Отчет по лабораторной работе No 8

Дисциплины: Архитектура компьютера

Pакутуманандзара Цантамписедрана Сарубиди

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

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

# 2 Задание

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

# 3 Выполнение лабораторной работы

1. Реализация циклов в NASM

Я создам каталог для программ лабораторных работ 8, зайду в него и создам файл lab8-1.asm(рис 1)

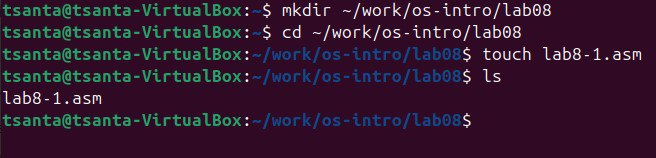


Рис. 1: рис 1

При реализации циклов в NASM с помощью инструкции цикла важно помнить, что эта инструкция использует регистр ecx в качестве счетчика и уменьшает его значение на единицу на каждом шаге. Теперь я открою созданный мной файл, затем скопирую и изучу текст данной программы(рис 2)

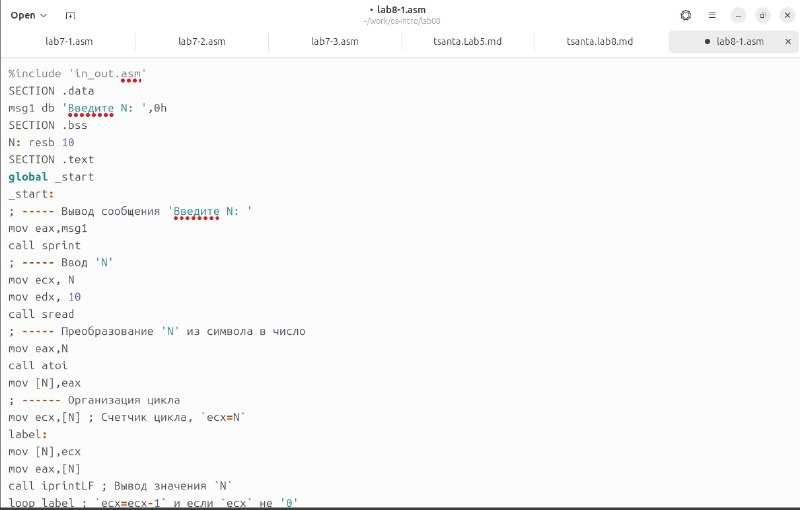


Рис. 2: рис 2

Теперь я создам исполняемый файл и запущу его(рис 3)

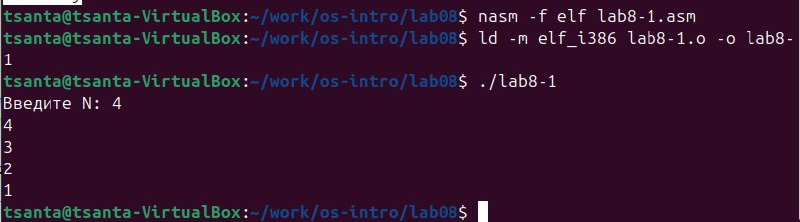


Рис. 3: рис 3

Я изменю текст программы, меняя в цикле значение регистра ecx(рис 4)

