# Отчёт по лабораторной работе №8

**Дисциплина: архитектура компьютера**

Кузнецова Елизавета Андреевна

# Содержание

# Список иллюстраций

# Список таблиц

# 1 Цель работы

Получение навыков по организации циклов и работе со стеком на языке NASM.

# 2 Задание

1. Ознакомиться с организацией стека.
2. Написать программу, которая находит сумму значений функции ☒(☒) для ☒ = ☒1, ☒2, …, ☒☒, т.е. программа должна выводить значение ☒(☒1) + ☒(☒2) + … + ☒(☒☒).
3. Загрузить файлы на Github.

# 3 Теоретическое введение

Стек – это специально выделенная область оперативной памяти, использующая механизм безадресной записи и выборки элементов данных. Этот механизм предполагает, что элемент, записанный последним, будет всегда прочитан первым. 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. Важно помнить, что стек растет в направлении к началу памяти и уменьшается в направлении к ее концу. Стек предназначен для временного хранения переменных, передачи параметров вызываемым подпрограммам и сохранения адреса возврата при вызове процедур и прерываний.

# 4 Выполнение лабораторной работы

С помощью утилиты mkdir создала директорию, в которой буду создавать файлы с программами для лабораторной работы. Перешла в созданный каталог с помощью утилиты cd (рис. [4.1]).


Рис. 4.1: Создание директории

С помощью утилиты touch создала файл lab8-1.asm (рис. [4.2]).


Рис. 4.2: Создание файла

Скопировала в текущий каталог файл in_out.asm с помощью утилиты cp, так как он будет использоваться в других программах (рис. [4.3]).

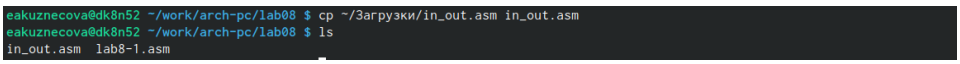


Рис. 4.3: Копирование файла

Открыла созданный файл lab8-1.asm, вставила в него программу вывода значения регистра eax (рис. [4.4]).


Рис. 4.4: Редактирование файла

Создала исполняемый файл программы и запустила его. Проверила его работу несколько раз(рис. [4.5]).

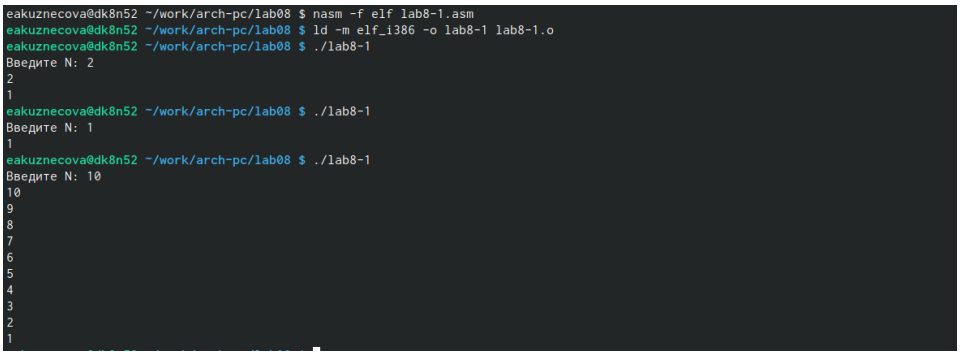


Рис. 4.5: Запуск исполняемого файла

Изменила текст программы файла lab8-1.asm, добавив изменение значения регистра ecx в цикле label (рис. [4.6]).

