Отчет по лабораторной работе №8

Дисциплина: архитектура компьютера

Романова Елизавета Романовна

Содержание

# 1 Цель работы

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

# 2 Задание

1. Реализация циклом в NASM
2. Обработка аргументов командной строки
3. Самостоятельное написание программы по материалам лабораторной 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.

# 4 Выполнение лабораторной работы

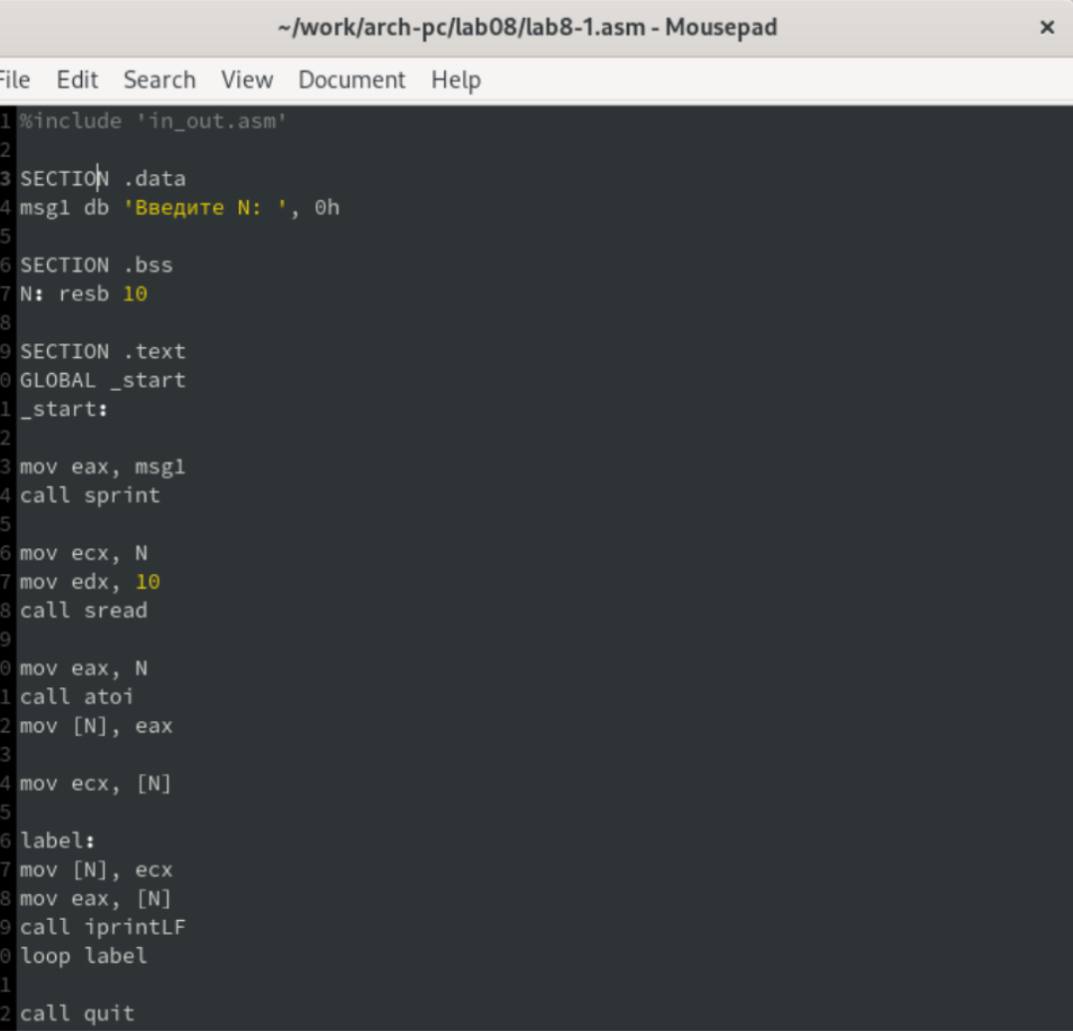
## 4.1 Реализация циклов в NASM

Создаю каталог для программ лабораторной работы №8.