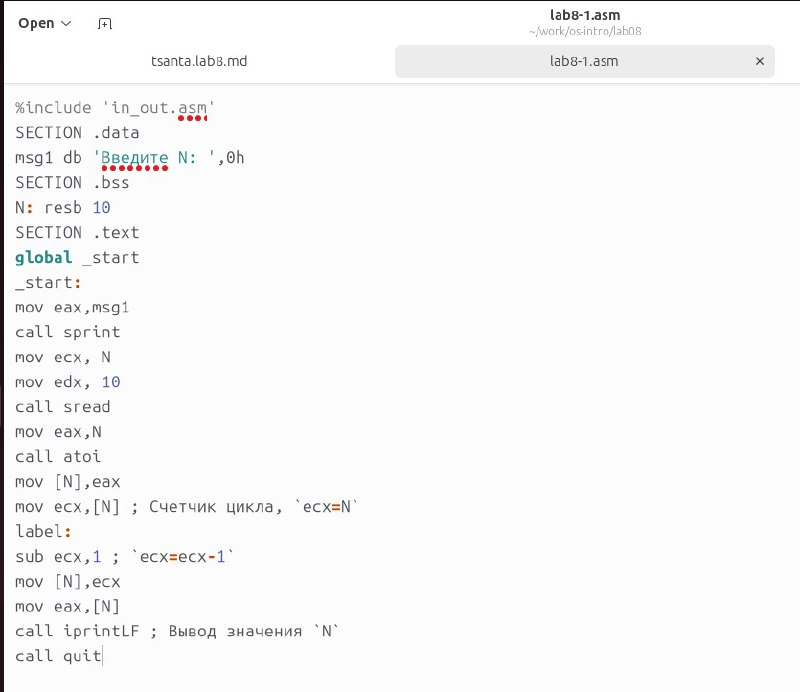


Рис. 4: рис 4

Я создам исполняемый файл и проверю его работу(рис 5)

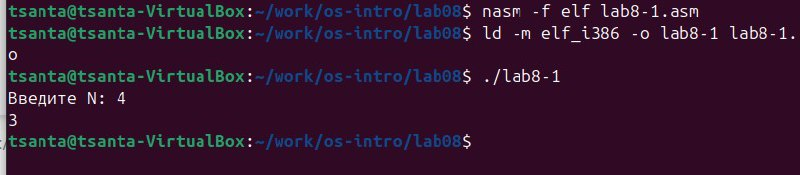


Рис. 5: рис 5

Когда я запускаю программу, она отображает значения 3 , количество циклов не соответствует значению n

Чтобы использовать регистр ecx в цикле и обеспечить правильную работу программы, мне нужно использовать стек. Поэтому я внесу изменения в текст программы, добавив команды push и pop (добавление в стек и извлечение из стека), чтобы сохранить значение счетчика цикла(рис 6)



Рис. 6: рис 6

Я создам исполняемый файл и проверю его работу(рис 7)

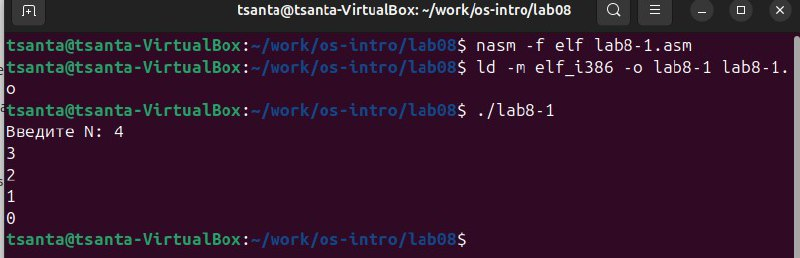


Рис. 7: рис 7

В этом случае количество проходов цикла соответствует значению N, введенному с клавиатуры

1. Обработка аргументов командной строки

Я создам файл lab8-2.asm с помощью команды touch(рис 8)

Рис. 8: рис 8

Рис. 8: рис 8

Когда вы запускаете программу, аргументы располагаются в стеке, поэтому, чтобы использовать аргументы в программе, их просто нужно извлечь из стека. Аргументы должны обрабатываться в цикле. Сначала вам нужно извлечь количество аргументов из стека, а затем просмотреть логику программы для каждого аргумента. Чтобы показать это, я скопирую данную программу в файл, который я только что создал(рис 9)

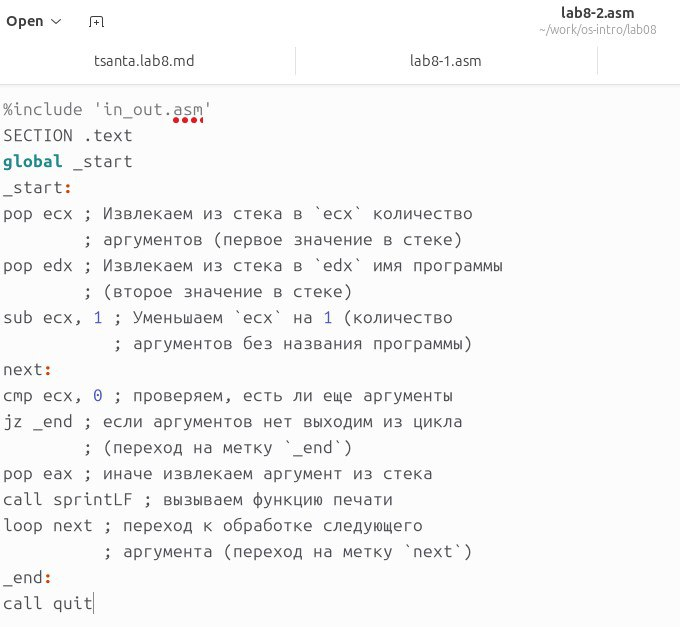


Рис. 9: рис 9

Я создам исполняемый файл и проверю его работу(рис 10)

