Отчёт по лабораторной работе №7

Дисциплина: архитектура компьютера.

Наговицын Арсений Владимирович

Содержание

# 1 Цель работы

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

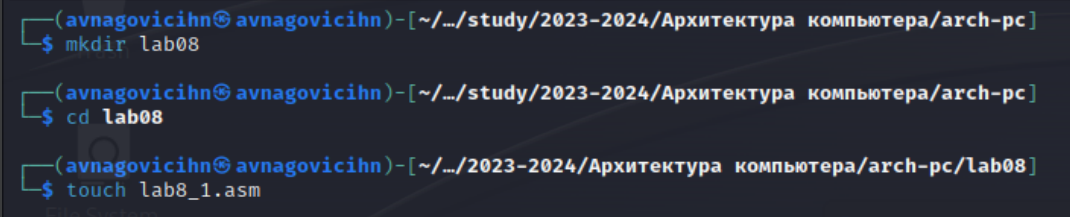
# 2 Задание

1. Реализация циклов в NASM
2. Обработка аргументов командной строки
3. Выполнение заданий для самостоятельной работы

# 3 Выполнение лабораторной работы

## 3.1 Реализация циклов в NASM

Создаю каталог для программ и перехожу в него. Создаю файл (рис. ??).



Создание каталога и файла

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

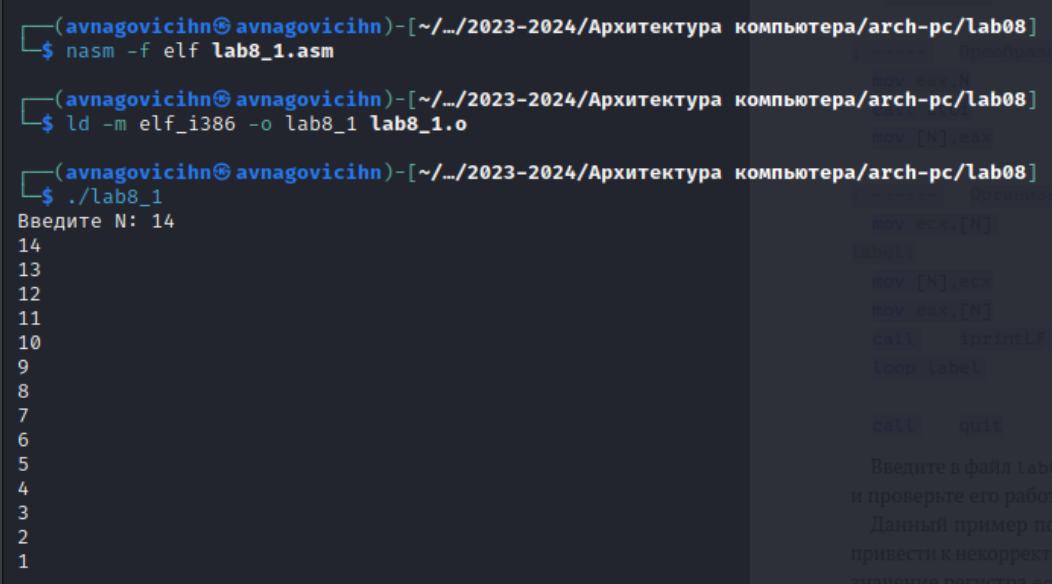


Редактирование файла

**Листинг 8.1. Программа вывода значений регистра ecx**

%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:  
mov [N],ecx  
mov eax,[N]  
call iprintLF ; Вывод значения `N`  
loop label ; `ecx=ecx-1` и если `ecx` не '0'  
; переход на `label`  
call quit

