

Отчет по выполнению лабораторной работы №8

Дисциплина: архитектура компьютеров

Ясиновская Дарья Олеговна

Содержание

Цель работы	5
Выполнение лабораторной работы	6
Реализация циклов в NASM	6
Самостоятельная работа	16
Выводы	19

Список иллюстраций

1	Создание каталога и файла	6
2	Ввод программы из листинга 8.1	7
3	Проверка работы файла	8
4	Изменение файла	8
5	Проверка работы файла	9
6	Внесение изменений в текст программы	10
7	Запуск программы	11
8	Создание файла lab8-2.asm	11
9	Ввод программы из листинга 8.2	12
10	Запуск программы	13
11	Ввод программы из листинга 8.3	13
12	Проверка работы файла	13
13	Изменение текста листинга 8.3	14
14	Проверка работы программы	15
1	Написание программы для самостоятельной работы	16
2	Запуск программы	17

Список таблиц

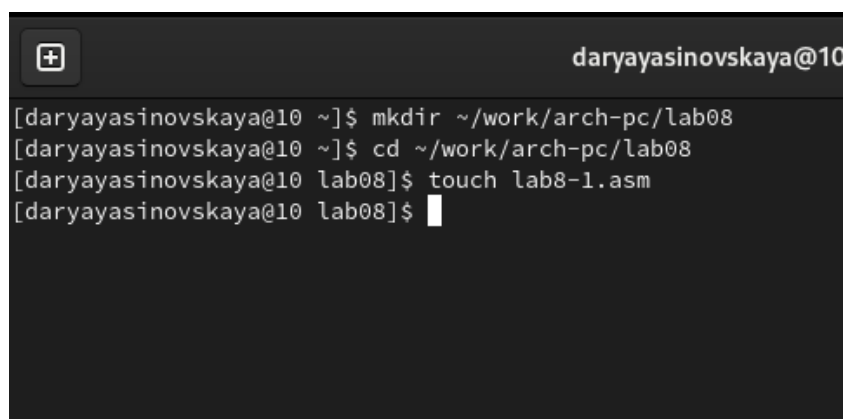
Цель работы

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

Выполнение лабораторной работы

Реализация циклов в NASM

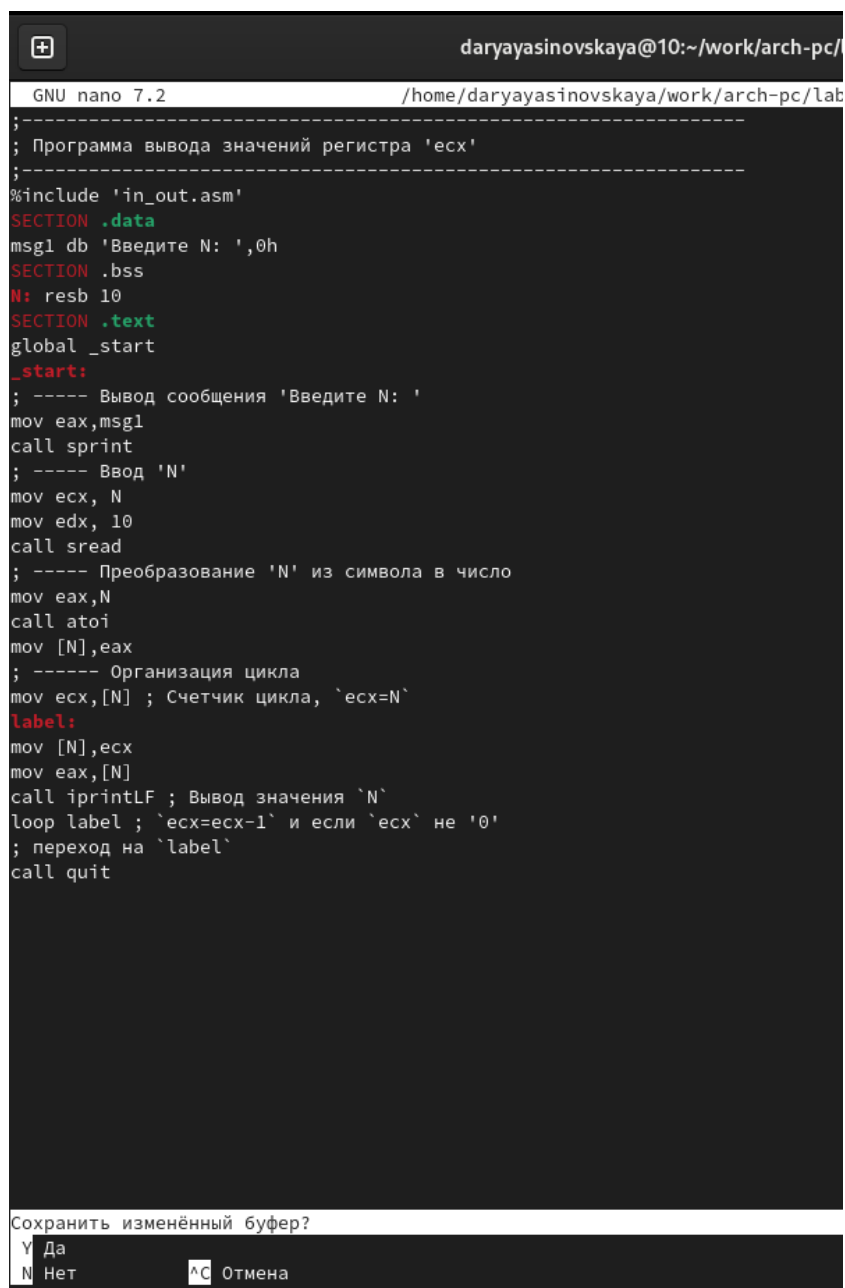
Создаю каталог для программ лабораторной работы №8, перехожу в него и создаю файл lab8-1.asm (рис. [-@fig:001]).