Создание каталога

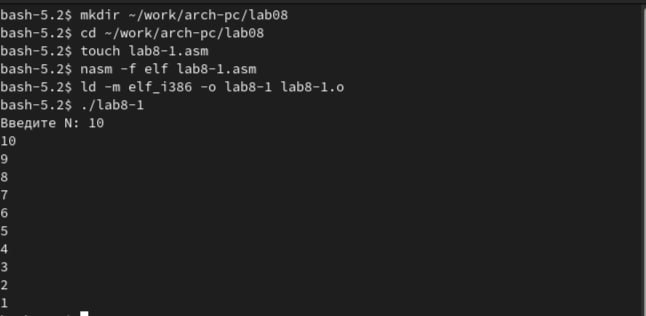
Создание каталога

Копирую в созданный файл программу из листинга.



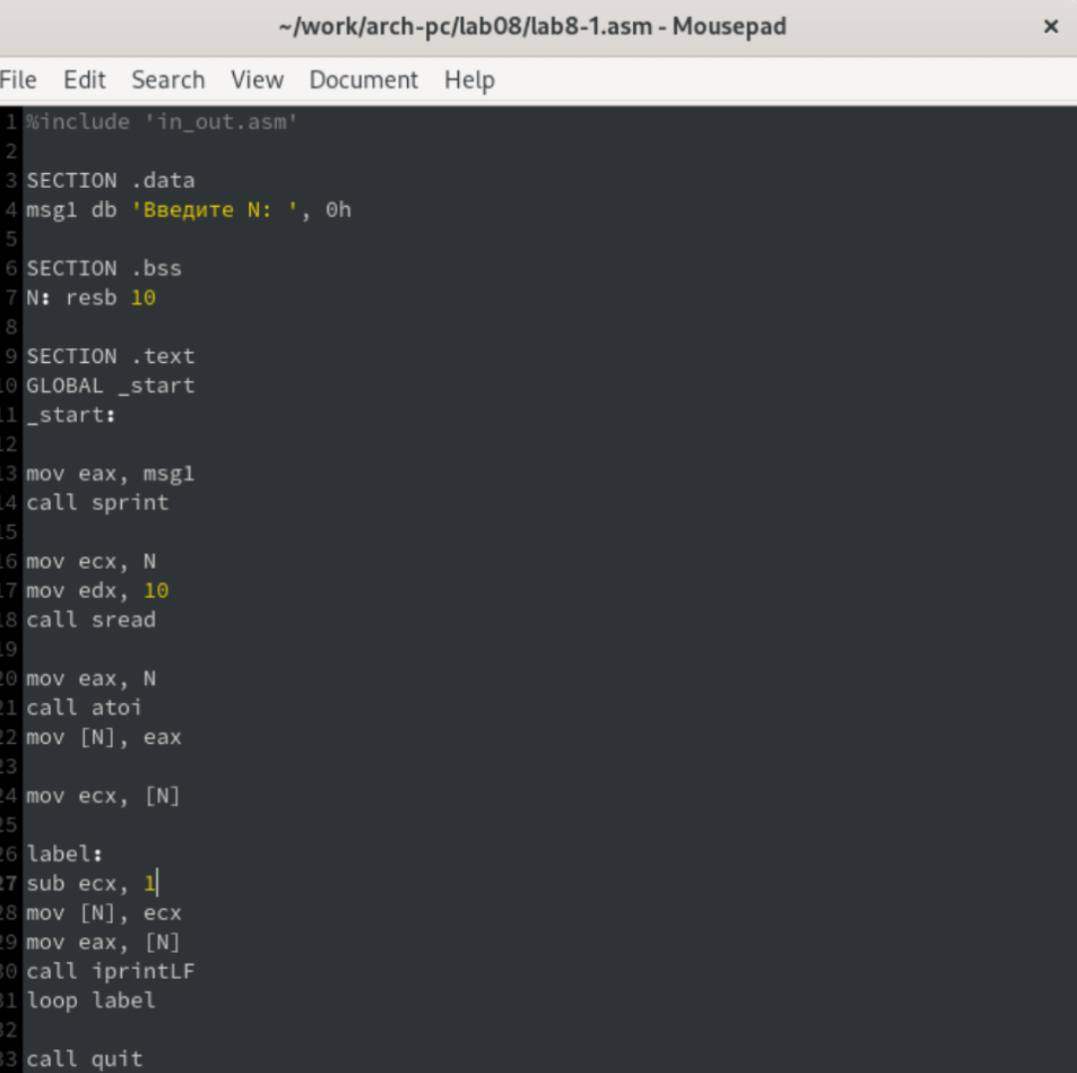
Копирование программы из листинга

Запускаю программу, она показывает работу циклов в NASM .



