Отчёт по лабораторной работе 8

Архитектура компьютера

Довран Илиев

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

1. Создал папку для файлов лабораторной № 8, перешел в нее и инициировал создание файла с именем lab8-1.asm.

2. Внес программный код в файл lab8-1.asm, соответствующий листингу 8.1. Скомпилировал его в исполняемый файл и осуществил проверку функционирования.

Open | lab8-1.asm ~/work/arch-pc/lab08 | Save

```
1 %include 'in_out.asm'
2 SECTION .data
3 msg1 db 'Введите N: ',0h
4 SECTION .bss
5 N: resb 10
6 SECTION .text
7 global _start
8 _start:
9 ; ----- Вывод сообщения 'Введите N: '
10 mov eax,msg1
11 call sprint
12 ; ----- Ввод 'N'
13 mov ecx, N
14 mov edx, 10
15 call sread
16 ; ----- Преобразование 'N' из символа в число
17 mov eax,N
18 call atoi
19 mov [N],eax
20 ; ------ Организация цикла
21 mov ecx,[N] ; Счетчик цикла, `ecx=N`
22 label:
23 mov [N],ecx
24 mov eax,[N]
25 call iprintLF ; Вывод значения `N`
26 loop label ; `ecx=ecx-1` и если `ecx` не '0'
27 ; переход на `label`
28 call quit
```

Рис. 2.1: Программа lab8-1.asm

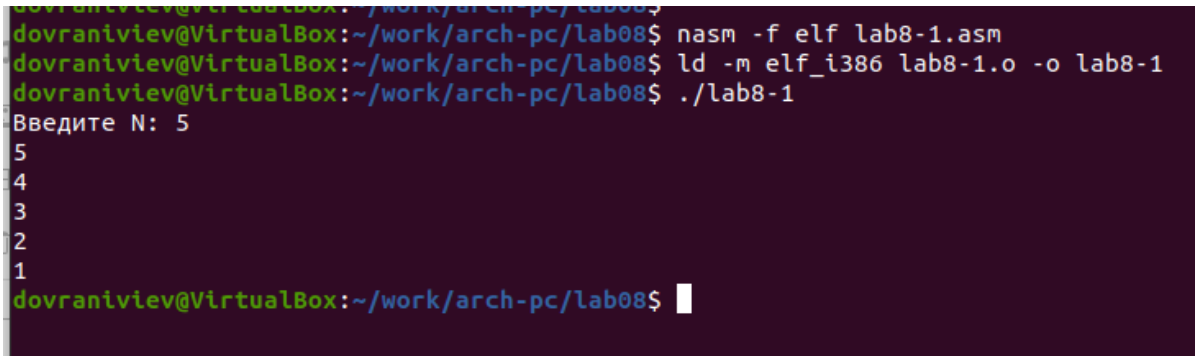

Рис. 2.2: Запуск программы lab8-1.asm

3. Приведенный пример иллюстрирует, что манипулирование регистром ecx внутри цикла loop может привести к ошибкам в работе программы. Модифицировал код, включив изменение значения регистра ecx в процессе выполнения цикла:

При нечетном N программа запускается в бесконечный цикл, а при четном N выводит исключительно нечетные числа.

Open | lab8-1.asm ~/work/arch-pc/lab08 | Save

```
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ----- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ----- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ------ Организация цикла
mov ecx,[N] ; Счетчик цикла, `ecx=N`
label:
sub ecx,1 ; `ecx=ecx-1`
mov [N],ecx
mov eax,[N]
call iprintLF
loop label
; переход на `label`
call quit
```

Рис. 2.3: Программа lab8-1.asm

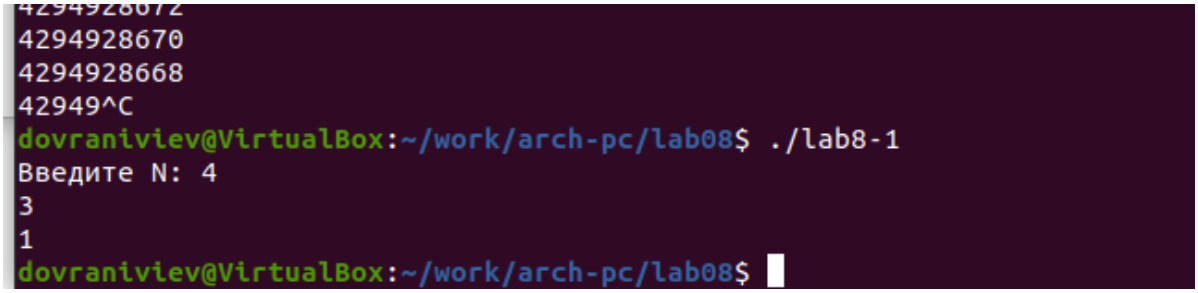

Рис. 2.4: Запуск программы lab8-1.asm

4. Чтобы корректно использовать регистр ecx внутри цикла, применим стек. Внес изменения в код программы, добавив команды push и pop для сохранения и восстановления значения счетчика цикла loop. Скомпилировал исполняемый файл и провел проверку его функционирования.