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



Компиляция и запуск файла

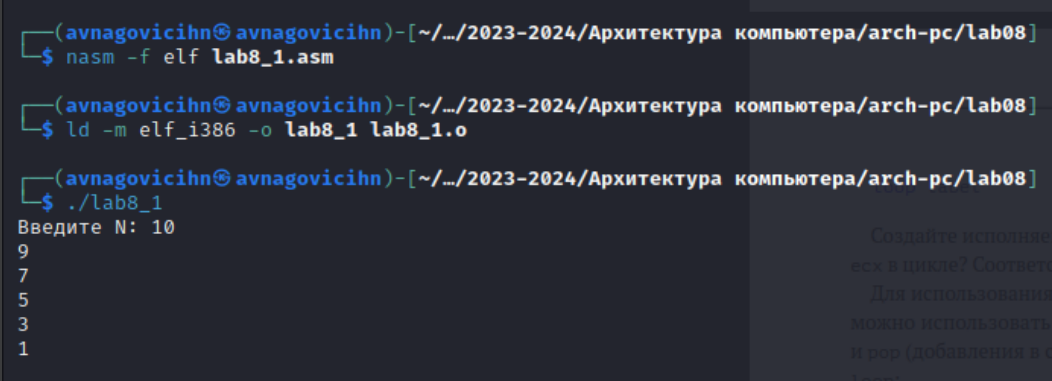
Данная программа выводит все числа от N до 1 включительно. Изменяю текст программы (рис. ??).

label:  
sub ecx,1 ; `ecx=ecx-1`  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
loop label



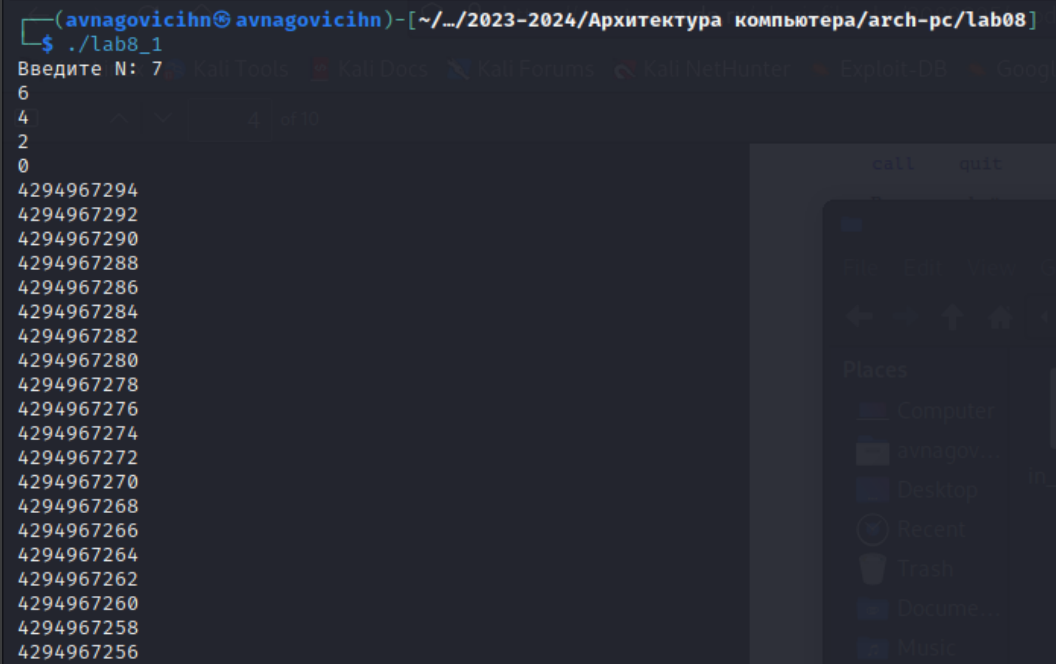
Редактирование файла

Создаю исполняемый файл и запускаю его. Ввожу в него четное значение (рис. ??).



Компиляция и запуск файла

Программа выводит все числа от N-1 до 1 с интервалом в 2 - то есть все нечётные числа. Ввожу нечетное число. Данная программа выводит бесконечную последовательность значений, так как значение регистра ecx переваливает за 0(рис. ??).