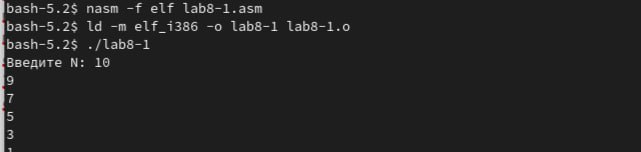
Запуск программы

Заменяю программу изначальную так, что в теле цикла я изменяю значение регистра ecx.



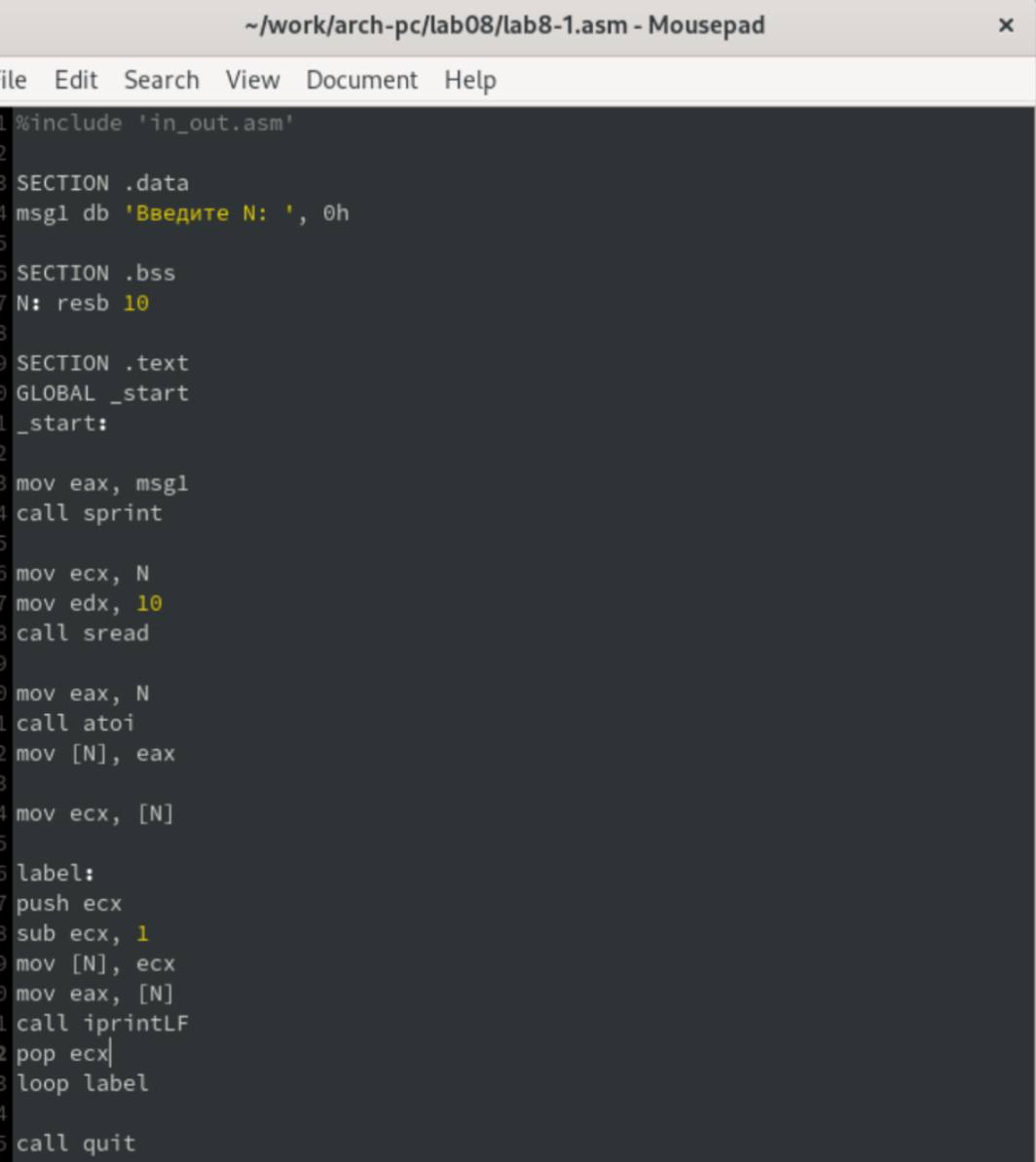
Изменение программы

Из-за того, что теперь регистр ecx на каждой итерации уменьшается на 2 значения, количество итераций уменьшается вдвое.