Программа отображает числа от N-1 до 0, количество итераций соответствует введенному числу N.

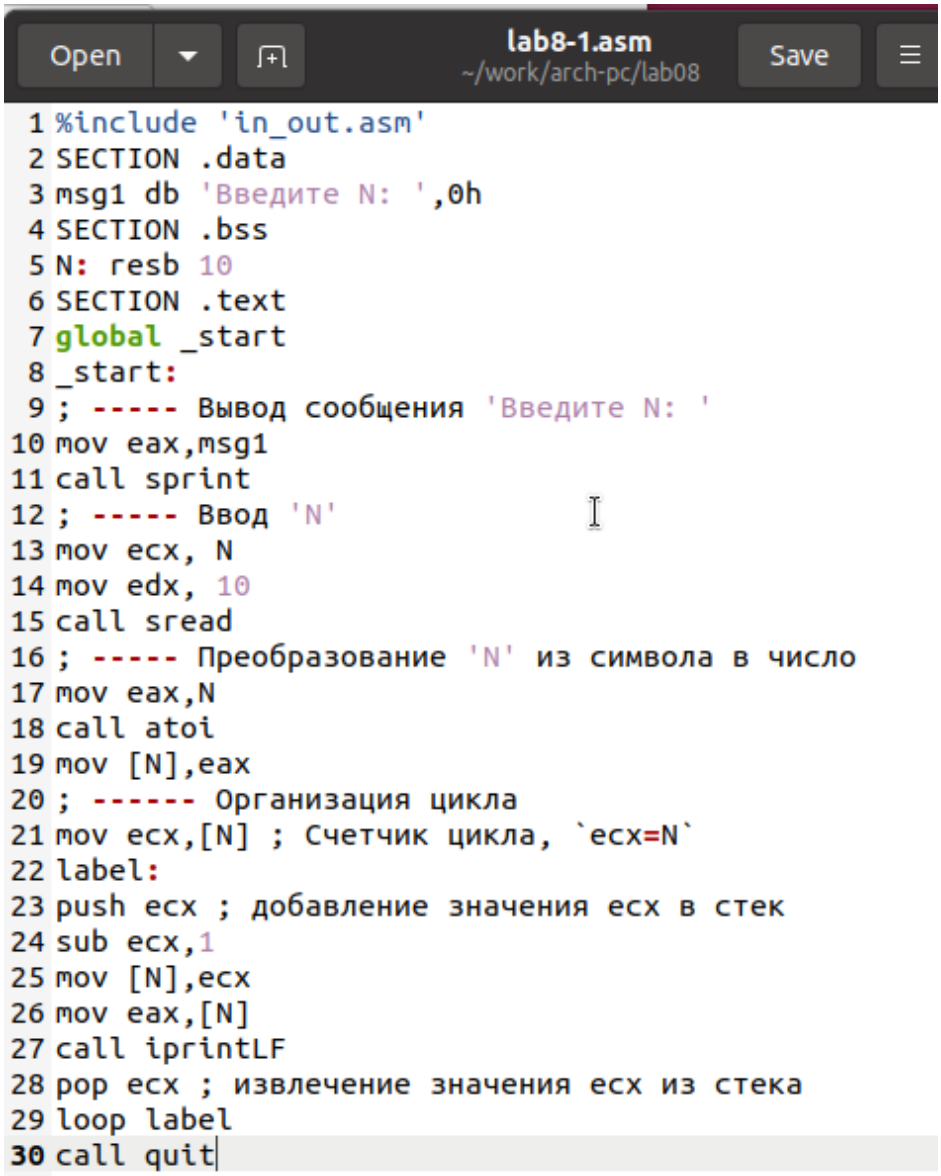

Рис. 2.5: Программа lab8-1.asm

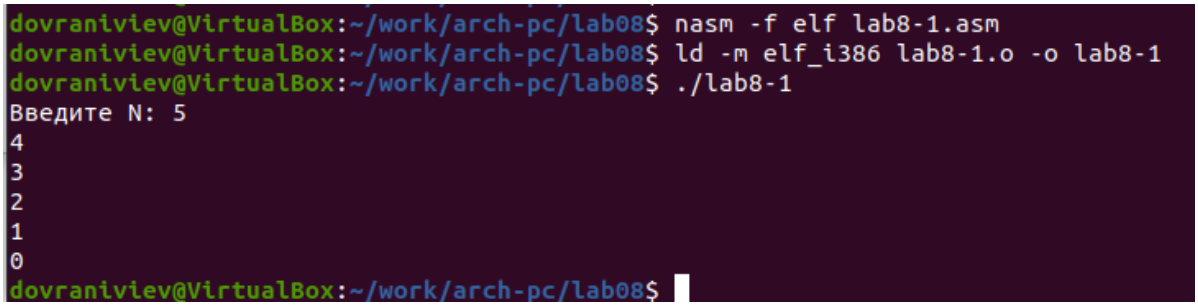

Рис. 2.6: Запуск программы lab8-1.asm

5. Сформировал файл lab8-2.asm в директории ~/work/arch-pc/lab08 и заполнил его кодом из листинга 8.2. Скомпилировал в исполняемый файл и выполнил его с аргументами.

Программа обработала пять аргументов.