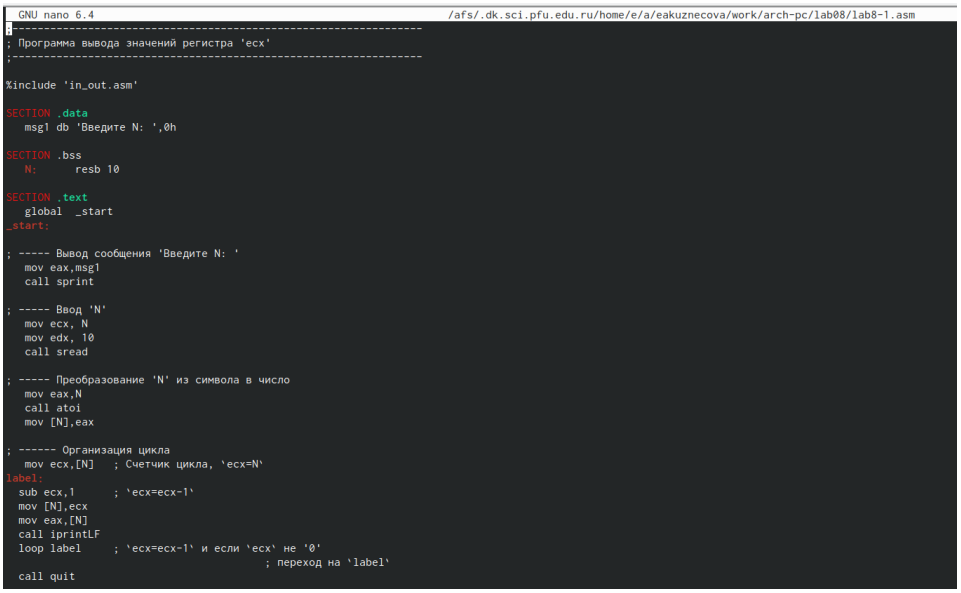


Рис. 4.6: Редактирование файла

Создала новый исполняемый файл программы и запустила его. В этой программе уменьшился изначальный индекс на 1. Получился результат, который отличается от ожидаемого. Получили N/2 значений (рис. [4.7]).

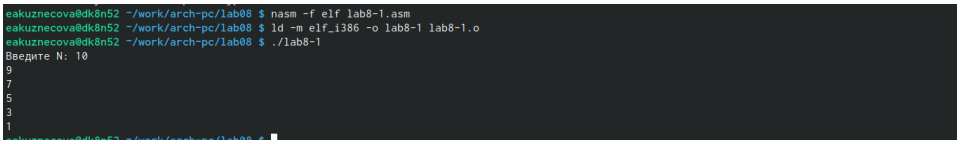


Рис. 4.7: Создание и запуск нового исполняемого файла

Изменила текст программы файла lab8-1.asm, в которую были добавлены изменения значения регистра ecx в цикле label, использовала стек (воспользовалась командами push и pop) (рис. [4.8]).

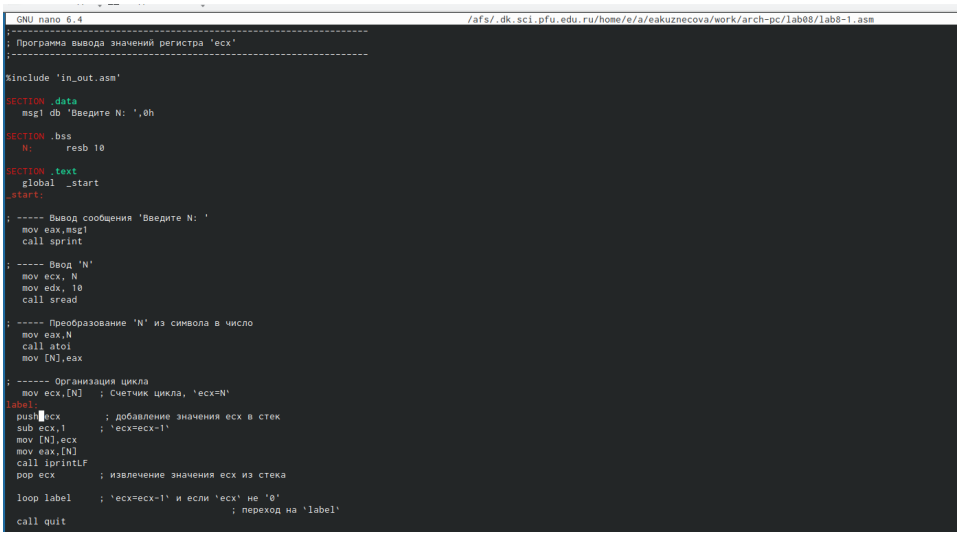


Рис. 4.8: Редактирование файла

Создала новый исполняемый файл программы и запустила его. В этой программе вывелся нужный нам результат (N значений) (рис. [4.9]).