Компиляция и запуск файла

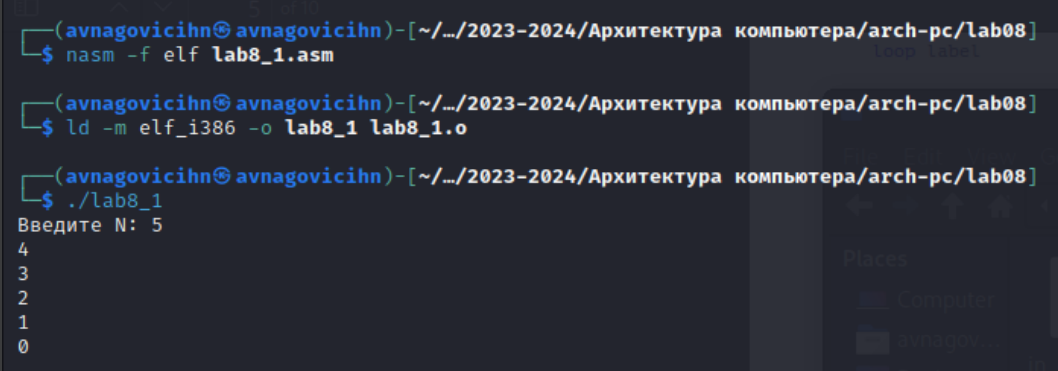
Изменяю текст программы: (рис. ??).

label:  
push ecx ; добавление значения ecx в стек  
sub ecx,1  
mov [N],ecx  
mov eax,[N]  
call iprintLF  
pop ecx ; извлечение значения ecx из стека  
loop labe



Редактирование файла

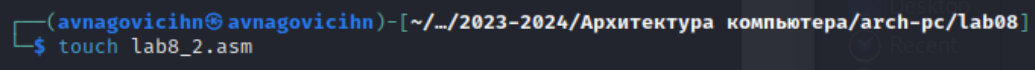
Создаю исполняемый файл и запускаю его. Ввожу в него нечетное значение (рис. ??).



Компиляция и запуск файла

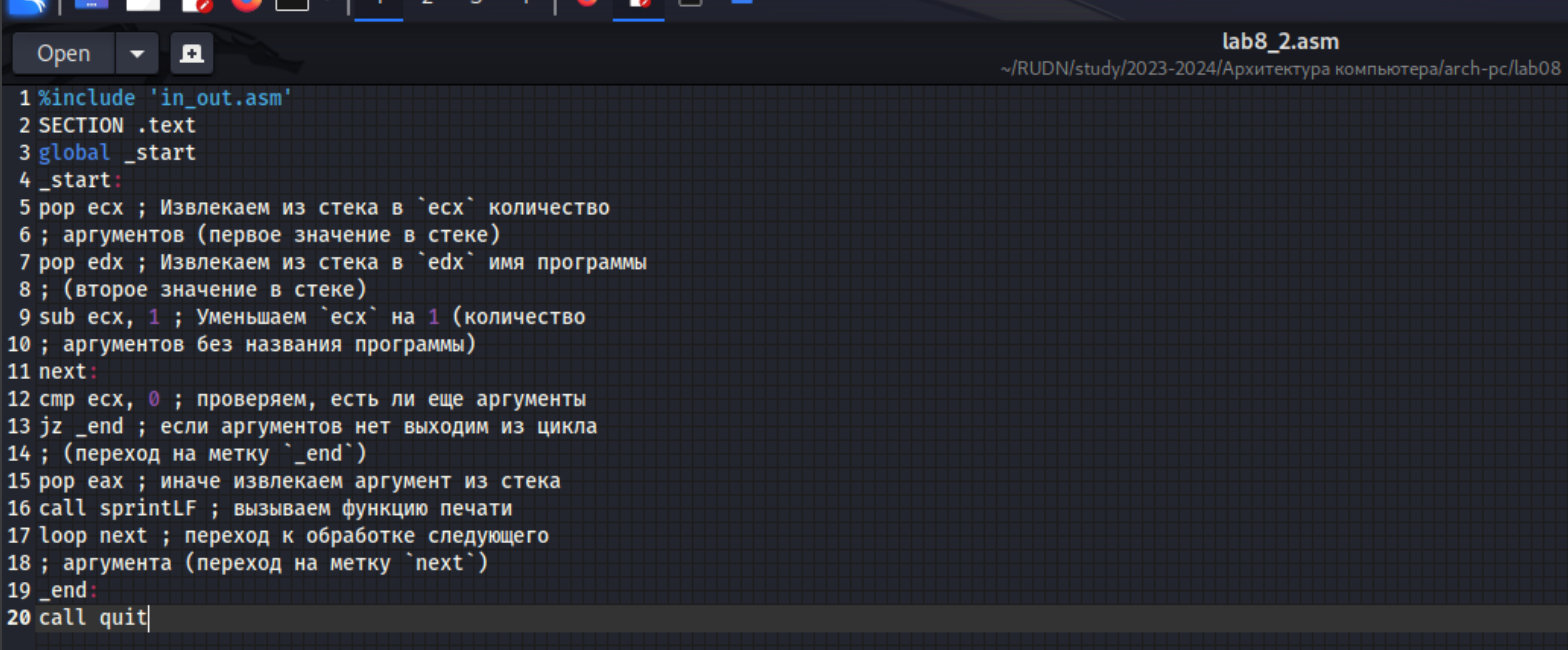
## 3.2 Обработка аргументов командной строки

