустим его. Результат работы данной программы будет следующим:



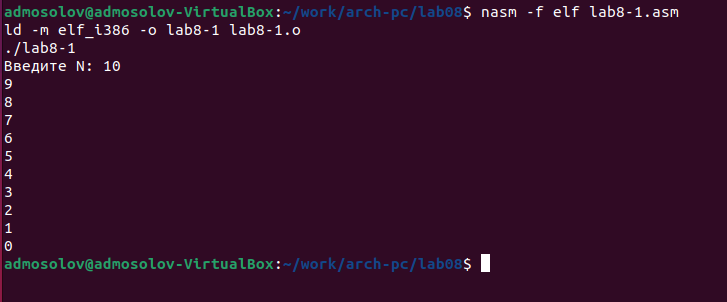
Вывод программы из листинга 8.1

Данный пример показывает, что использование регистра *ecx* в теле цилка *loop* может привести к некорректной работе программы. Изменим текст программы добавив изменение значение регистра *ecx* в цикле:



Изменённый текст программы в соответствии с листингом 8.1

Транслируем текст, создаём объектный файл, компилируем его и запускаем программу *lab8-1*:



Вывод изменённой программы из листинга 8.1

Число проходов цикла соответствует значению *N* введенному с клавиатуры, регистр *ecx* принимает значения все нечётные значения от *0* до *N*.

## 2.2 Обработка аргументов командной строки

Для того чтобы использовать аргументы в программе, их просто нужно извлечь из стека. Обработку аргументов нужно проводить в цикле. Т.е. сначала нужно извлечь из стека количество аргументов, а затем циклично для каждого аргумента выполнить логику программы. В качестве примера рассмотрим программу, которая выводит на экран аргументы командной строки. Внимательно изучим текст программы *(Листинг 8.2)*.



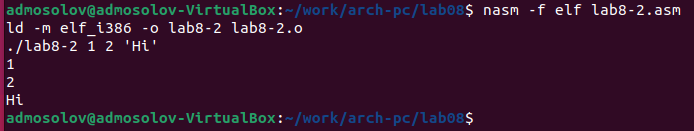
Текст программы lab8-2

Транслируем текст, создаём объектный файл, компилируем его и запускаем программу *lab8-2*:

*nasm -f elf lab8-2.asm*

*ld -m elf\_i386 -o lab8-2 lab8-2.o*

*./lab8-2 1 2 ‘Hi’*



Вывод программы lab8-2

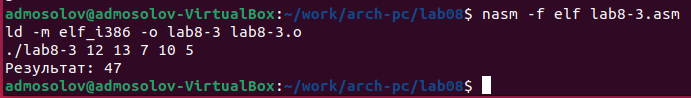
Программа обрабатывает *3* аргумента.

Рассмотрим еще один пример программы которая выводит сумму чисел, которые передаются в программу как аргументы. Создадим файл *lab8-3.asm* в каталоге *~/work/arch-pc/lab08* и введём в него текст программы из листинга *8.3*.