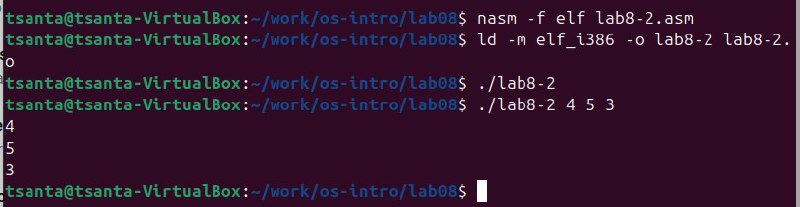


Рис. 10: рис 10

Я ввела три аргумента, и программа обработала количество введенных мной аргументов.

Я создам файл lab8-2.asm с помощью команды touch(рис 11)

Рис. 11: рис 11

Рис. 11: рис 11

Я открою его и скопирую в него заданную программу, программа отображает сумму чисел, которые передаются программе в качестве аргументов(рис 12)

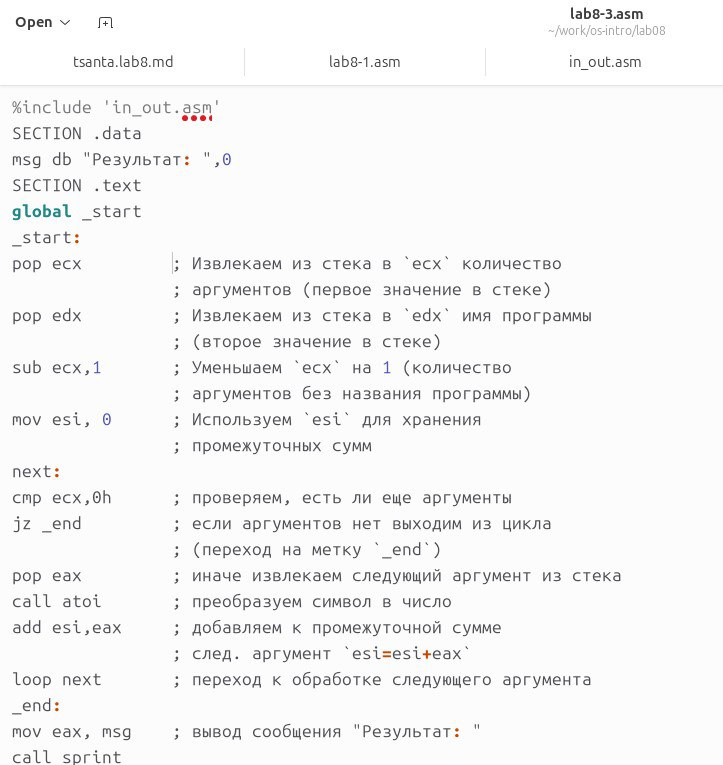


Рис. 12: рис 12

Я создам исполняемый файл и проверю его работу(рис 13)

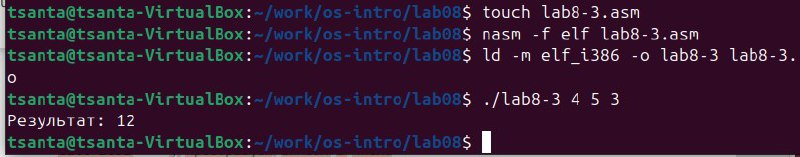


Рис. 13: рис 13

Я изменю программу так, чтобы она вычисляла произведение аргументов командной строки(рис 14)



Рис. 14: рис 14

Я создам исполняемый файл и проверю его работу(рис 15)

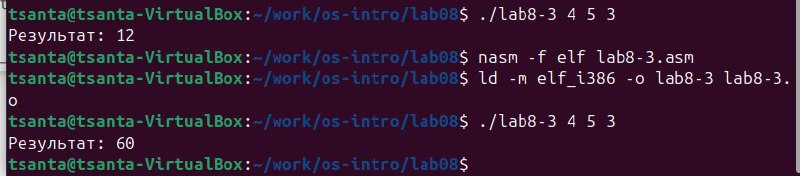


Рис. 15: рис 15

1. Задание для самостоятельной работы

Я создам файл lab8-4.asm с помощью команды touch(рис 16)

