РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8

Студент: Барханоева Раяна Магометовна

Ст.билет: 1032252468

Группа: НКАбд-01-25

МОСКВА

2025 г

Содержание

1. Цель работы

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

2. Выполнение работы

Создание каталога (рисунок 1).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc$ mkdir lab08
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc$ cd lab08/
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ touch lab8-1.asm
```

Рисунок 1.

Создание каталога и файла lab8-1.asm

Программа (рисунок 2).

Рисунок 2.

Программа вывода значений регистра ecx

Тест программы (рисунок 3).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 la
b8-1.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$
```

Рисунок 3.

Данный пример показывает, что использование регистра ecx в теле цилка loop может привести к некорректной работе программы.

Программа (рисунок 4).

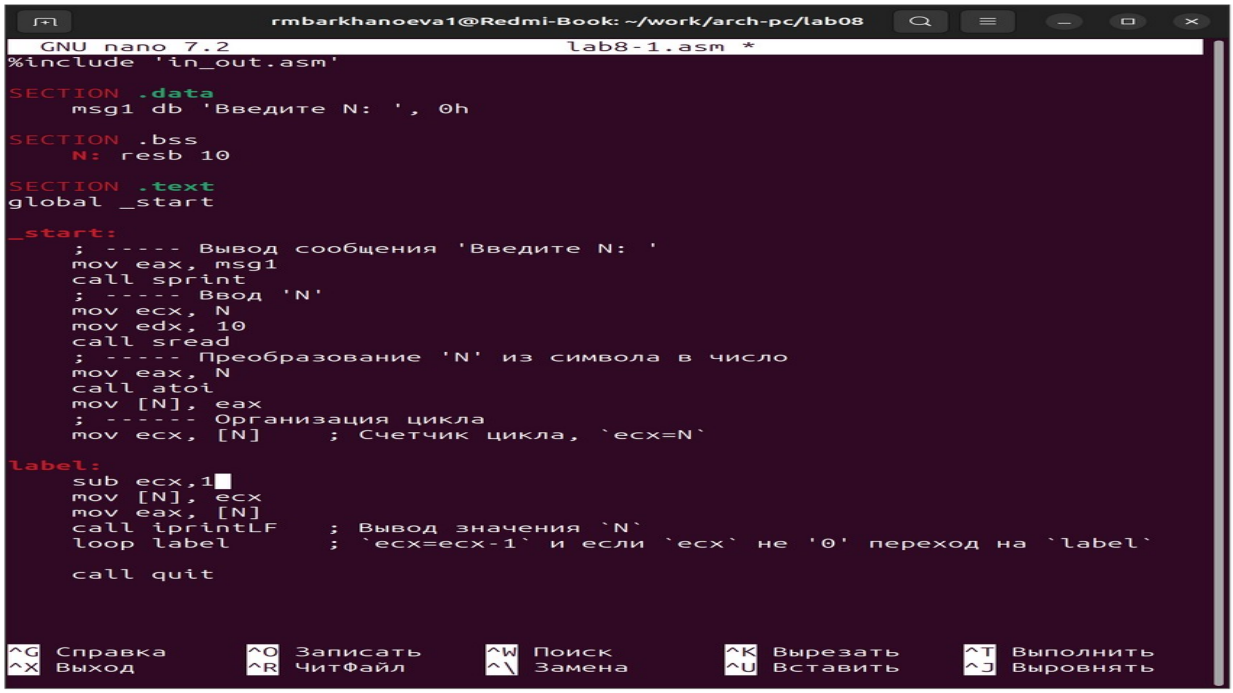

Рисунок 4.

Добавили изменение значения регистра ecx в цикле

Тест программы (рисунок 5.)