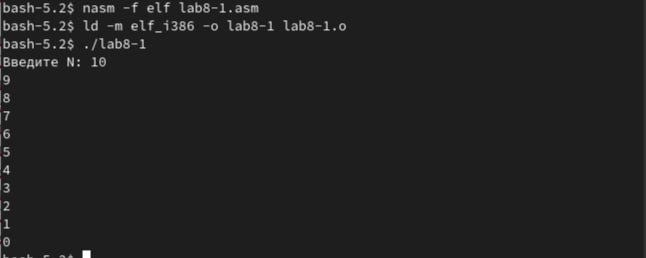
Запуск измененной программы

Добавляю команды push и pop в программу.



Добавление push и pop в цикл программы

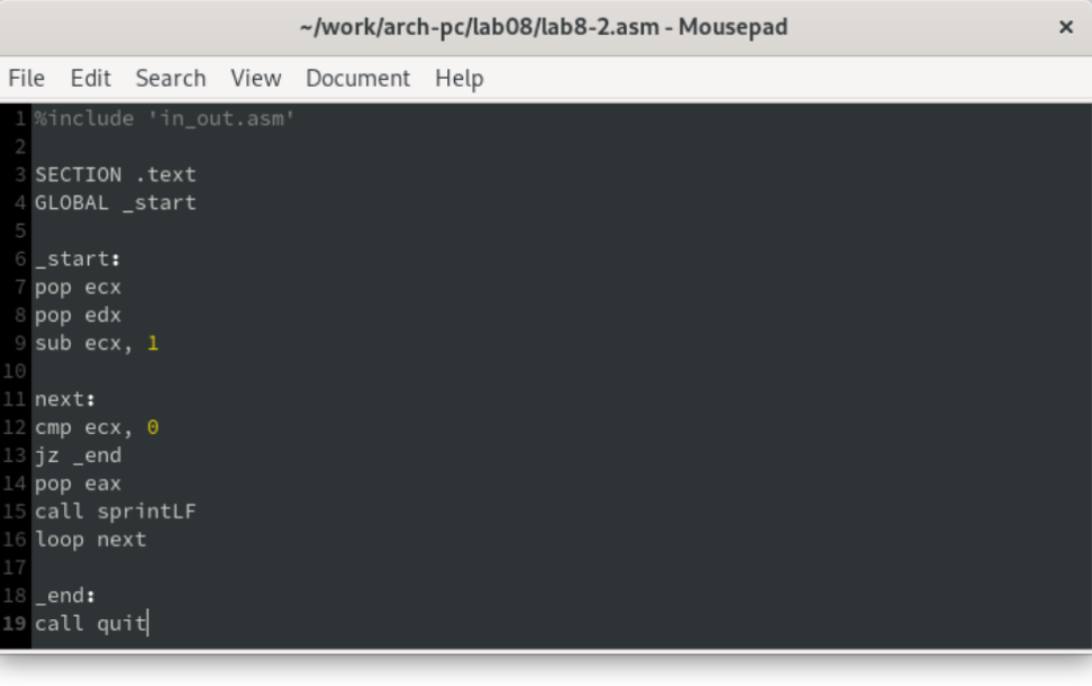
Теперь количество итераций совпадает введенному N, но произошло смещение выводимых чисел на -1 .



Запуск измененной программы

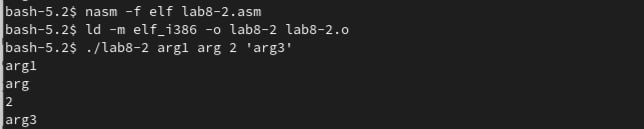
## 4.2 Обработка аргументов командной строки

