

Лабораторная работа №8

Архитектура компьютера и операционные системы

Бабенко Константин, НКАбд-01-23

Содержание

1	Цель работы	1
2	Выполнение лабораторной работы.....	1
2.1	Домашняя работа.....	Ошибка! Закладка не определена.
3	Выводы	6

1 Цель работы

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

2 Выполнение лабораторной работы

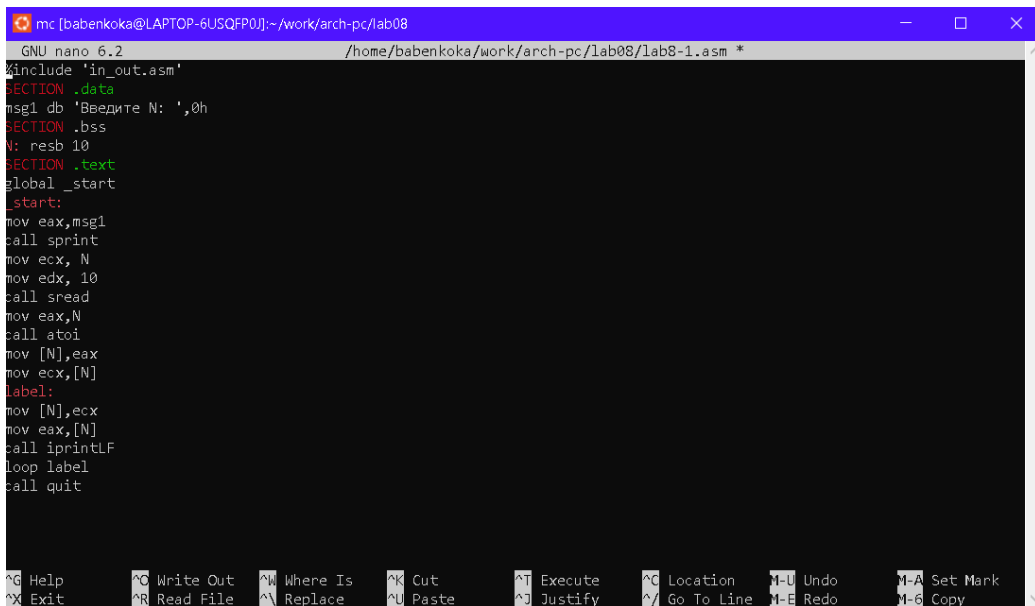
Реализация циклов в NASM

Создаю каталог для программ лабораторной работы № 8, перехожу в него и создаю файл lab8-1.asm:

```
abenkoka@LAPTOP-6USQFP0J:~$ mkdir ~/work/arch-pc/lab07
abenkoka@LAPTOP-6USQFP0J:~$ cd ~/work/arch-pc/lab07
abenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab07$ touch lab7-1.asm
```

Figure 1: Создаю файл.

Ввожу в файл lab8-1.asm текст программы из листинга 8.1:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab08/lab8-1.asm *
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx, N
mov edx, 10
call sread
mov eax,N
call atoi
mov [N],eax
mov ecx,[N]
label:
mov [N],ecx
mov eax,[N]
call iprintLF
loop label
call quit
```

Figure 2: Ввожу текст программы.

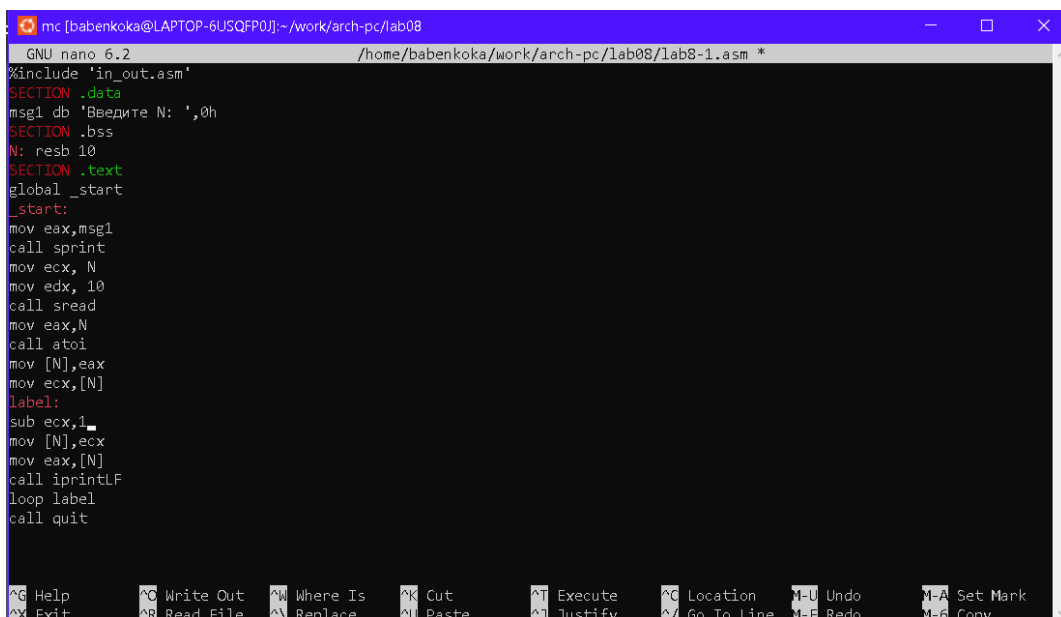
Создаю исполняемый файл и запускаю его:



```
root@LAPTOP-6USQFP0J: /home# ./lab8-1
Введите N: 10
10
9
8
7
6
5
4
3
2
1
root@LAPTOP-6USQFP0J: /home#
```

Figure 3: Работа файла.

Изменяю текст программы добавив изменение значения регистра ecx в цикле:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab08/lab8-1.asm *
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx, N
mov edx, 10
call sread
mov eax,N
call atoi
mov [N],eax
mov ecx,[N]
label:
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
loop label
call quit
```

Figure 4: Измененный файл.

Создаю исполняемый файл и запускаю его:

```
root@LAPTOP-6USQFP0J:/home# ./lab8-1
Введите N: 10
9
7
5
3
1
root@LAPTOP-6USQFP0J:/home#
```

Figure 5: Работа измененного файла.

Регистр принимает значения: 9, 7, 5, 3, 1. Число проходов цикла не соответствует 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:

```
mc [babenkoka@LAPTOP-6USQFP0J:~/work/arch-pc/lab08]
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab08/lab8-1.asm *
#include "in_out.asm"
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx, N
mov edx, 10
call sread
mov eax,N
call atoi
mov [N],eax
mov ecx,[N]
label:
push ecx
sub ecx,1
mov [N],ecx
mov eax,[N]
call iprintLF
pop ecx
loop label
call quit
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark
^X Exit      ^R Read File  ^N Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-R Redo      M-G Copy
```

Figure 6: Измененный текст.

Создаю исполняемый файл и запускаю его:

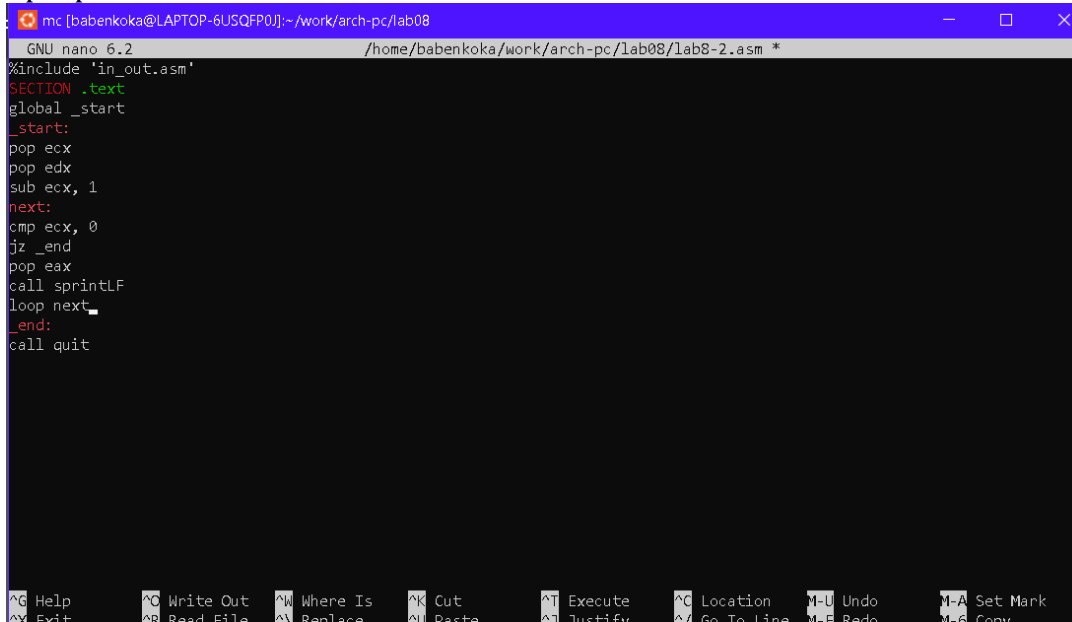
```
root@LAPTOP-6USQFP0J:/home# ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
root@LAPTOP-6USQFP0J:/home#
```