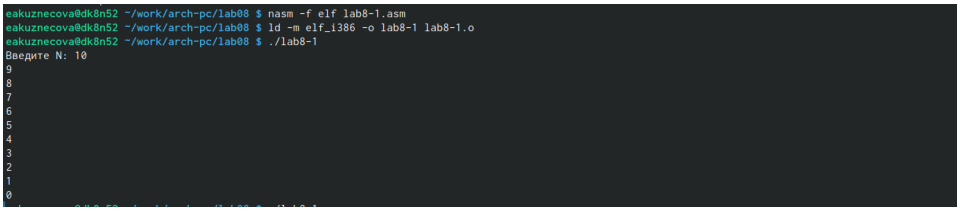


Рис. 4.9: Создание и запуск нового исполняемого файла

С помощью утилиты touch создала файл lab8-2.asm (рис. [4.10]).


Рис. 4.10: Создание файла

Открыла созданный файл lab8-2.asm, вставила в него программу по обработке аргументов командной строки (рис. [4.11]).

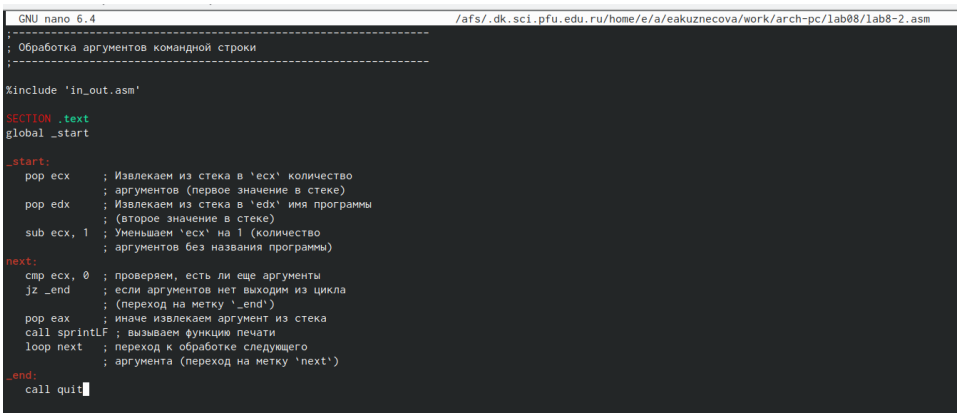


Рис. 4.11: Редактирование файла

Создала новый исполняемый файл программы и запустила его, указав аргументы. Были обработы все введенные аргументы (рис. [4.12]).

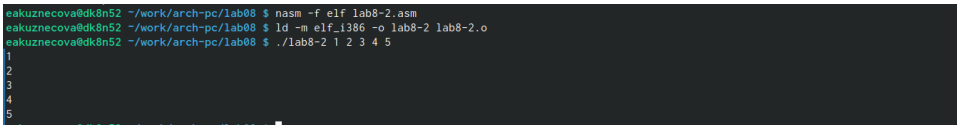


Рис. 4.12: Создание и запуск нового исполняемого файла

С помощью утилиты touch создала файл lab8-3.asm (рис. [4.10]).


Рис. 4.13: Создание файла

Открыла созданный файл lab8-3.asm, вставила в него программу вычисления суммы аргументов командной строки (рис. [4.14]).