Создаю новый файл для программы и копирую в него код из следующего листинга



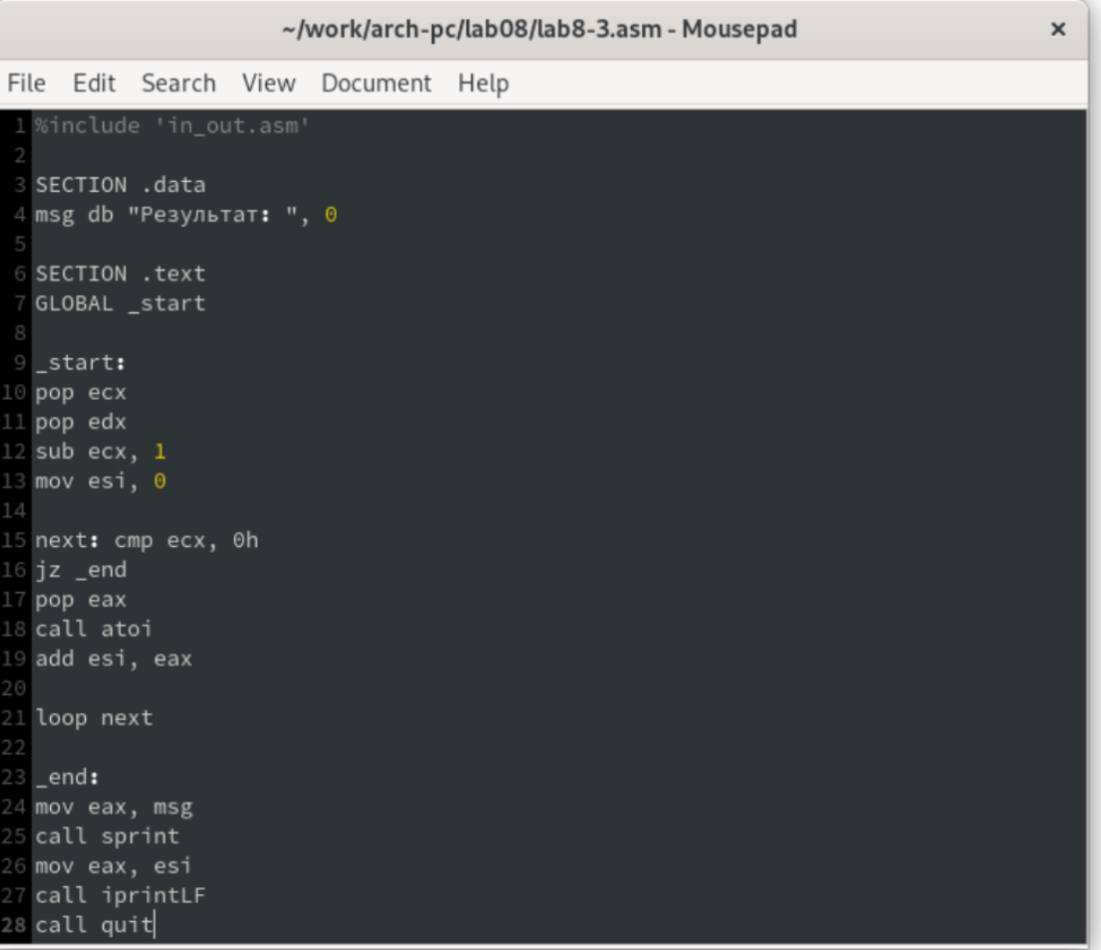
Копирование программы из листинга

Компилирую программу и запускаю, указав аргументы. Программой было обратоно то же количество аргументов, что и было введено.



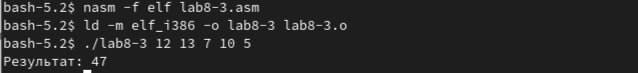
Запуск второй программы

Создаю новый файл для программы и копирую в него код из третьего листинга.



