Отчет по лабораторной работе №8

Дисциплина: архитектура компьютера

Ермакова Анастасия Алексеевна

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

Цель данной лабораторной работы - приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

# 2 Задание

1. Реализация циклов в NASM
2. Обработка аргументов командной строки
3. 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).

# 4 Выполнение лабораторной работы

## 4.1 Реализация циклов в NASM

Создаю каталог для программ лабораторной работы №8, перехожу в него и создаю файл lab8-1.asm. (рис. 1).

Рис. 1: Создание каталога и файла

Рис. 1: Создание каталога и файла

Ввожу в файл текст программы из листинга 8.1. (рис. 2).

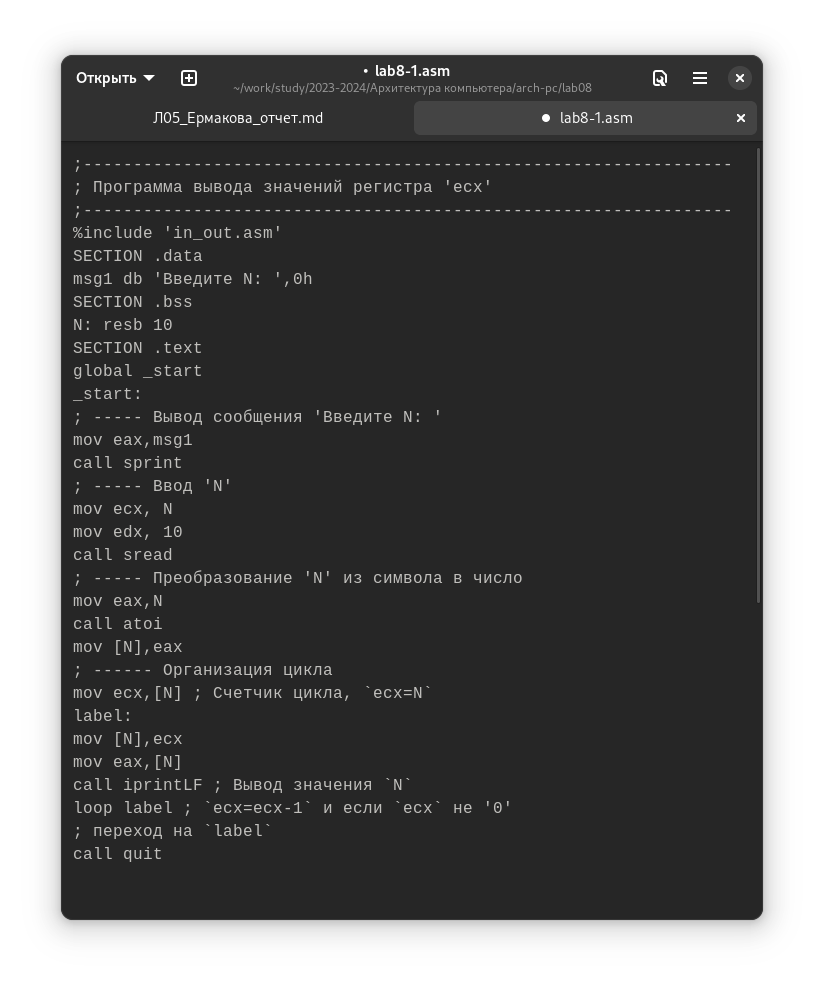


Рис. 2: Текст программы файла lsb8-1.asm

Создаю исполняемый файл и запускаю его. (рис. 3). Результат программы - работа циклов в NASM.

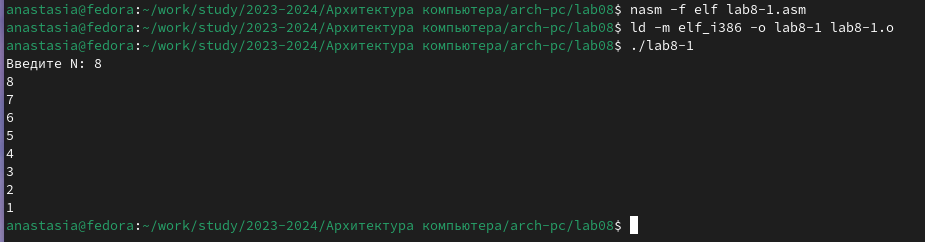


Рис. 3: Результат работы программы

Изменяю текст программы, добавив изменение в значении регистра ecx. (рис. 4).