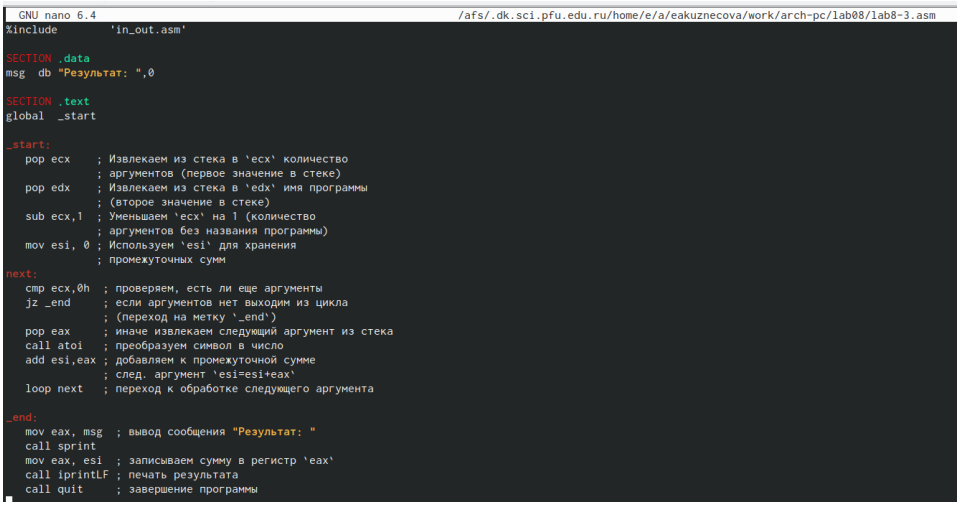


Рис. 4.14: Редактирование файла

Создала новый исполняемый файл программы и запустила его, указав аргументы. Была выведена правильная сумма аргументов (рис. [4.15]).

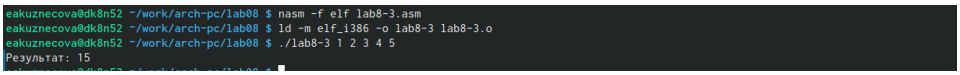


Рис. 4.15: Создание и запуск нового исполняемого файла

С помощью утилиты touch создала файл lab8-4.asm (рис. [4.16]).

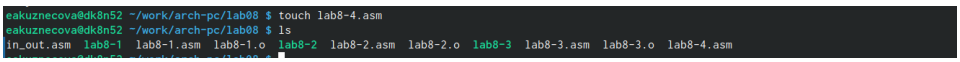


Рис. 4.16: Создание файла

Открыла созданный файл lab8-4.asm, вставила в него программу вычисления произведения аргументов командной строки (рис. [4.17]).

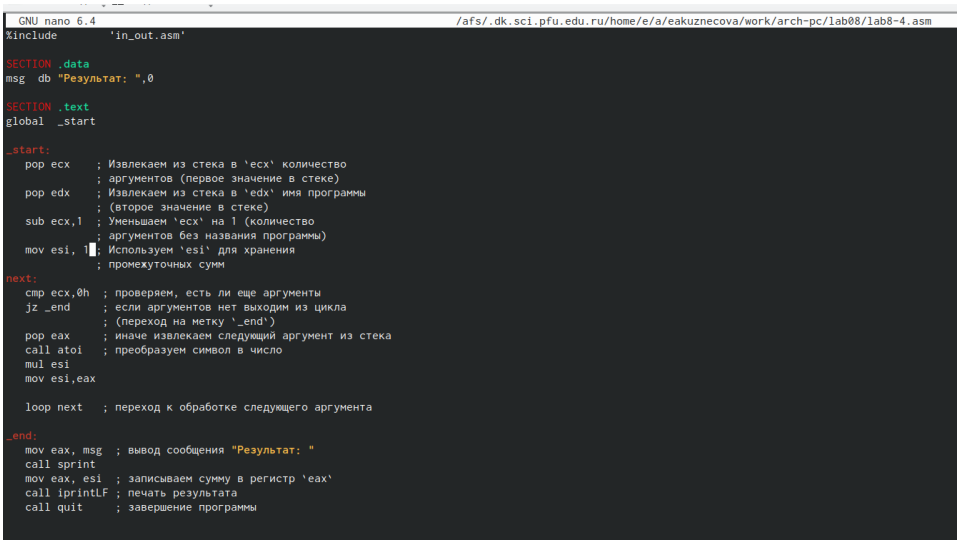


Рис. 4.17: Редактирование файла

Создала новый исполняемый файл программы и запустила его, указав аргументы. Было выведено правильное произведение аргументов (рис. [4.18]).