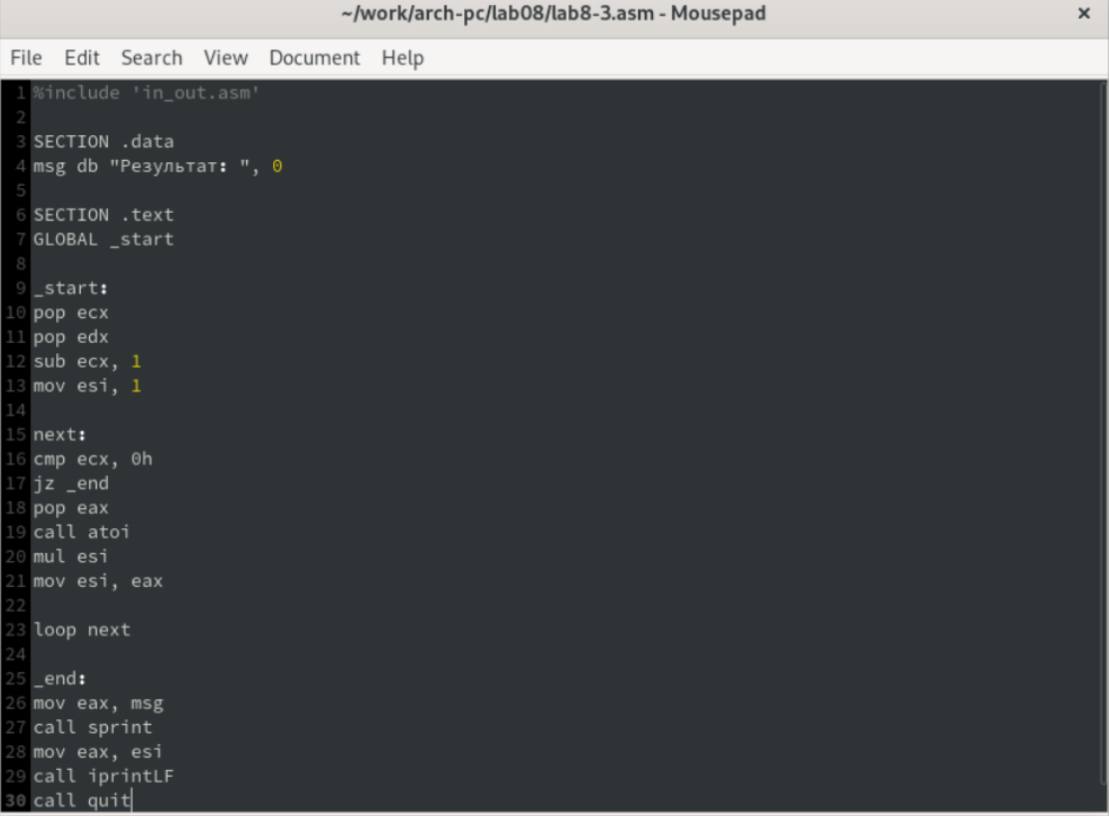
Копирование программы из третьего листинга

Компилирую программу и запускаю, указав в качестве аргументов некоторые числа, программа их складывает.



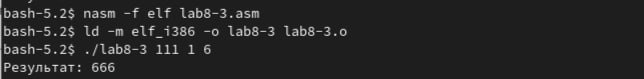
Запуск третьей программы

Изменяю поведение программы так, чтобы указанные аргументы она умножала, а не складывала.



Изменение третьей программы

Программа действительно теперь умножает данные на вход числа.



Запуск измененной третьей программы

## 4.3 Задание для самостоятельной работы

Пишу программму, которая будет находить сумма значений для функции f(x) = 10x-5, которая совпадает с моим девытым варинтом.

Код программы:

%include ‘in\_out.asm’

SECTION .data msg\_func db “Функция: f(x) = 10x - 5”, 0 msg\_result db “Результат:”, 0

SECTION .text GLOBAL \_start

\_start: mov eax, msg\_func call sprintLF

pop ecx pop edx sub ecx, 1 mov esi, 0

next: cmp ecx, 0h jz \_end pop eax call atoi

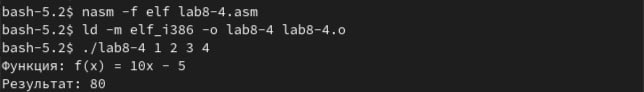
mov ebx, 10 mul ebx sub eax, 5

add esi, eax

loop next

\_end: mov eax, msg\_result call sprint mov eax, esi call iprintLF call quit

Проверяю работу программы, указав в качестве аргумента несколько чисел.



Запуск программы для самостоятельной работы

# 5 Выводы

В результате выполнения данной лабораторной работы я приобрел навыки написания программ с использованием циклов а также научился обрабатывать аргументы командной строки.

# 6 Список литературы

1. [Курс на ТУИС](https://esystem.rudn.ru/course/view.php?id=112)
2. [Лабораторная работа №8](https://esystem.rudn.ru/pluginfile.php/2089095/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%968.%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D1%86%D0%B8%D0%BA%D0%BB%D0%B0.%20%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0%20%D0%B0%D1%80%D0%B3%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%BE%D0%B2%20%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%BE%D0%B9%20%D1%81%D1%82%D1%80%D0%BE%D0%BA%D0%B8.pdf)
3. [Программирование на языке ассемблера NASM Столяров А. В.](https://esystem.rudn.ru/pluginfile.php/2088953/mod_resource/content/2/%D0%A1%D1%82%D0%BE%D0%BB%D1%8F%D1%80%D0%BE%D0%B2%20%D0%90.%20%D0%92.%20-%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%BD%D0%B0%20%D1%8F%D0%B7%D1%8B%D0%BA%D0%B5%20%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0%20NASM%20%D0%B4%D0%BB%D1%8F%20%D0%9E%D0%A1%20Unix.pdf)