Создаю новый файл lab8\_2.asm (рис. ??).



Создание файла

Ввожу в файл текст программы из листинга 8.2(рис. ??).

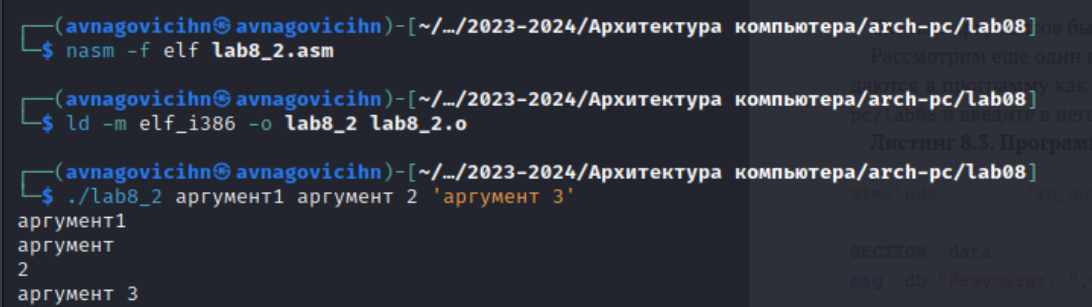


Редактирование файла

**Листинг 8.2. Программа выводящая на экран аргументы командной строки**

%include 'in\_out.asm'  
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)  
next:  
cmp ecx, 0 ; проверяем, есть ли еще аргументы  
jz \_end ; если аргументов нет выходим из цикла  
; (переход на метку `\_end`)  
pop eax ; иначе извлекаем аргумент из стека  
call sprintLF ; вызываем функцию печати  
loop next ; переход к обработке следующего  
; аргумента (переход на метку `next`)  
\_end:  
call quit