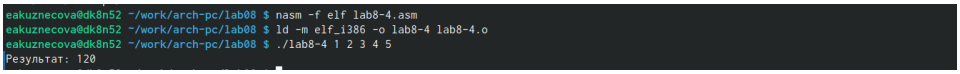


Рис. 4.18: Создание и запуск нового исполняемого файла

# 5 Программа для 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 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
    mul esi
    mov 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
```

# 6 Выполнение заданий для самостоятельной работы

Создала файл lab8-5.asm с помощью утилиты touch (рис. [6.1]).


Рис. 6.1: Создание файла

Открыла созданный файл и ввела в него текст программы, которая находит сумму значений функции f(x)=3(10+x), которое было под 20 вариантом (рис. [6.2]).

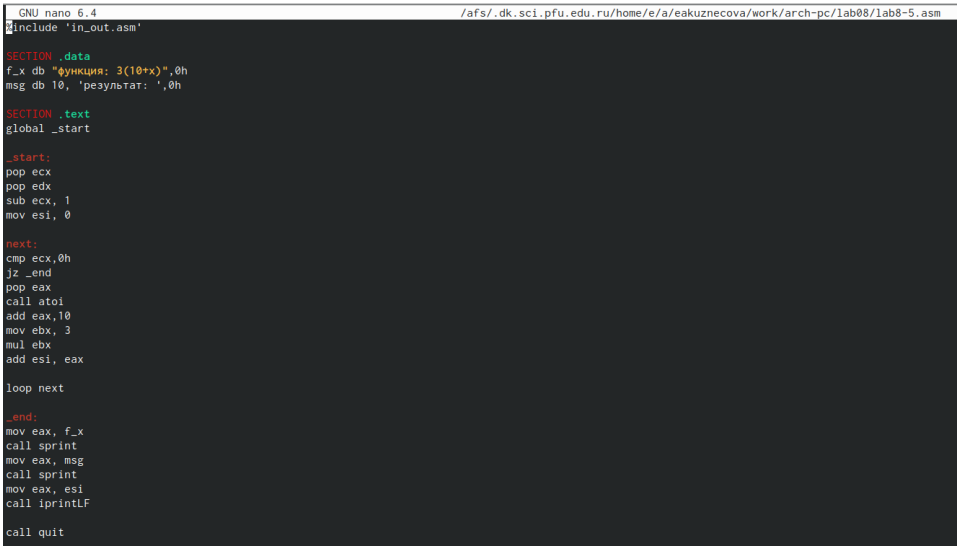


Рис. 6.2: Написание и редактирование программы

Создала новый исполняемый файл и запустила его. Проверила свою программу

на трех наборах x. Все значения выводятся верно. (рис. [6.3]).

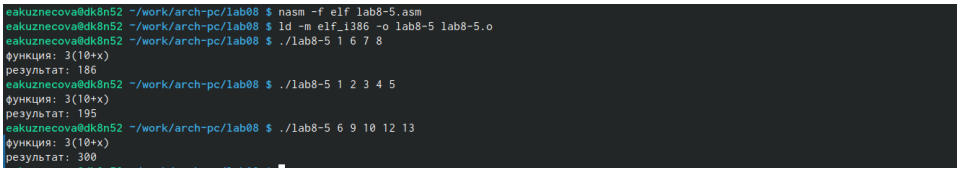


Рис. 6.3: Запуск исполняемого файла

# 7 Программа для вычисления значения выражения 3(10+x)

```
%include 'in_out.asm'

SECTION .data
f_x db "функция: 3(10+x)",0h
msg db 10, 'результат: ',0h

SECTION .text
global _start

_start:
pop ecx
pop edx
sub ecx, 1
mov esi, 0

next:
cmp ecx,0h
jz _end
pop eax
call atoi
```

```
add eax,10
mov ebx, 3
mul ebx
add esi, eax

loop next

_end:
mov eax, f_x
call sprint
mov eax, msg
call sprint
mov eax, esi
call iprintLF

call quit
```

# 8 Выводы

В ходе этой лабораторной работы были получены навыки по организации циклов и работе со стеком на языке NASM.