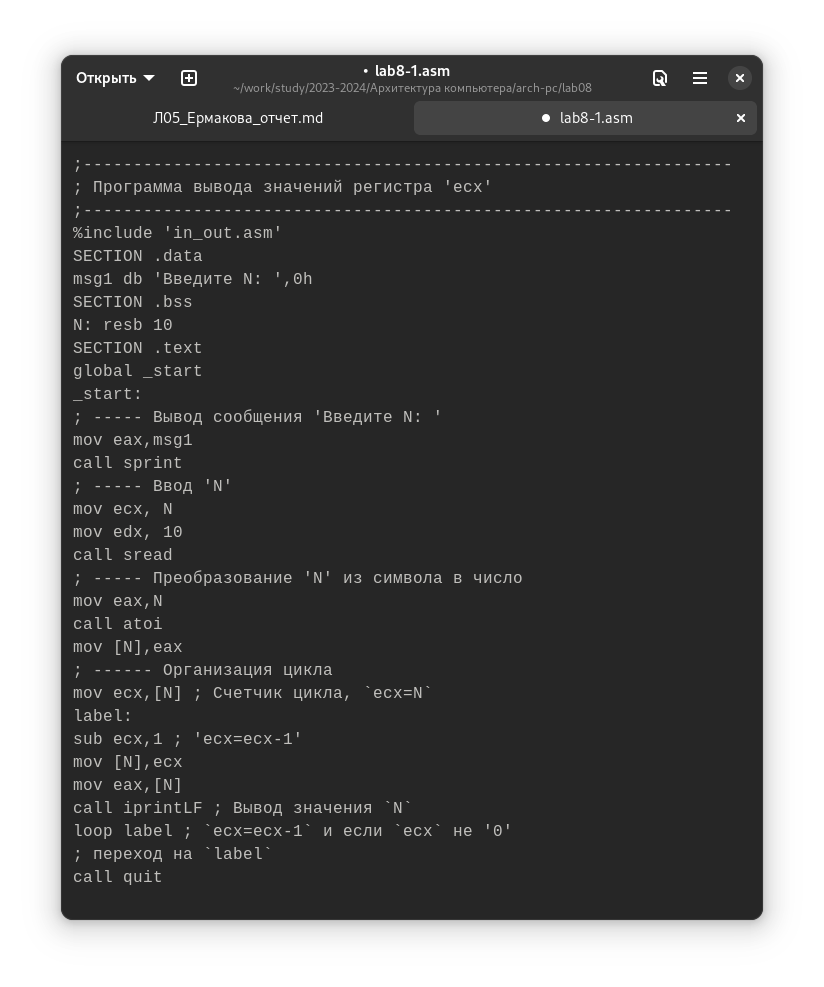


Рис. 4: Измененный текст файла

Создаю исполняемый файл и запускаю его. (рис. 5). Регистр ecx каждый раз уменьшается на 2, количество итераций уменьшается вдвое.

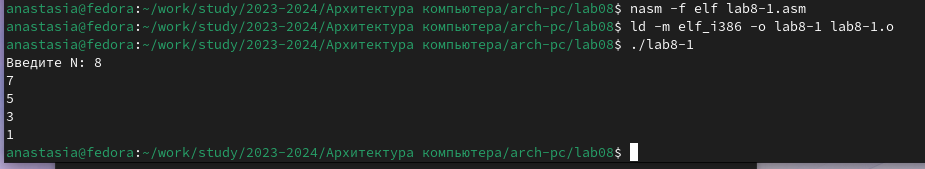


Рис. 5: Результат работы программы

Вношу изменения в текст программы, добавив команды push и pop для сохранения значения счетчика цикла loop. (рис. 6).