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



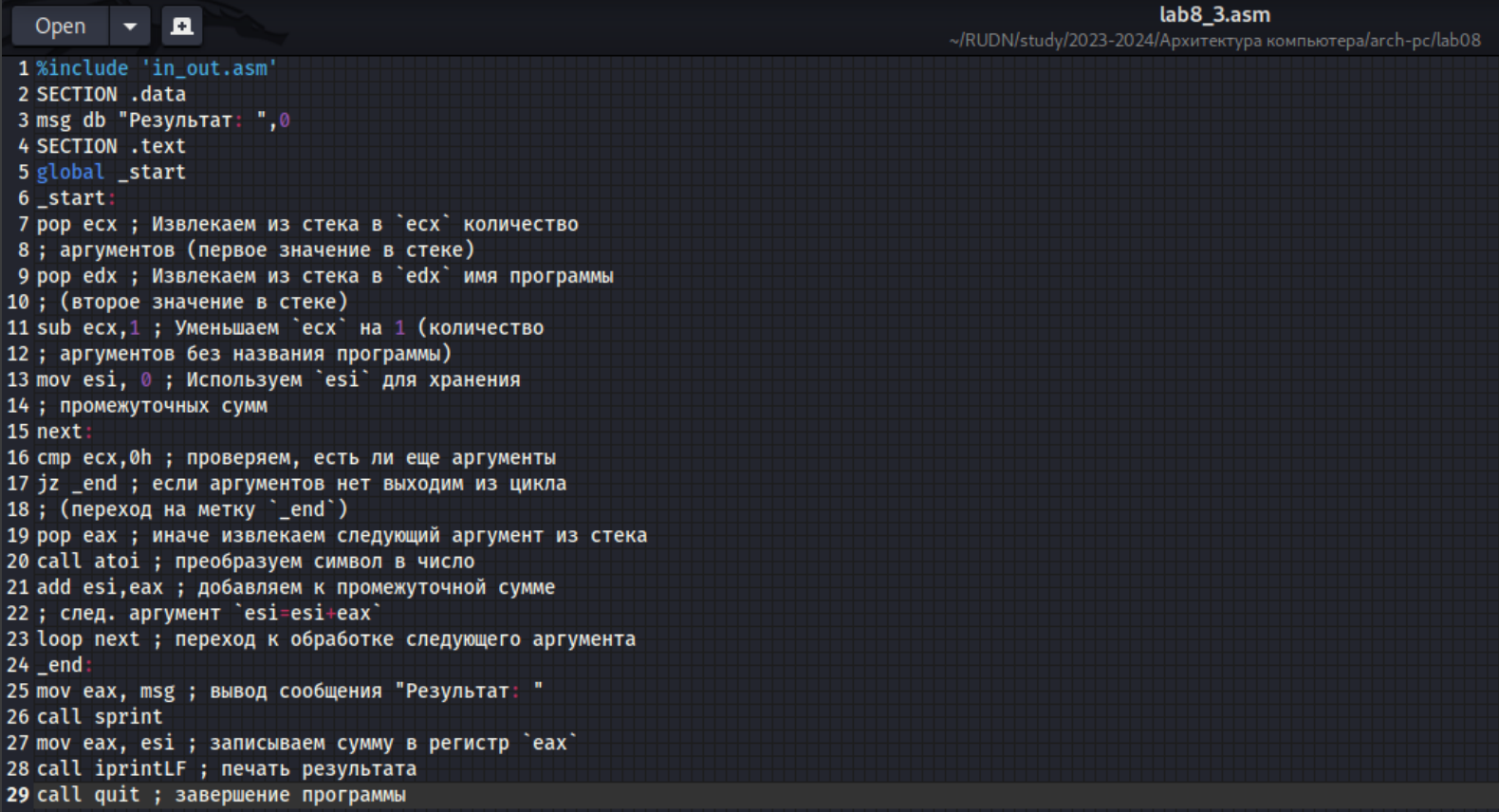
Компиляция и запуск файла

Создаю новый файл lab8\_3.asm (рис. ??).

Создание файла

Создание файла

Ввожу в файл текст программы из листинга 8.3(рис. ??).