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 la
b8-1.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
7
5
3
1
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$
```

Рисунок 5.

Создали исполняемый файл и проверили на работоспособность. Не соответствует количеству проходов.

Программа (рисунок 6).

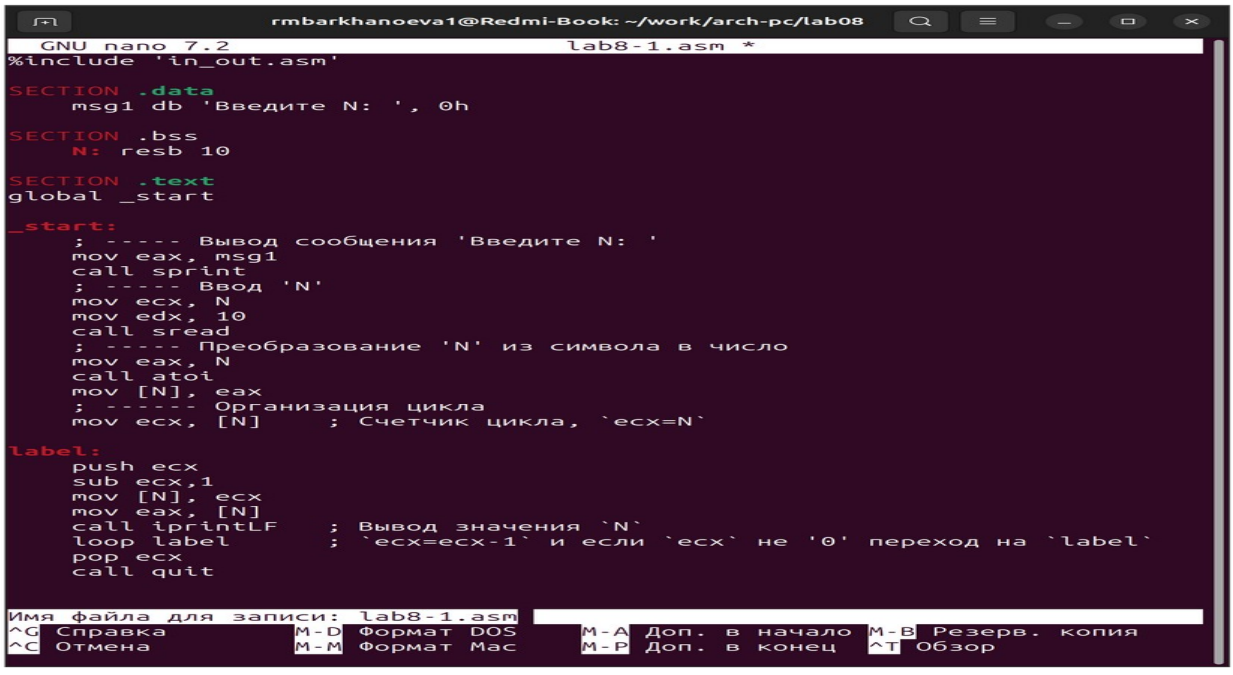

Рисунок 6.

Для использования регистра ecx в цикле и сохранения корректности работы программы можно использовать стек. Внесли изменения в текст программы добавив команды push и pop.

Тест программы (рисунок 7).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 la
b8-1.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
7
5
3
1
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$
```

Рисунок 7.

Не соответствует число проходов.

Программа (рисунок 8).

```
  GNU nano 7.2                          lab8-2.asm *
%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
```

Рисунок 8.