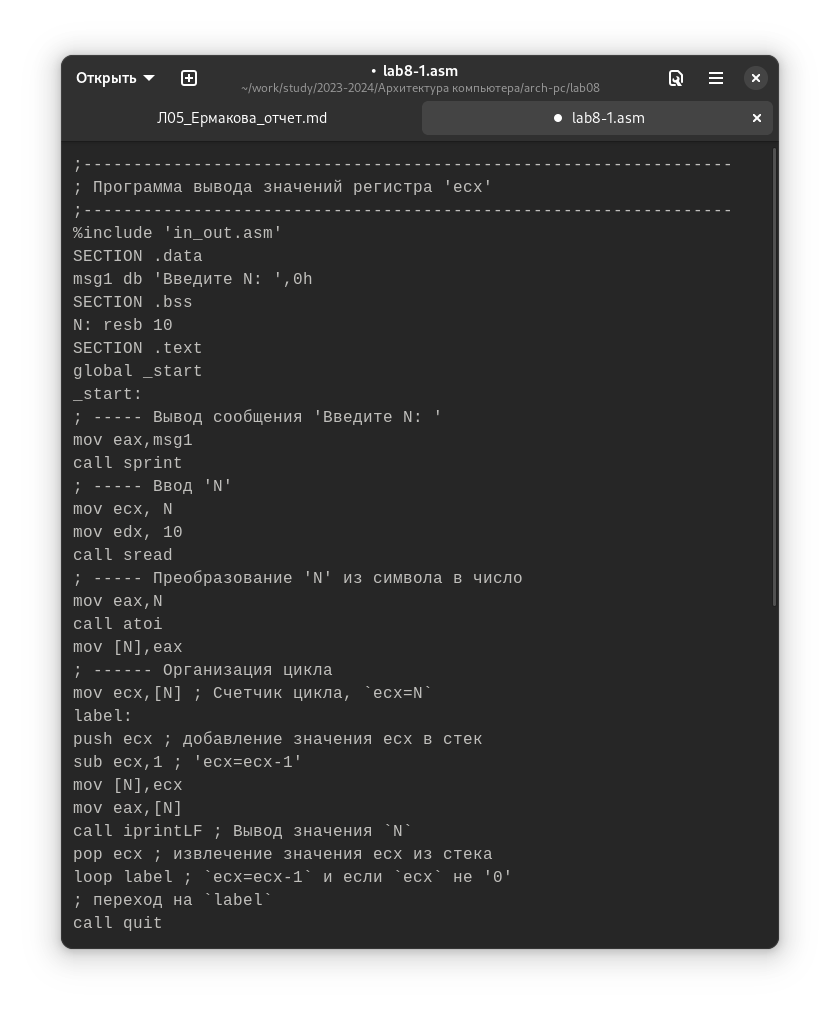


Рис. 6: Измененный текст программы

Создаю исполняемый файл и запускаю его. (рис. 7). Теперь количество подходов цикла совпадает с введенным значением, но из-за смещения на 1 вывод начался со значения N-1 и заканчивается нулем.

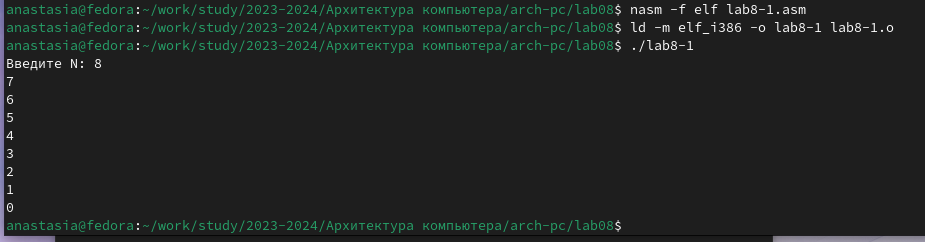


Рис. 7: Результат работы программы

## 4.2 Обработка аргументов командной строки

Создаю файл lab8-2.asm в текущем каталоге. (рис. 8).

Рис. 8: Создание файла lab8-2.asm

Рис. 8: Создание файла lab8-2.asm

Ввожу текст программы из листинга 8.2. (рис. 9).