Рис. 16: рис 16

Рис. 16: рис 16

В созданном мной файле я напишу программу, которая находит сумму значений функции f(x) для x = x1, x2,x3…. и т. д. Программа должна вывести значение f(x1) + f(x2) + … + f(xn). Значения x передаются в качестве аргументов. Тип функции f(x) я выберу из данной таблицы вариантов задания в соответствии с вариантом, полученным мной в ходе лабораторной работы 6.Мой вариант — вариант 12;f(x) = (15x-9)(рис 17)

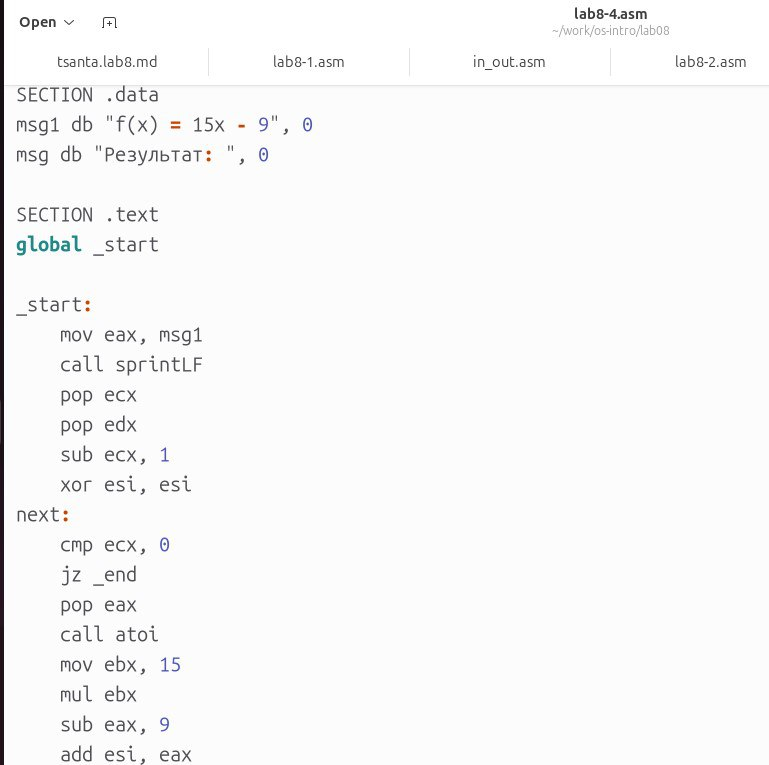
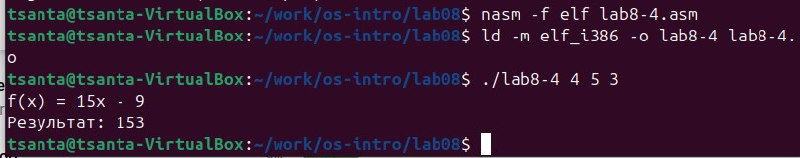


Рис. 17: рис 17

Я создам исполняемый файл и проверю его работу(рис 18)

{#fig:0018 width=70%

Текстовая программа для самостоятельной работы

%include 'in\_out.asm'  
  
SECTION .data  
msg1 db "f(x) = 15x - 9", 0  
msg db "Результат: ", 0  
SECTION .text  
global \_start  
\_start:  
 mov eax, msg1  
 call sprintLF  
 pop ecx   
 pop edx   
 sub ecx, 1   
 xor esi, esi   
next:  
 cmp ecx, 0   
 jz \_end   
 pop eax   
 call atoi   
 mov ebx, 15   
 mul ebx   
 sub eax, 9   
 add esi, eax   
 dec ecx   
 jmp next   
\_end:  
 mov eax, msg   
 call sprint   
 mov eax, esi   
 call iprintLF   
 call quit

# 4 Выводы

Здесь кратко описываются итоги проделанной работы.Выполняя эту лабораторную работу, я приобрел навыки написания программ с использованием циклов и обработки аргументов командной строки.

# Список литературы

Архитектура ЭВМ