Программа выводящая на экран аргументы командной строки

Тест программы (рисунок 9).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 la
b8-2.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-2 10 20 30
10
20
30
```

Рисунок 9.

Скомпелировал файл lab8-2.asm и выводом стало три аргумента, которые я ввел.

Программа (рисунок 10).

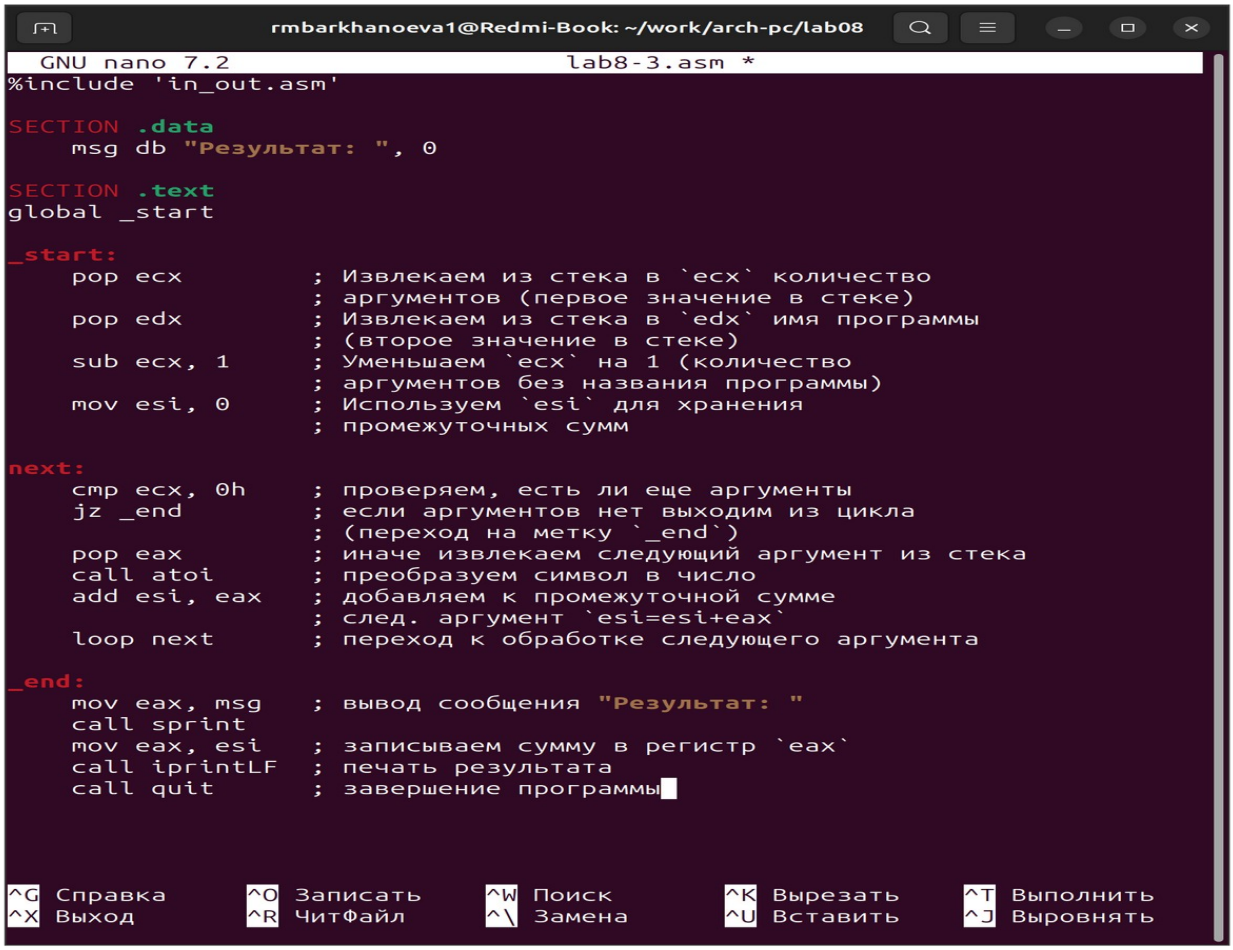

Рисунок 10.

Программа, которая выводит сумму чисел, которые передаются в программу как аргументы.

Тест программы (рисунок 11).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 la
b8-3.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$
```

Рисунок 11.

Скомпелировали программу, ввели аргументы и получили сумму.

Программа (рисунок 12).