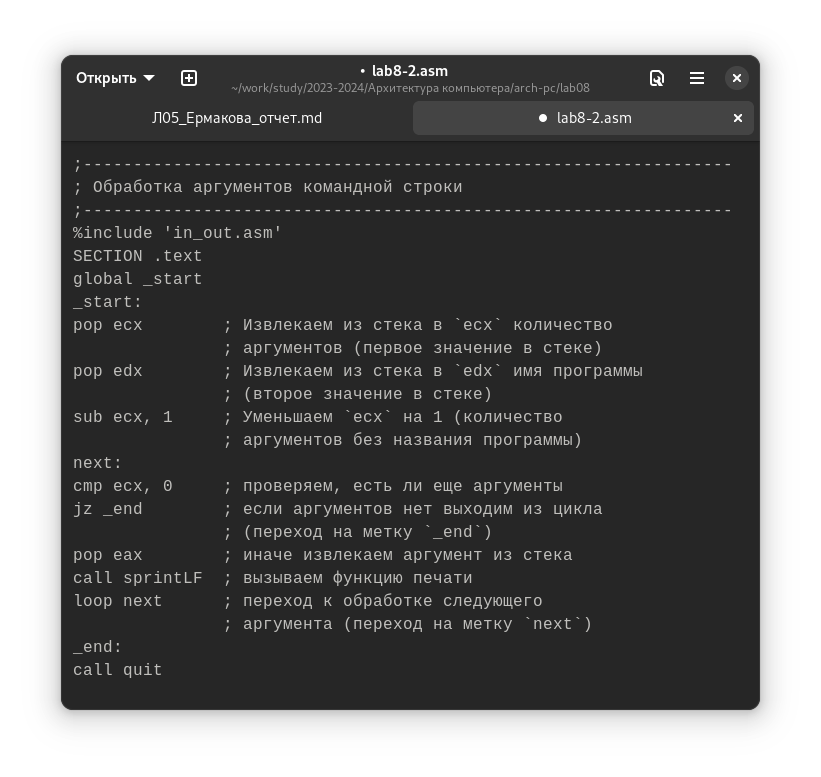


Рис. 9: Текст программы файла lab8-2.asm

Создаю исполняемый файл и запускаю его, указав аргументы. (рис. 10). Все три аргумента были обработаны программой

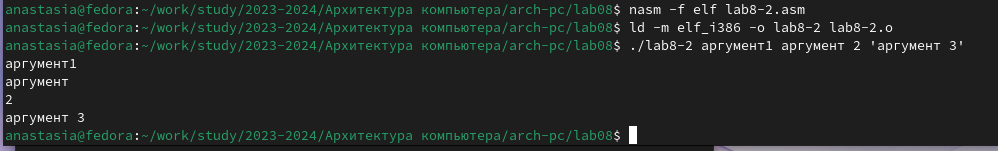


Рис. 10: Результат работы программы

Создаю файл lab8-3.asm и ввожу текст программы из листинга 8.3. (рис. 11).

