РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8

Студентка: Белакова Полина Вячеславовна

Ст.билет: 1032252589

Группа: НКАбд-01-25

МОСКВА

2025 г

Цель работы

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

Порядок выполнения лабораторной работы

Реализация циклов в NASM

Создаю каталог для программам лабораторной работы № 8, перехожу в него и создаю файл lab8-1.asm (рисунок 1).

```
pvbelakova2@localhost-live:~$ mkdir ~/work/arch-pc/lab08
pvbelakova2@localhost-live:~$ cd ~/work/arch-pc/lab08
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ touch lab8-1.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab08$
```

Рисунок 1 - создание каталога lab08, перемещение в каталог и создание файла lab8-1.asm.

Ввожу в файл lab8-1.asm текст программы из листинга 8.1 (рисунок 2).

```
mc [pvbelakova2@localhost-live]:~/work/arch-pc/lab08 — /us
lab8-1.asm        [-M--]  9 L:[  1+30  31/ 31] *(844 / 844b) <E
;-------------------------------------------------------------
; 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
```

Рисунок 2 - код программы из листинга 8.1.

Создаю исполняемый файл и проверяю его работу (рисунок 3).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 11
11
10
9
8
7
6
5
4
3
2
1
```

Рисунок 3 - создание исполняемого файла и проверка его работы.

Изменяю текст программы, добавляя изменение значения регистра ecx в цикле: label (рисунок 4).

```
mc [pvbelakova2@localhost-live]:~/work/arch-pc/lab08 —
lab8-1.asm          [-M--]  9 L:[  1+31  32/ 32] *(834 / 834b)
;-------------------------------------------------------------
; 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:
sub ecx,1 ; `ecx=ecx-1`
mov [N],ecx
mov eax,[N]
call iprintLF
loop label ; `ecx=ecx-1` и если `ecx` не '0'
; переход на `label`
call quit
```

Рисунок 4 - изменение label в кода программы.

Создаю исполняемый файл и проверяю его работу (рисунок 5).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm -o lab8-1_1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1_1 lab8-1_1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ ./lab8-1_1
Введите N: 10
9
7
5
3
1
pvbelakova2@localhost-live:~/work/arch-pc/lab08$
```

Рисунок 5 - создание исполняемого файла и проверка его работы.

Регистр ecx каждый цикл принимает значение меньше на 2 (сначала меньше на 1 в команде sub, затем меньше на 1 в loop). Число проходов не соответствует введенному с клавиатуры значению N.

Вношу изменения в текст программы, добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop (рисунок 6).

```
mc [pvbelakova2@localhost-live]:~/work/arch-pc/lab08 — /
lab8-1.asm        [-M--]  9 L:[  1+33  34/ 34] *(953 / 953b)
;---------------------------------------------------------------
; 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:
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 label ; `ecx=ecx-1` и если `ecx` не '0'
; переход на `label`
call quit
```

Рисунок 6 - Рисунок 4 - изменение label в кода программы.

Создаю исполняемый файл и проверяю его работу (рисунок 7).

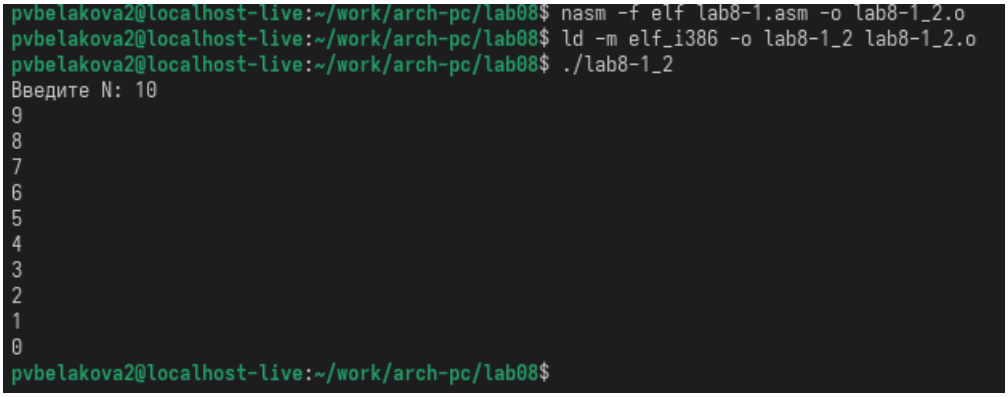

Рисунок 7 - создание исполняемого файла и проверка его работы.

В данном случае число переходов цикла соответствует введенному с клавиатуры N.

Обработка аргументов командной строки

Создаю файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и ввожу в него текст программы из листинга 8.2 (рисунок 8).