A terminal window with a dark background. The title bar shows a plus icon and the username 'daryayasinovskaya@10'. The terminal contains the following commands and their outputs:

```
[daryayasinovskaya@10 ~]$ mkdir ~/work/arch-pc/lab08  
[daryayasinovskaya@10 ~]$ cd ~/work/arch-pc/lab08  
[daryayasinovskaya@10 lab08]$ touch lab8-1.asm  
[daryayasinovskaya@10 lab08]$
```

Рис. 1: Создание каталога и файла

Ввожу в файл lab8-1.asm текст программы из листинга 8.1.(рис. [-@fig:002]).



```
GNU nano 7.2 /home/daryayasinovskaya/work/arch-pc/lab
;-----
; 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

Сохранить изменённый буфер?
Y Да
N Нет ^C Отмена
```

Рис. 2: Ввод программы из листинга 8.1

Созаю исполняемый файл и проверяю его работу.(рис. [-@fig:003]).

```

[daryayasinovskaya@10 lab08]$ nasm -f elf lab8-1.asm
[daryayasinovskaya@10 lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[daryayasinovskaya@10 lab08]$ ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
[daryayasinovskaya@10 lab08]$ █

```

Рис. 3: Проверка работы файла

Далее изменяю текст программы, добавив изменение значение регистра `ecx` в цикле.(рис. [-@fig:004]).

```

GNU nano 7.2 /home/daryayasinovskaya/work/arch-pc/lab08/
;-----
; 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 ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit

```

Рис. 4: Изменение файла

Создаю исполняемый файл и проверяю его работу.(рис. [-@fig:005]).

```
[daryayasinovskaya@10 lab08]$ nasm -f elf lab8-1.asm
[daryayasinovskaya@10 lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[daryayasinovskaya@10 lab08]$ ./lab8-1
Введите N: 10
9
7
5
3
1
[daryayasinovskaya@10 lab08]$
```

Рис. 5: Проверка работы файла

Мы видим, что число проходов цикла не соответствует значению N.

Вношу изменения в текст программы, добавив команды push и pop.(рис. [-@fig:006]).

```

GNU nano 7.2 /home/daryayasinovskaya/work/arch-p
;-----
; 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
call quit

```

Рис. 6: Внесение изменений в текст программы

Создаю исполняемый файл и запускаю его.(рис. [-@fig:007]).

```
[daryayasinovskaya@10 lab08]$ nasm -f elf lab8-1.asm
[daryayasinovskaya@10 lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[daryayasinovskaya@10 lab08]$ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
[daryayasinovskaya@10 lab08]$
```

Рис. 7: Запуск программы

В данном случае число проходов цикла соответствует значению N.
Создаю файл lab8-2.asm.

```
[daryayasinovskaya@10 lab08]$ touch lab8-2.asm
```

