

Отчет по лабораторной работе №8

дисциплина: Архитектура компьютера

Сергеев Даниил Олегович

Содержание

1	Цель лабораторной работы	4
2	Ход выполнения лабораторной работы	5
3	Ход выполнения заданий для самостоятельной работы	14
4	Вывод	18

Список иллюстраций

2.1	Создание lab8-1.asm	5
2.2	Код файла lab8-1.asm	6
2.3	Проверка работы lab8-1	6
2.4	Изменение lab8-1.asm	7
2.5	Вывод программы с нечетными N	8
2.6	Вывод программы с четными N	8
2.7	Изменение lab8-1.asm со стеком	9
2.8	Вывод программы со стеком	9
2.9	Создание lab8-2.asm	10
2.10	Код программы lab8-2 из листинга	10
2.11	Проверка файла lab8-2	10
2.12	Создание lab8-3.asm	11
2.13	Код программы lab8-3 из листинга	11
2.14	Проверка файла lab8-3	12
2.15	Изменение кода программы lab8-3.asm	12
2.16	Проверка работы измененного файла lab8-3	13
3.1	Создаем файл lab8-4.asm	14
3.2	Результат работы программы lab8-4	17

1 Цель лабораторной работы

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

2 Ход выполнения лабораторной работы

1. Создаем каталог для программ лабораторной работы №8. Переходим в него и создаем файл lab8-1.asm.

```
frigatzero@fedora:~$ mkdir ~/work/arch-pc/lab08
frigatzero@fedora:~$ cd ~/work/arch-pc/lab08
frigatzero@fedora:~/work/arch-pc/lab08$ touch lab8-1.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ls
lab8-1.asm
frigatzero@fedora:~/work/arch-pc/lab08$
```

Рис. 2.1: Создание lab8-1.asm

2. Вводим в файл lab8-1.asm код из листинга 8.1, создаем исполняемый файл и проверяем его работу.

```

lab8-1.asm      [----] 17 L: [ 1+29 30/ 30] *(338 / 338b) <EOF>  [*] [X]
#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

```

Рис. 2.2: Код файла lab8-1.asm

```

frigatzero@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 5
5
4
3
2
1
frigatzero@fedora:~/work/arch-pc/lab08$

```

Рис. 2.3: Проверка работы lab8-1

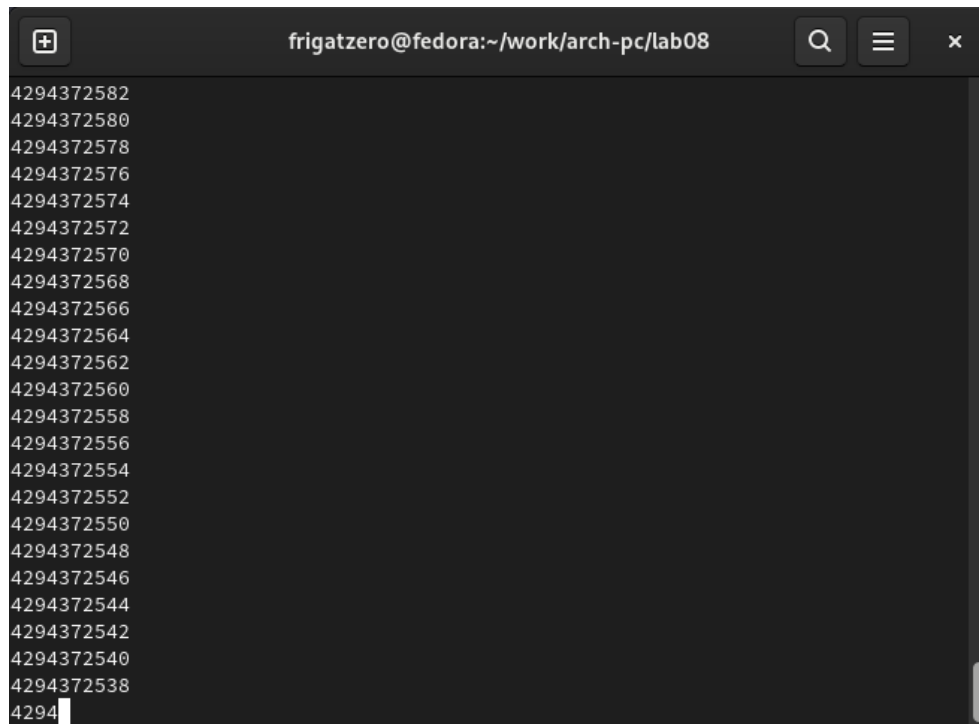
3. Изменяем текст программы так, чтобы значение регистра esx из-