Рисунок 8 - создание файла lab8-2.asm.

Ввожу в файл lab8-2.asm текст программы из листинга 8.2 (рисунок 9).

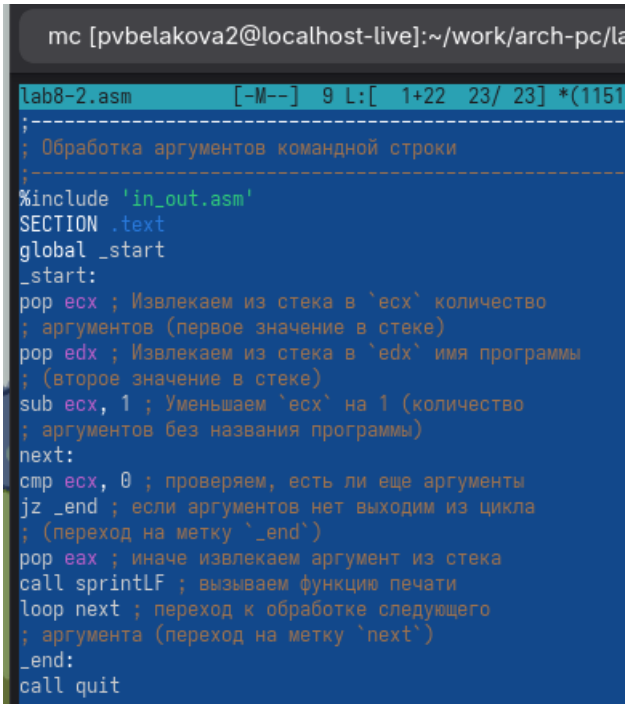

Рисунок 9 - код программы из листинга 8.2.

Создаю исполняемый файл и запускаю его, указав аргументы (рисунок 10).

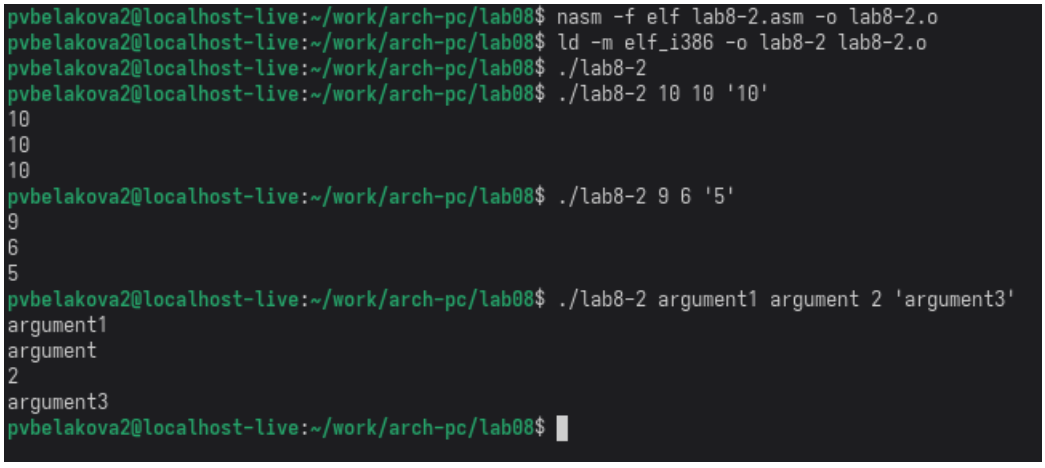

Рисунок 10 - создание исполняемого файла и проверка его работы.

При указании аргументов: argument1 argument 2 ‘argument 3’, было обработано 4 аргумента.

Создаю файл lab8-3.asm (рисунок 11) и ввожу в него текст программы из листинга 8.3 (рисунок 12).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ touch lab8-3.asm
```

Рисунок 11 - создание файла lab8-2.asm.

```
mc [pvbelakova2@localhost-live]:~/work/arch-pc/lab08
lab8-3.asm        [-M--] 32 L:[  1+28  29/ 29] *(1428/1428
%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
```

Рисунок 12 - код программы из листинга 8.3.

Создаю исполняемый файл и запускаю его, указав аргументы (рисунок 13).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm -o lab8-3.o
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
pvbelakova2@localhost-live:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
```

Рисунок 13 - создание исполняемого файла и проверка его работы.

Изменяю текст программы из листинга 8.3 для вычисления произведения аргументов командной строки (рисунок 14).