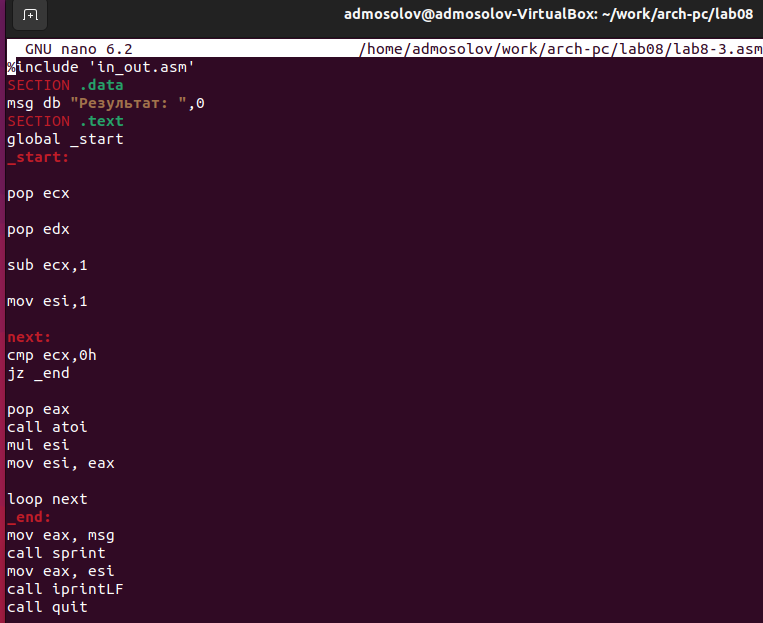
Текст программы из листинга 8.3

Создадим исполняемый файл и запустим его, указав аргументы.



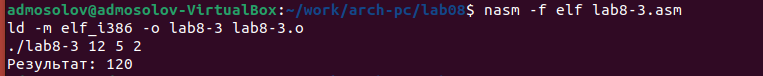
Проверяем работу программы из листинга 8.3

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



Изменённый текст программы lab8-3

Проверим работу программы:



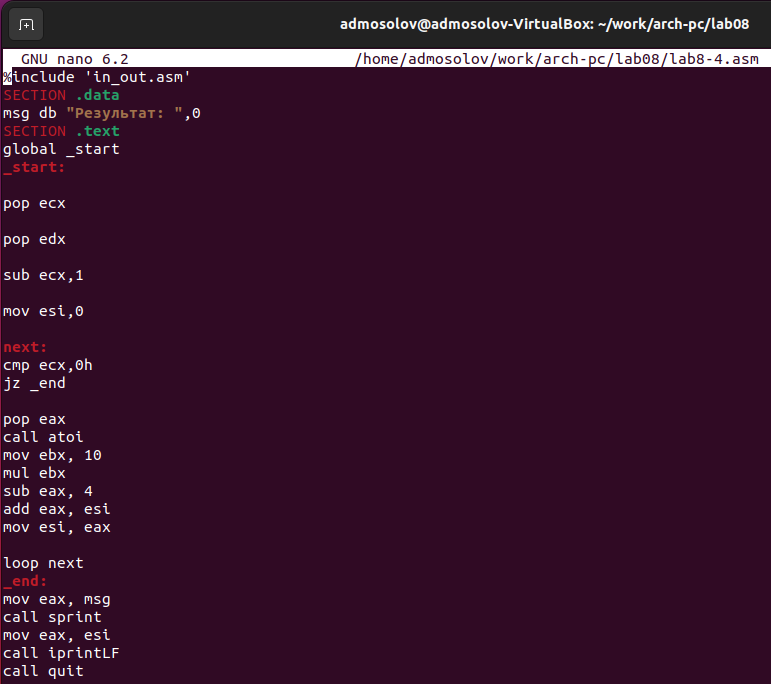
Вывод изменённой программы lab8-3

А теперь на вывод программы с использованием *iprint*.

## 2.3 Задание для самостоятельной работы

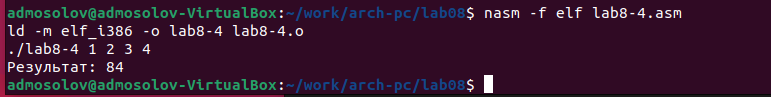
Напишите программу, которая находит сумму значений функции *f(x)* для *x = x1, x2, …, xn*, т.е. программа должна выводить значение *f(x1) + f(x2) + … + f(xn)*. Значения *xi* передаются как аргументы. Вид функции *f(x)* выбрать из таблицы *8.1* вариантов заданий в соответствии с вариантом *(в моём случае вариант 9)*, полученным при выполнении лабораторной работы *№ 7*. Создайте исполняемый файл и проверьте его работу на нескольких наборах *x = x1, x2, …, xn*.

Напишем текст программы для *f(x) = 10x - 4* (вариант *9*):



Текст программы из задания для самостоятельного выполнения *(lab8-4)*

Убедимся в правильности написанной программы.



Вывод программы из задания для самостоятельного выполнения

# 3 Выводы

В ходе лабораторной работы были приобретены навыки написания программ с использованием циклов и обработкой аргументов командной строки.