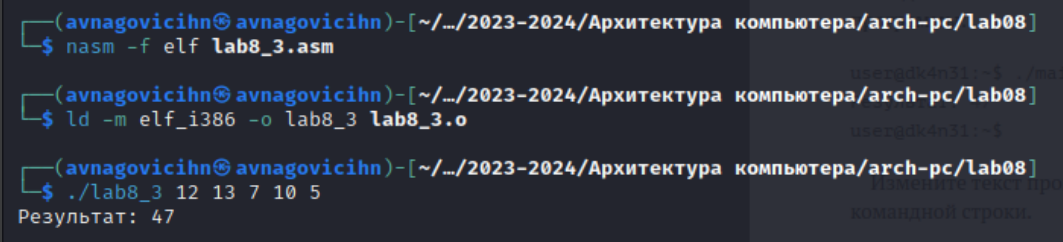


Редактирование файла

**Листинг 8.3. Программа вычисления суммы аргументов командной строки**

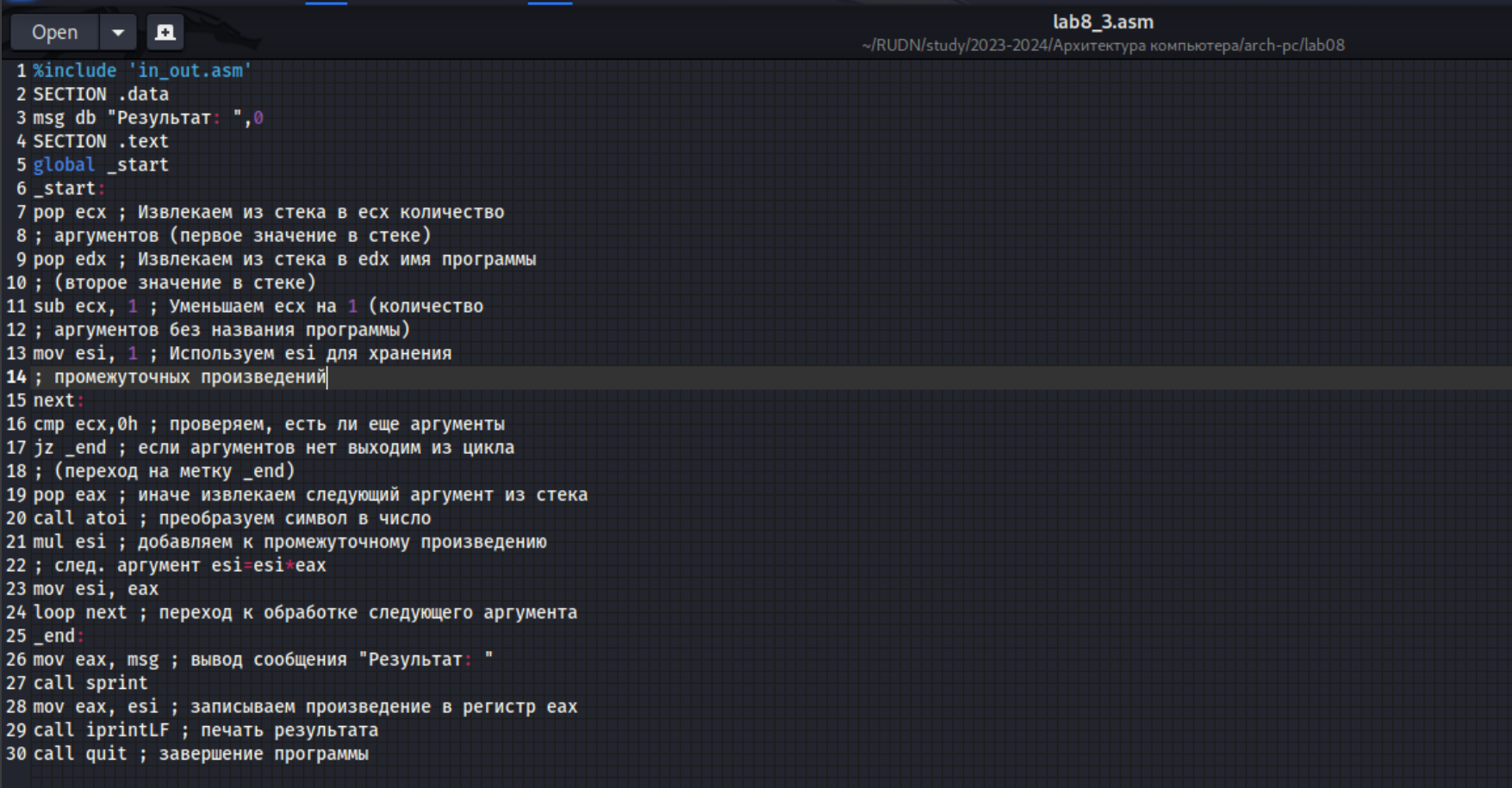
%include 'in\_out.asm'  
SECTION .data  
msg db "Результат: ",0  
SECTION .text  
global \_start  
\_start:  
pop ecx ; Извлекаем из стека в `ecx` количество  
; аргументов (первое значение в стеке)  
pop edx ; Извлекаем из стека в `edx` имя программы  
; (второе значение в стеке)  
sub ecx,1 ; Уменьшаем `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

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



Компиляция и запуск файла

Изменяю текст программы из листинга 8.3 (рис. ??).