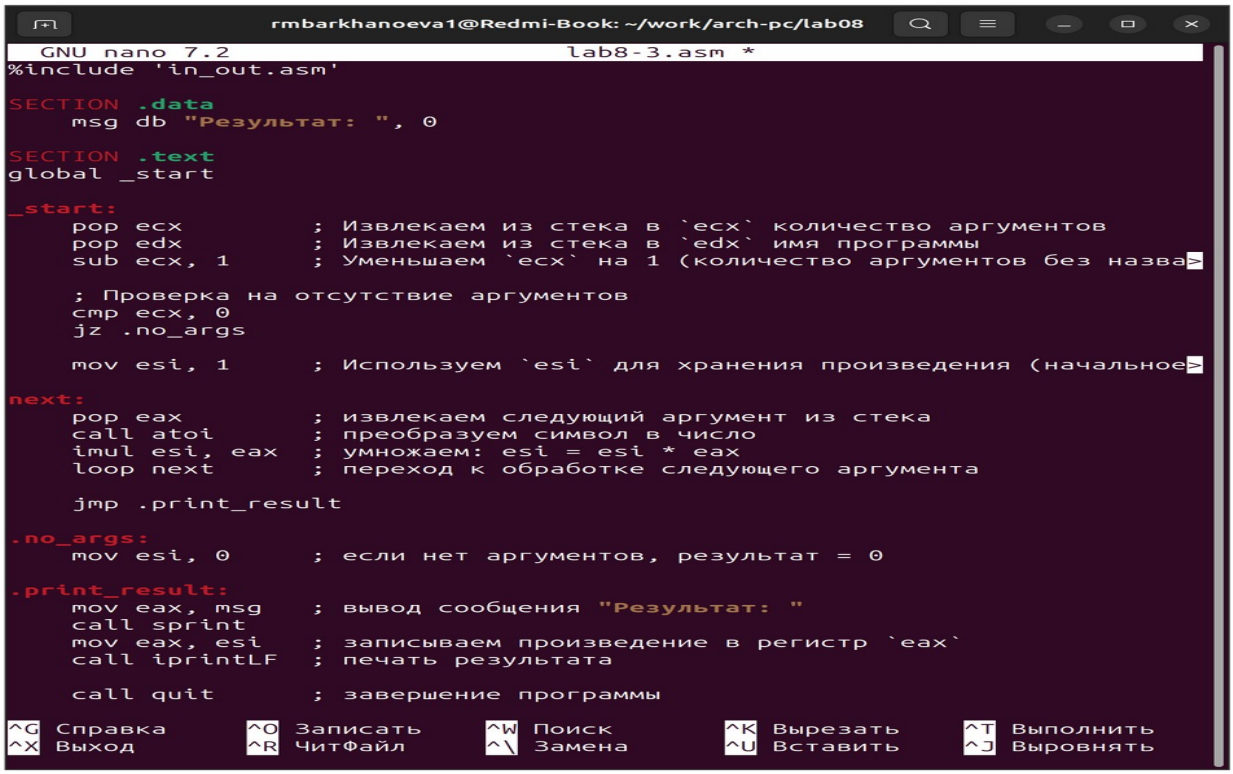

Рисунок 12.

Программа которая перемножает аргументы.

Тест программы (рисунок 13).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 la
b8-3.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-3 5 5 3
Результат: 75
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$
```

Рисунок 13.

Скомпелировали файл, ввели аргументы и программа корректно их перемножила.

3. Самостоятельная работа.

Программа (рисунок 14).

Рисунок 14.

Программа, 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.

Тест программы (рисунок 15).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ nasm -f elf lab8-samrab1.as
m
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-samr
ab1 lab8-samrab1.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-samrab1
Функция: f(x)=4x+3
Результат: 0
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-samrab1 1 2 3 4
Функция: f(x)=4x+3
Результат: 52
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-samrab1 2 2 2 2
Функция: f(x)=4x+3
Результат: 44
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab08$ ./lab8-samrab1 4 3 2 1
Функция: f(x)=4x+3
Результат: 52
```

Рисунок 15.

Скомпелировали файл и ввели аргументы.

Вывод

В ходе лабораторной работы №8 были изучены принципы программирования циклов в NASM и обработки аргументов командной строки. Освоена организация стека, его назначение и операции `push` и `pop`, а также роль регистра `ecx` как счётчика цикла при использовании инструкции `loop`. На практике рассмотрены ошибки, возникающие при изменении регистра `ecx` внутри цикла, и способы их устранения с помощью стека. Также были получены навыки извлечения аргументов командной строки из стека, их последовательной обработки в цикле и выполнения над ними арифметических операций. В результате выполненных заданий сформировано понимание работы стека, циклов и механизмов передачи данных в программах на языке ассемблера NASM.