Рис. 8: Создание файла lab8-2.asm

Ввожу в него программу из листинга 8.2

```

GNU nano 7.2 /home/daryayasinovskaya/work/
#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

```

Рис. 9: Ввод программы из листинга 8.2

Создаю исполняемый файл и запускаю его с аргументами 306 10 и “12”.(рис. [-@fig:010]).

```
[daryayasinovskaya@10 lab08]$ nasm -f elf lab8-2.asm
[daryayasinovskaya@10 lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[daryayasinovskaya@10 lab08]$ ./lab8-2 30 10 '12'
30
10
12
[daryayasinovskaya@10 lab08]$
```

Рис. 10: Запуск программы

Программой было обработано 3 аргумента.

Создаю файл lab8-3.asm и ввожу в него программу из листинга 8.3.(рис. [-@fig:011]).

```
[daryayasinovskaya@10 lab08]$ touch lab8-3.asm
```

Рис. 11: Ввод программы из листинга 8.3

Созаю исполняемый файл и запускаю его, указав аргументы.(рис. [-@fig:002]).

```
[daryayasinovskaya@10 lab08]$ nasm -f elf lab8-3.asm
[daryayasinovskaya@10 lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o
[daryayasinovskaya@10 lab08]$ ./lab8-3 17 24 5 10
Результат: 56
[daryayasinovskaya@10 lab08]$
```

Рис. 12: Проверка работы файла

Программа работает.

Теперь изменяю текст программы из листинга 8.3 так, чтобы он вычислял произведение аргументов каждой строки.(рис. [-@fig:013]).

```

GNU nano 7.2 /home/daryayasinovskaya/work/arch-
%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 iprintf ; печать результата
call quit ;завершение программы

```

Рис. 13: Изменение текста листинга 8.3

Создаю исполняемый файл и запускаю его, указав аргументы.(рис. [-@fig:114]).

```
[daryayasinovskaya@10 lab08]$ nasm -f elf lab8-3.asm
[daryayasinovskaya@10 lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o
[daryayasinovskaya@10 lab08]$ ./lab8-3 2 6 9
Результат: 108
[daryayasinovskaya@10 lab08]$
```

Рис. 14: Проверка работы программы

Самостоятельная работа

Создаю файл lab8-4.asm и начинаю написание программы, которая находит сумму значений функции $f(x)$ для своего варианта(вариант 16).(рис. [-@fig:015]).

```
GNU nano 7.2 /home/daryayasnovskaya/work/a
#include 'in_out.asm'
SECTION .data
msg1 db "Результат: ",0
msg2 db "Функция: f(x)=4x-3"
SECTION .text
global _start
_start:
mov eax,msg2
call sprintf
pop ecx
pop edx
sub ecx,1
mov esi, 0
mov edi, 4
next:
cmp ecx,0h
jz _end
pop eax
call atoi
mul edi
sub eax,3
add esi,eax
loop next
_end:
mov eax, msg1
call sprintf
mov eax, esi
call iprintfLF
call quit
```

Рис. 1: Написание программы для самостоятельной работы

Создаю исполняемый файл и запускаю его, указав аргументы.(рис. [-@fig:014]).

```
[daryayasinovskaya@10 lab08]$ nasm -f elf lab8-4.asm
[daryayasinovskaya@10 lab08]$ ld -m elf_i386 -o lab8-4 lab8-4.o
[daryayasinovskaya@10 lab08]$ ./lab8-4 1 2 3 4
Функция :  $f(x) = 4x - 3$ Linker: LLD 16.0.6
Результат: 28
[daryayasinovskaya@10 lab08]$
```

Рис. 2: Запуск программы

Программы работает корректно.

Текст программы:

`%include 'in_out.asm'`

`SECTION .data`

`msg db "Результат:",0`

`msg2 db "Функция: $f(x)=4x-3$ "`

`SECTION .text`

`global _start`

`_start:`

`mov eax,msg2`

`call sprintLF`

`pop ecx`

`pop edx`

`sub ecx,1`

`mov esi,0`

`mov edi,4 next:`

`cmp ecx,0h`

`jz _end`

`pop eax`

`call atoi`

`mul edi`

`sub eax,3`

```
add esi,eax
loop next
_end:
mov eax, msg
call sprint
mov eax, esi
call iprintLF
call quit
```

Выводы

Я научилась написанию программ с использованием циклов и обработкой аргументов командной строки.