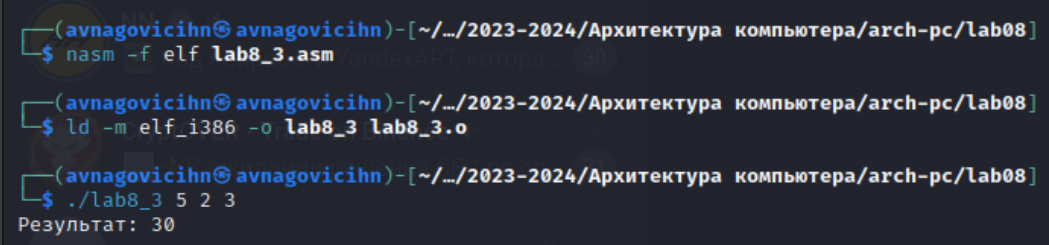


Редактирование файла

**Листинг 8.3.1 Программа вычисления произведения аргументов командной строки**

%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, 1 ; Используем esi для хранения  
; промежуточных произведений  
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  
mul esi ; добавляем к промежуточному произведению  
; след. аргумент esi=esi\*eax  
mov esi, eax  
loop next ; переход к обработке следующего аргумента  
\_end:  
mov eax, msg ; вывод сообщения "Результат: "  
call sprint  
mov eax, esi ; записываем произведение в регистр eax  
call iprintLF ; печать результата  
call quit ; завершение программы

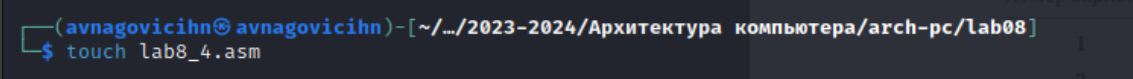
Создаю исполняемый файл и запускаю его (рис. ??).