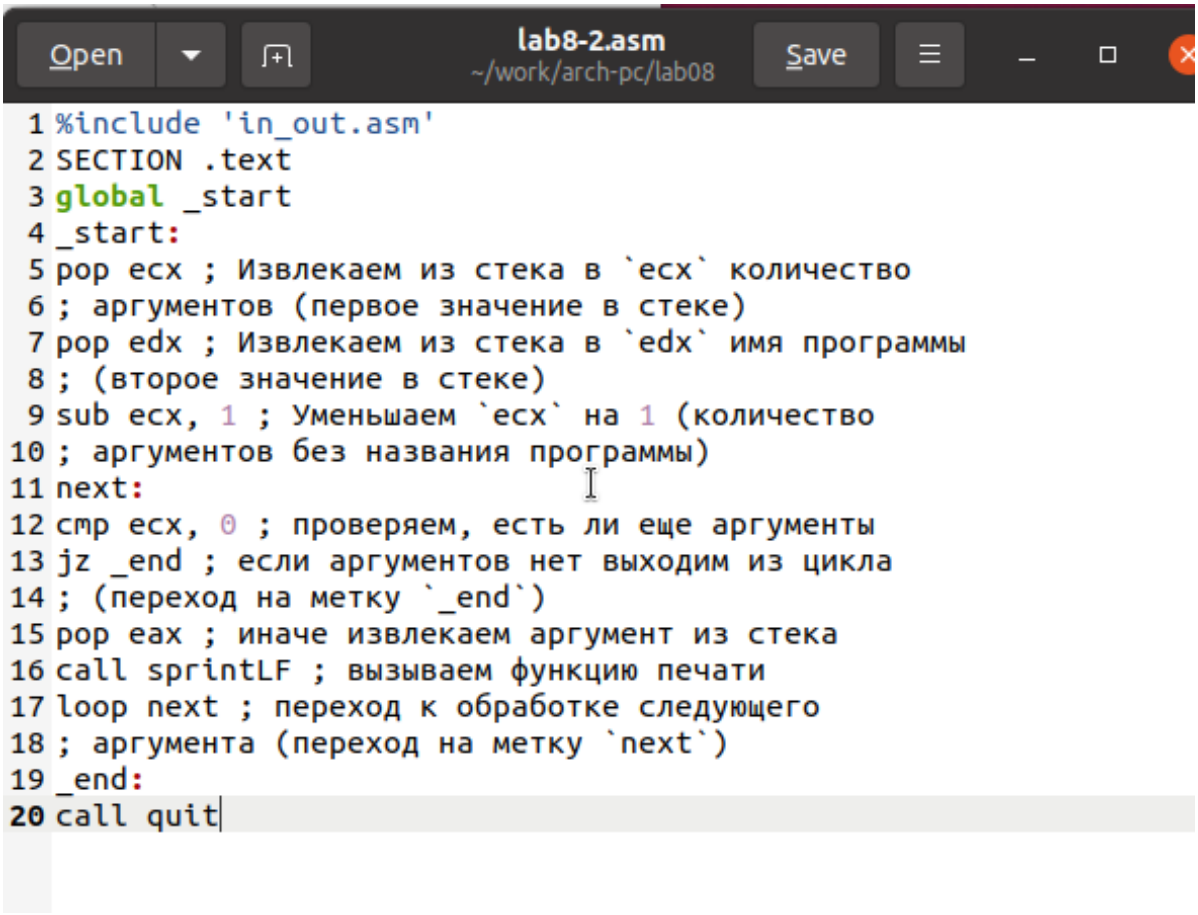

Рис. 2.7: Программа lab8-2.asm

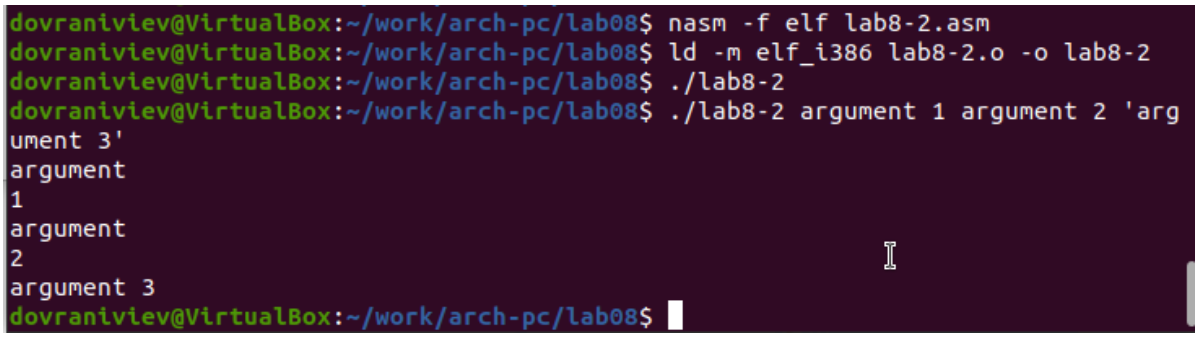

Рис. 2.8: Запуск программы lab8-2.asm

6. Рассмотрим еще один пример программы которая выводит сумму чисел, которые передаются в программу как аргументы.

Open lab8-3.asm ~/work/arch-pc/lab08 Save

```
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx ; Извлекаем из стека в `ecx` количество
; аргументов (первое значение в стеке)
pop edx ; Извлекаем из стека в `edx` имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем `ecx` на 1 (количество
; аргументов без названия программы)
mov esi, 0 ; Используем `esi` для хранения
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку `_end`)
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
add esi,eax ; добавляем к промежуточной сумме
; след. аргумент `esi=esi+eax`
loop next ; переход к обработке следующего аргумента
_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint
mov eax, esi ; записываем сумму в регистр `eax`
call iprintLF ; печать результата
call quit ; завершение программы
```

Рис. 2.9: Программа lab8-3.asm

```
dovraniviev@VirtualBox:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
dovraniviev@VirtualBox:~/work/arch-pc/lab08$ ld -m elf_i386 lab8-3.o -o lab8-3
dovraniviev@VirtualBox:~/work/arch-pc/lab08$ ./lab8-3
Результат: 0
dovraniviev@VirtualBox:~/work/arch-pc/lab08$ ./lab8-3 3 6 9
Результат: 18
dovraniviev@VirtualBox:~/work/arch-pc/lab08$
```

Рис. 2.10: Запуск программы lab8-3.asm

7. Переписал код программы из листинга 8.3 так, чтобы она вычисляла произведение переданных через командную строку аргументов.