Figure 7: Работа файла.

Число проходов цикла соответствует значению N.

Обработка аргументов командной строки

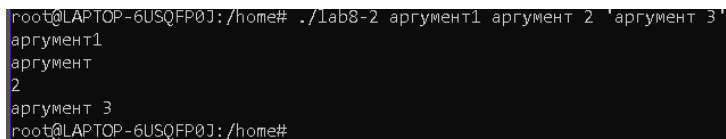
Создаю файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и ввожу в него текст программы из листинга 8.2:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab08/lab8-2.asm *
#include 'in_out.asm'
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx, 1
next:
cmp ecx, 0
jz _end
pop eax
call sprintf
loop next
_end:
call quit
```

Figure 8: Ввожу текст программы.

Создаю исполняемый файл и запускаю его, указав аргументы:



```
root@LAPTOP-6USQFP0J:/home# ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
root@LAPTOP-6USQFP0J:/home#
```

Figure 9: Работа файла.

Создаю файл lab8-3.asm в каталоге ~/work/archpc/lab08 и ввожу в него текст программы из листинга 8.3:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab08/lab8-3.asm *
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
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 esi, eax
loop next
_end:
mov eax, msg
call sprintf
mov eax, esi
call iprintf
call quit
```

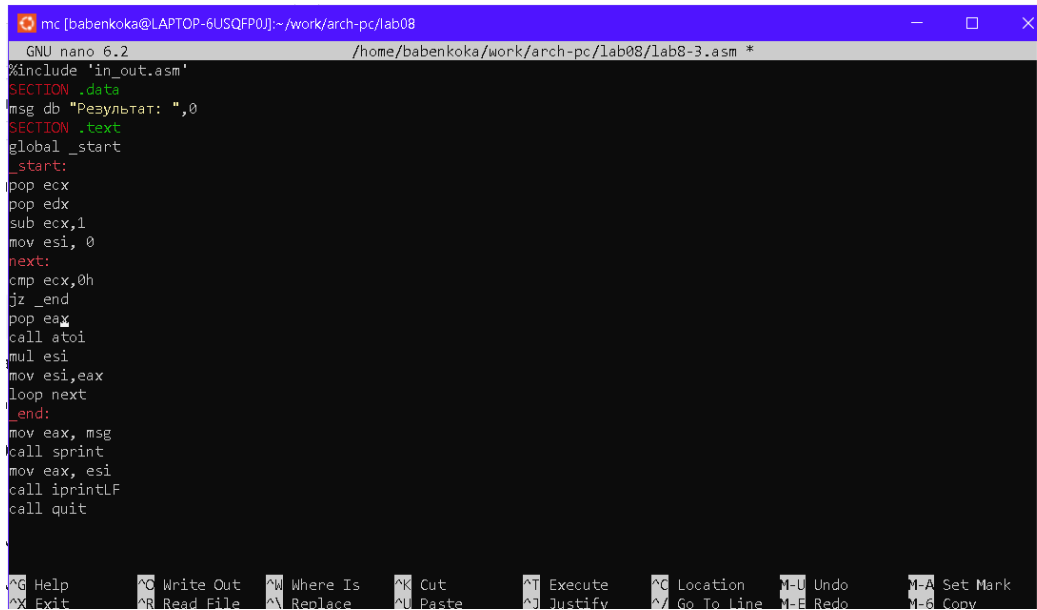
Figure 10: Программа.