Компиляция и запуск файла

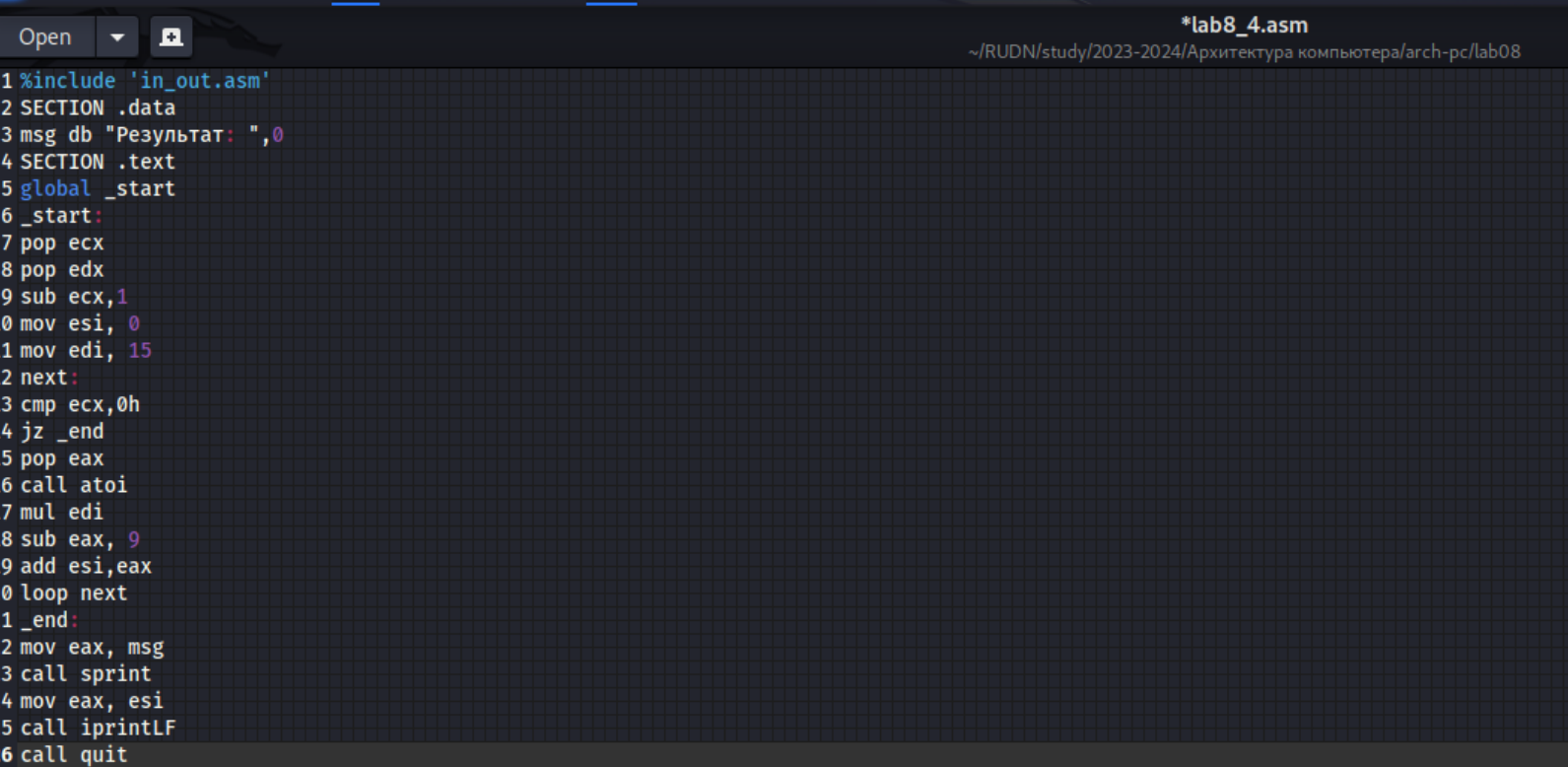
## 3.3 Выполнение заданий для самостоятельной работы

Создаю новый файл lab8\_4.asm (рис. ??).



Создание файла

Пишу программу, которая находит сумму функции f = 15x-9 (12 вариант).(рис. ??).

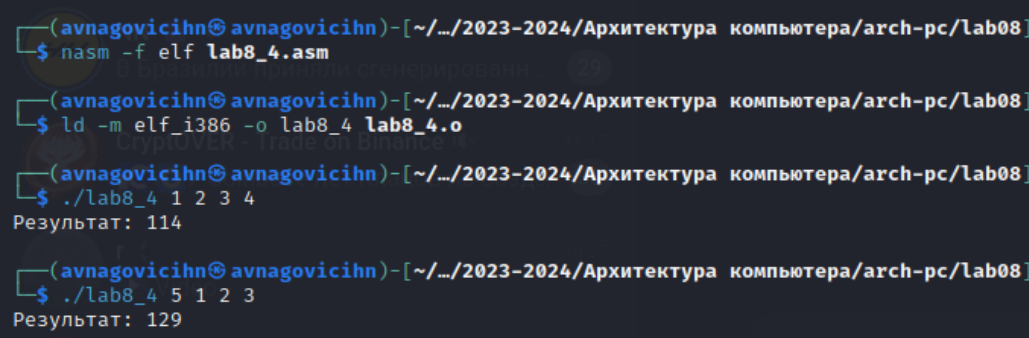


Редактирование файла

**Листинг 8.4 Программа вычисления суммы функции f = 15x-9.**

%include 'in\_out.asm'  
SECTION .data  
msg db "Результат: ",0  
SECTION .text  
global \_start  
\_start:  
pop ecx   
pop edx   
sub ecx,1   
mov esi, 0   
mov edi, 15   
next:  
cmp ecx,0h   
jz \_end   
pop eax   
call atoi   
mul edi  
sub eax, 9  
add esi,eax   
loop next   
\_end:  
mov eax, msg   
call sprint  
mov eax, esi   
call iprintLF   
call quit

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



Компиляция и запуск файла

# 4 Выводы

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