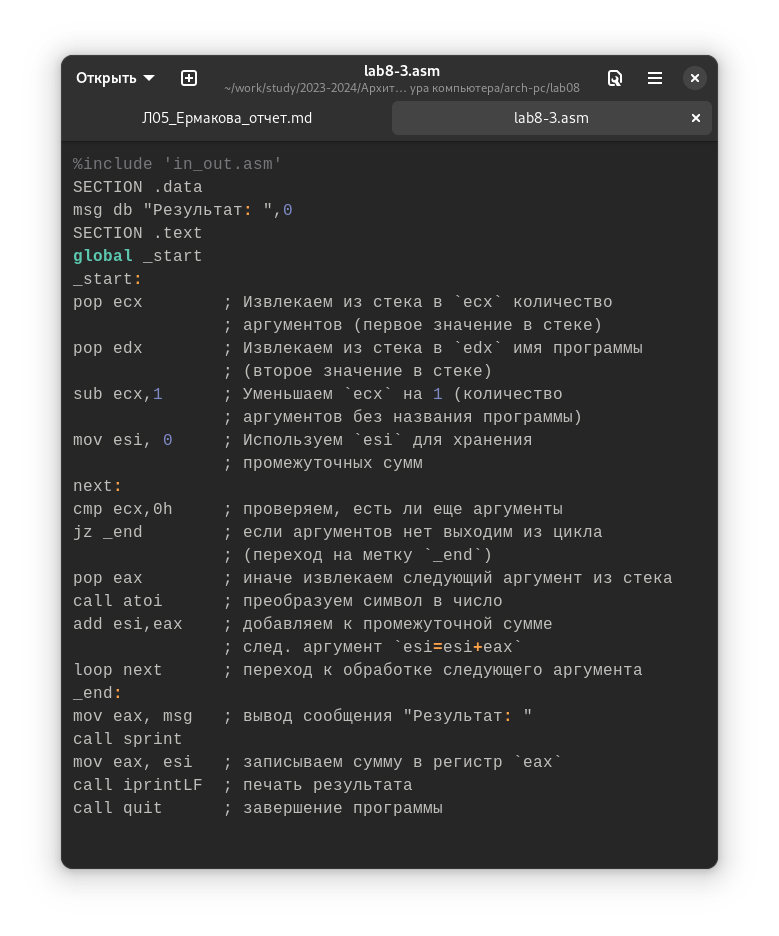


Рис. 11: Текст программы файла lab8-3.asm

Создаю исполняемый файл и запускаю его, указав аргументы. (рис. 12). Программа выводит результат - сумма введенных аргументов.

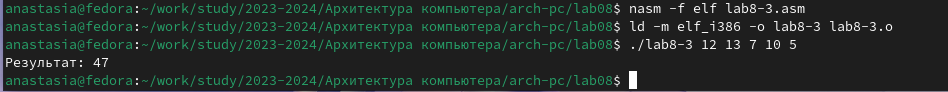


Рис. 12: Результат работы программы

Изменяю текст программы так, чтобы она считала произведение введенных аргументов. (рис. 13).

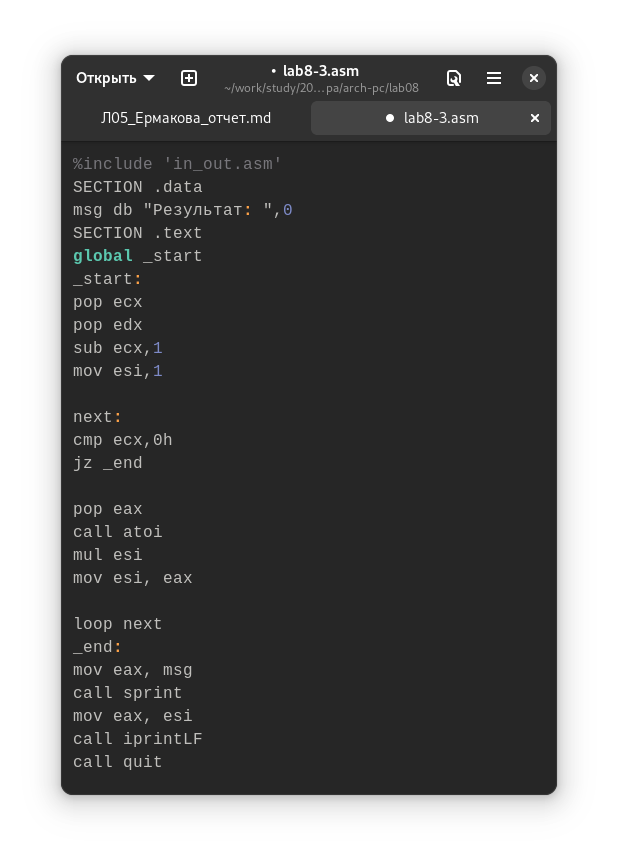


Рис. 13: Измененный текст программы

Создаю исполняемый файл и запускаю его. (рис. 14). Программа работает исправно.

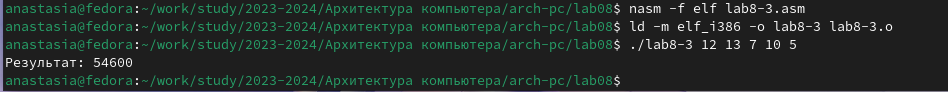


Рис. 14: Результат работы программы

## 4.3 Выполнение заданий для самостоятельной работы

Напишу программу, которая находит сумму значений функции f(x) для введенных значений х. Из таблицы 8.1 выбираю свой вариант (19), полученный в ходе выполнения одной из предыдущих лабораторных работ. Создаю файл lab8-4.asm для написания кода. (рис. 15).

Рис. 15: Создание файла lab8-4.asm

Рис. 15: Создание файла lab8-4.asm

Ввожу текст кода. (рис. 16).