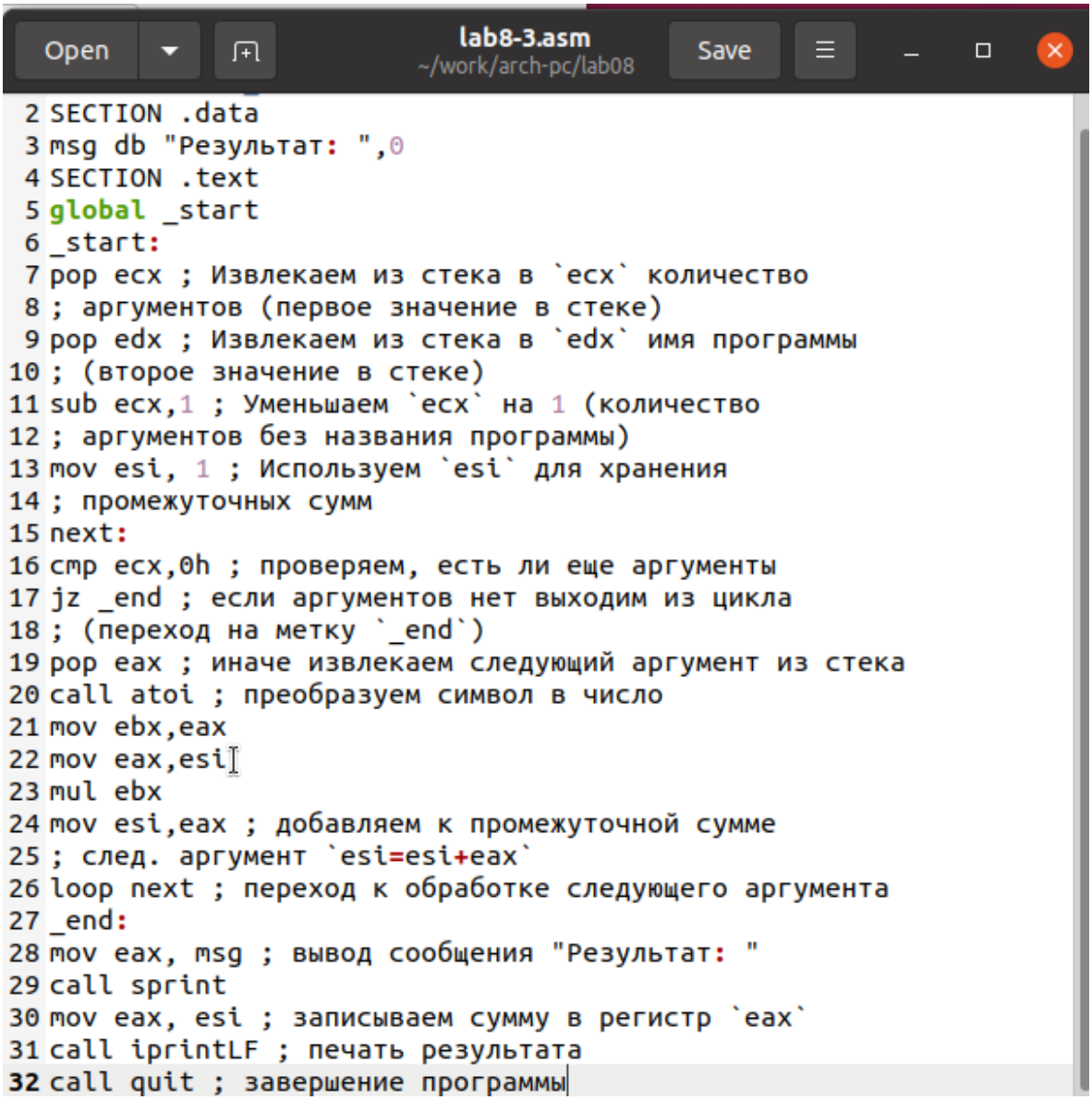

Рис. 2.11: Программа lab8-3.asm

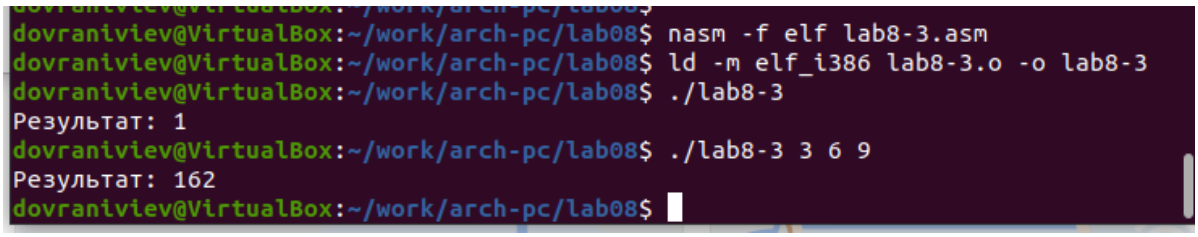

Рис. 2.12: Запуск программы lab8-3.asm

8. 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 x передаются как аргументы. Вид функции f(x) выбрать из таблицы 8.1 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 6. Создайте исполняемый файл и проверьте его работу на нескольких наборах x.