Создаю исполняемый файл и проверяю его работу:

```
root@LAPTOP-6USQFP0J:/home# ./lab8-3 12 13 7 10 5
Результат: 47
root@LAPTOP-6USQFP0J:/home#
```

Figure 11: Работа файла.

Изменяю текст программы из листинга 8.3 для вычисления произведения аргументов командной строки:



```
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab08/lab8-3.asm *
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
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
mul esi, eax
mov esi, eax
loop next
_end:
mov eax, msg
call sprint
mov eax, esi
call iprintLF
call quit
```

Figure 12: Измененная программа.

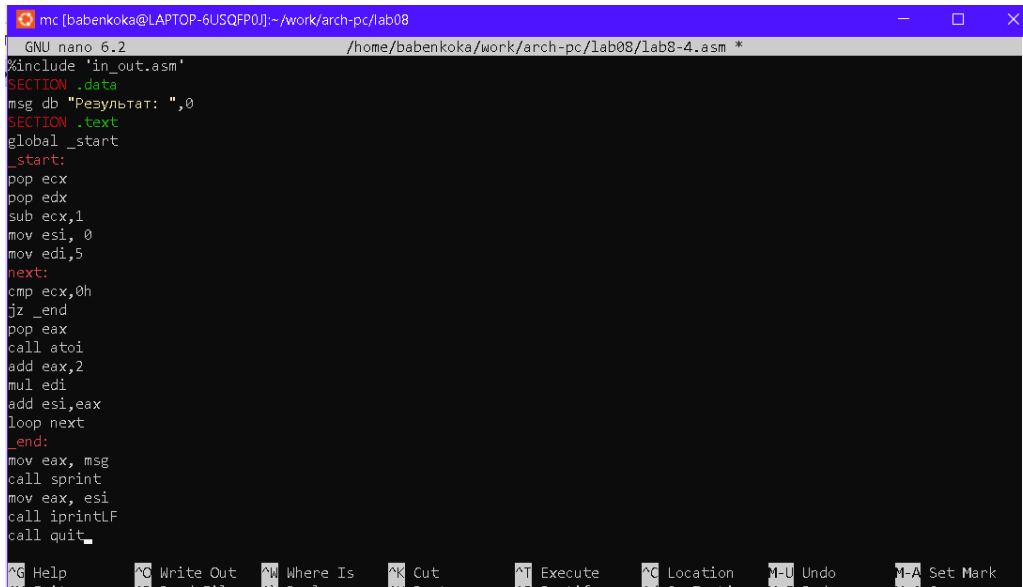
Создаю исполняемый файл и проверяю его работу:

```
root@LAPTOP-6USQFP0J:/home# ./lab8-3 12 13 7 10 5
segmentation fault
root@LAPTOP-6USQFP0J:/home#
```

Figure 13: Работа файла.

2 Домашняя работа

Программа, которая находит сумму значений функции $f(x) = 5 \cdot (2 + x)$ для $x = x_1, x_2, \dots, x_n$, т.е. программа выводит значение $f(x_1) + f(x_2) + \dots + f(x_n)$:



```
mc [babenkoka@LAPTOP-6USQFP0J]~/work/arch-pc/lab08
GNU nano 6.2 /home/babenkoka/work/arch-pc/lab08/lab8-4.asm *
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx
pop edx
sub ecx,1
mov esi, 0
mov edi,5
next:
cmp ecx,0h
jz _end
pop eax
call atoi
add eax,2
mul edi
add esi,eax
loop next
_end:
mov eax, msg
call sprint
mov eax, esi
call iprintf
call quit
```

Figure 14: Текст файла.

Создаю исполняемый файл и проверяю его работу:



```
root@LAPTOP-6USQFP0J:/home# ./lab8-4 1 2
Результат: 35
root@LAPTOP-6USQFP0J:/home#
```

Figure 15 : Работа файла.

3 Выводы

Я приобрел навыки написания программ с использованием циклов и обработкой аргументов командной строки.