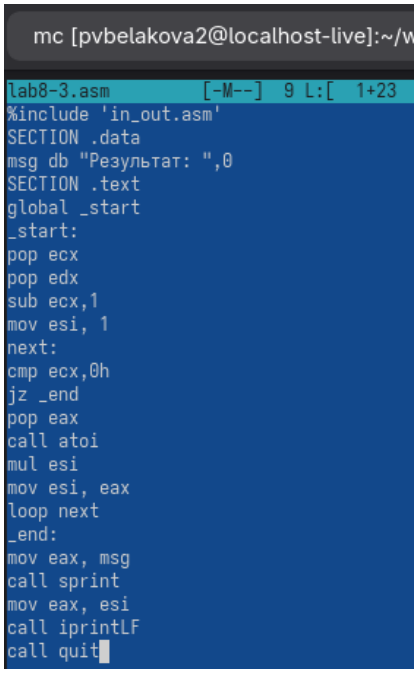

Рисунок 14 - код программы из листинга 8.3.

Создаю исполняемый файл и запускаю его, указав аргументы (рисунок 15).

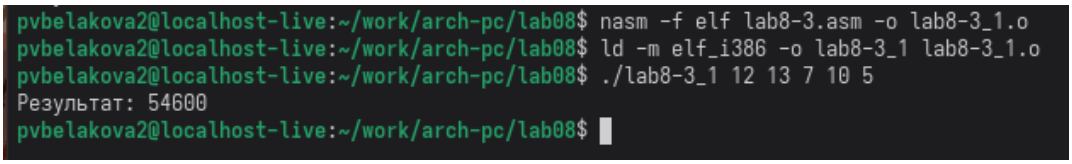

Рисунок 15 - создание исполняемого файла и проверка его работы.

Задание для самостоятельной работы (вариант 10)

1. Создаю файл lab8-task.asm (рисунок 16).

Рисунок 16 - создание файла lab8-task.asm.

Пишу программу, которая находит сумму значений функции 5(2+x) для $x = x1 , x2 , ..., xn$, т.е. программа должна выводить значение $f(x1) + f(x2) + ... + f(xn)$ (рисунок 17).

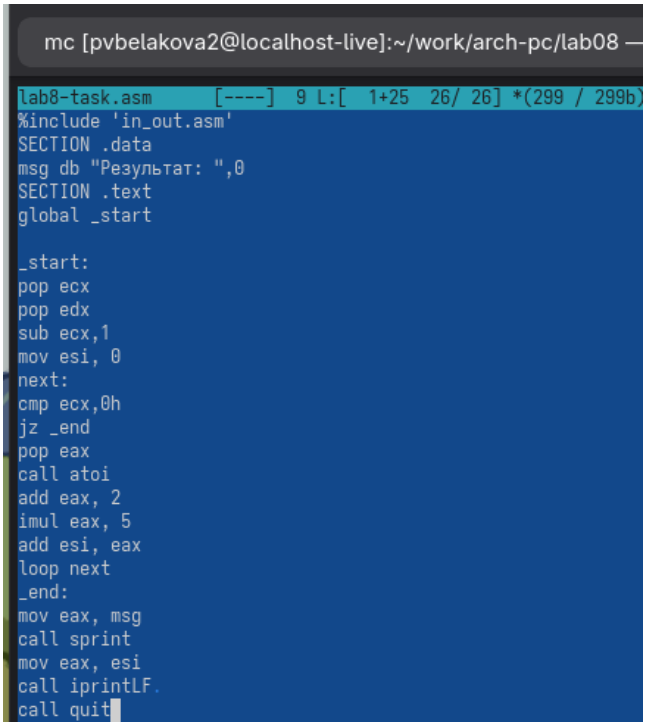

Рисунок 17 - код программы для вычисления суммы 5(2 + x) при разных значения x.

Создаю исполняемый файл и проверяю его работу на нескольких наборах $x = x1 , x2 , ..., xn$ (рисунок 18).

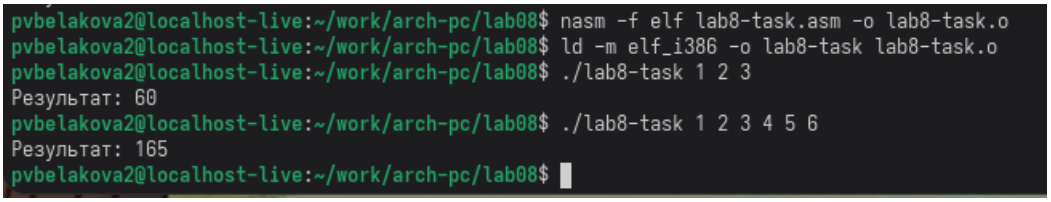

Рисунок 18 - создание исполняемого файла и проверка его работы.

Выводы

В ходе лабораторной работы были приобретены навыки написания программ с использованием циклов и обработкой аргументов командной строки.