для варивнта 5 $f(x) = 4x + 3$

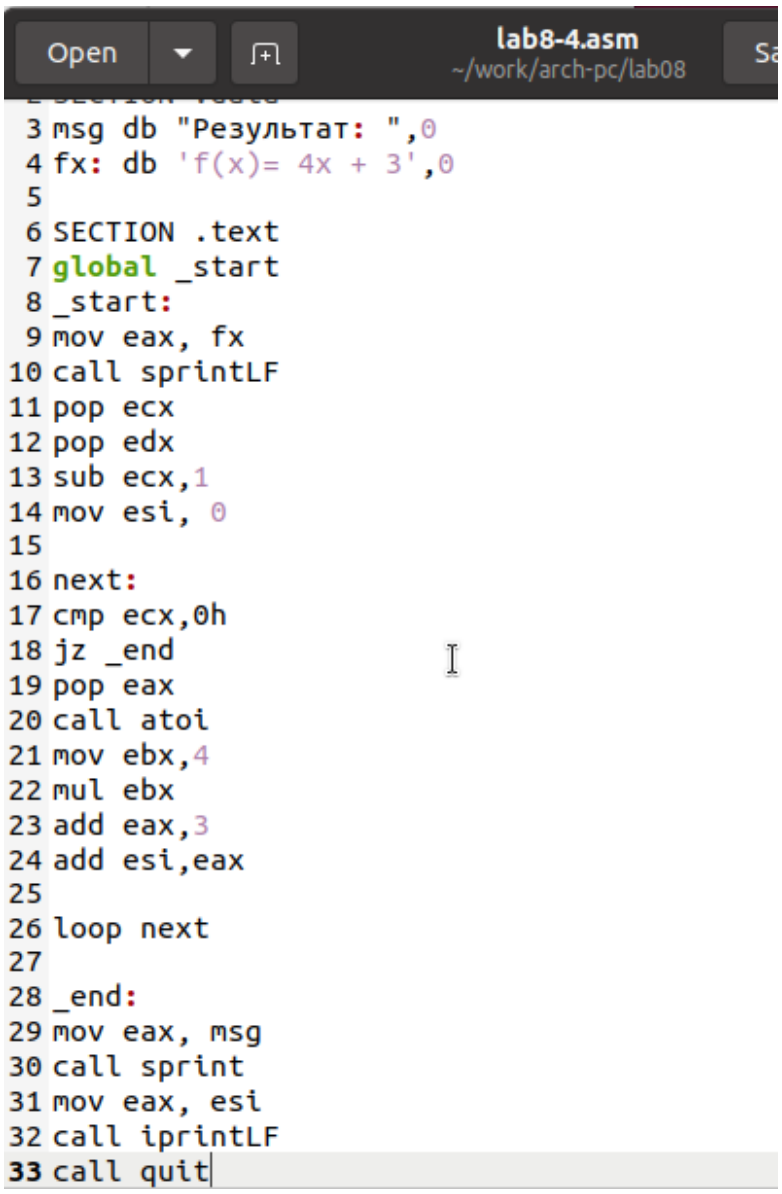

Рис. 2.13: Программа lab8-4.asm

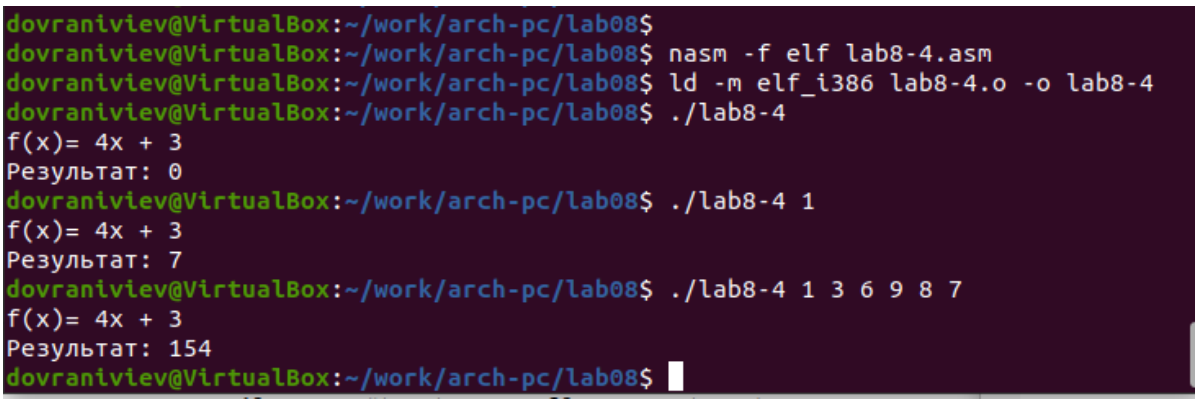

Рис. 2.14: Запуск программы lab8-4.asm

3 Выводы

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