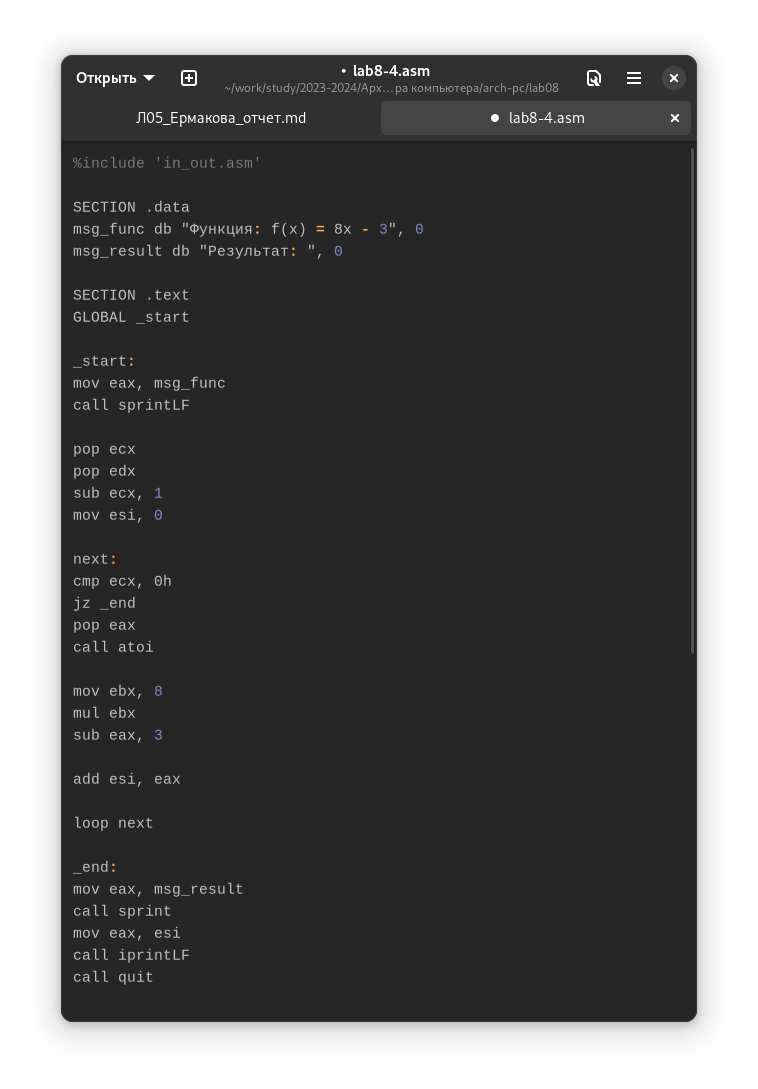


Рис. 16: Текст программы файла

Код программы:

%include 'in\_out.asm'  
  
SECTION .data  
msg\_func db "Функция: f(x) = 8x - 3", 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, 8  
mul ebx  
sub eax, 3  
  
add esi, eax  
  
loop next  
  
\_end:   
mov eax, msg\_result  
call sprint  
mov eax, esi  
call iprintLF  
call quit

Создаю исполняемый файл и запускаю его. (рис. 17).

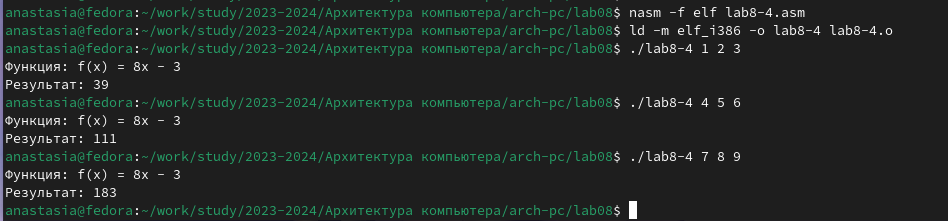


Рис. 17: Результат работы файла

Проверяю работу программы трижды, введя разные аргументы. Программа работает исправно, это легко проверить, посчитав самому.

# 5 Выводы

В ходе выполнения данной лабораторной работы я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.

# 6 Список литературы

1. [Архитектура ЭВМ](https://esystem.rudn.ru/mod/resource/view.php?id=1030556)