менялось в цикле.

```
label:
<----->sub ecx, 1
<----->mov [N], ecx
<----->mov eax, [N]
<----->call iprintLF
<----->loop label
```

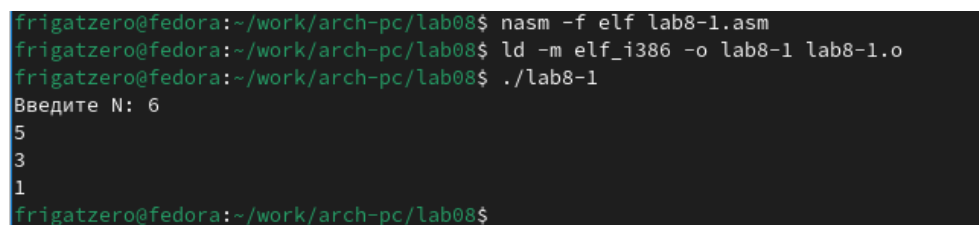
Рис. 2.4: Изменение lab8-1.asm

4. Создадим исполняемый файл измененной программы и проверим его работу. При нечетных N, значение ecx доходит до 0 и из него вычитается 1, из-за чего все биты ecx становятся равными 1 и счетчик начинает идти с максимального значения ecx. При четных N, значение ecx доходит до 1 и, после вычитания из него 1, цикл заканчивается, также число проходов цикла становится меньше значения N введенного с клавиатуры.



```
frigatzero@fedora:~/work/arch-pc/lab08
4294372582
4294372580
4294372578
4294372576
4294372574
4294372572
4294372570
4294372568
4294372566
4294372564
4294372562
4294372560
4294372558
4294372556
4294372554
4294372552
4294372550
4294372548
4294372546
4294372544
4294372542
4294372540
4294372538
4294
```

Рис. 2.5: Вывод программы с нечетными N



```
frigatzero@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 6
5
3
1
frigatzero@fedora:~/work/arch-pc/lab08$
```

Рис. 2.6: Вывод программы с четными N

5. Вписываем в программу команды для добавления в стек и извлечения из стека.


```

    label:
<----->push ecx
<----->sub ecx, 1
<----->mov [N], ecx
<----->mov eax, [N]
<----->call iprintLF
<----->pop ecx
<----->
<----->loop label

```

Рис. 2.7: Изменение lab8-1.asm со стеком

6. Создадим исполняемый файл и проверим его работу. В данном случае число проходов цикла равно введенному при любых значениях N, однако каждый элемент меньше на единицу.

```

frigatzero@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 5
4
3
2
1
0
frigatzero@fedora:~/work/arch-pc/lab08$

```

Рис. 2.8: Вывод программы со стеком

7. Создадим файл lab8-2.asm и введем в него текст программы из листинга 8.2. Создадим исполняемый файл и запустим его с заданными аргументами. В итоге были обработаны аргументы: аргумент1, аргумент, 2, 'аргумент 3'.

```

frigatzero@fedora:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1 lab8-1.asm lab8-1.o
frigatzero@fedora:~/work/arch-pc/lab08$ touch lab8-2.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1 lab8-1.asm lab8-1.o lab8-2.asm
frigatzero@fedora:~/work/arch-pc/lab08$

```

Рис. 2.9: Создание lab8-2.asm

```

lab8-2.asm [----] 15 L: [ 1+13 14/ 21] *(133 / 193b) 0010 0x00A [*] [X]
#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

```

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn10Quit

Рис. 2.10: Код программы lab8-2 из листинга

```

frigatzero@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
2
аргумент 3
frigatzero@fedora:~/work/arch-pc/lab08$

```

Рис. 2.11: Проверка файла lab8-2

8. Создадим файл lab8-3.asm и введем в него текст листинга 8.3. Создадим исполняемый файл и проверим его работу, указав аргументы.

```
frigatzero@fedora:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1 lab8-1.asm lab8-1.o lab8-2 lab8-2.asm lab8-2.o
frigatzero@fedora:~/work/arch-pc/lab08$ touch lab8-3.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1 lab8-1.asm lab8-1.o lab8-2 lab8-2.asm lab8-2.o lab8-3.asm
frigatzero@fedora:~/work/arch-pc/lab08$
```

Рис. 2.12: Создание lab8-3.asm

```
lab8-3.asm [----] 19 L: [ 1+ 3 4/ 30] *(65 / 319b) 0034 0x022 [*][X]
%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 sprint
<----->mov eax, esi
<----->call iprintLF
<----->call quit
```

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit

Рис. 2.13: Код программы lab8-3 из листинга

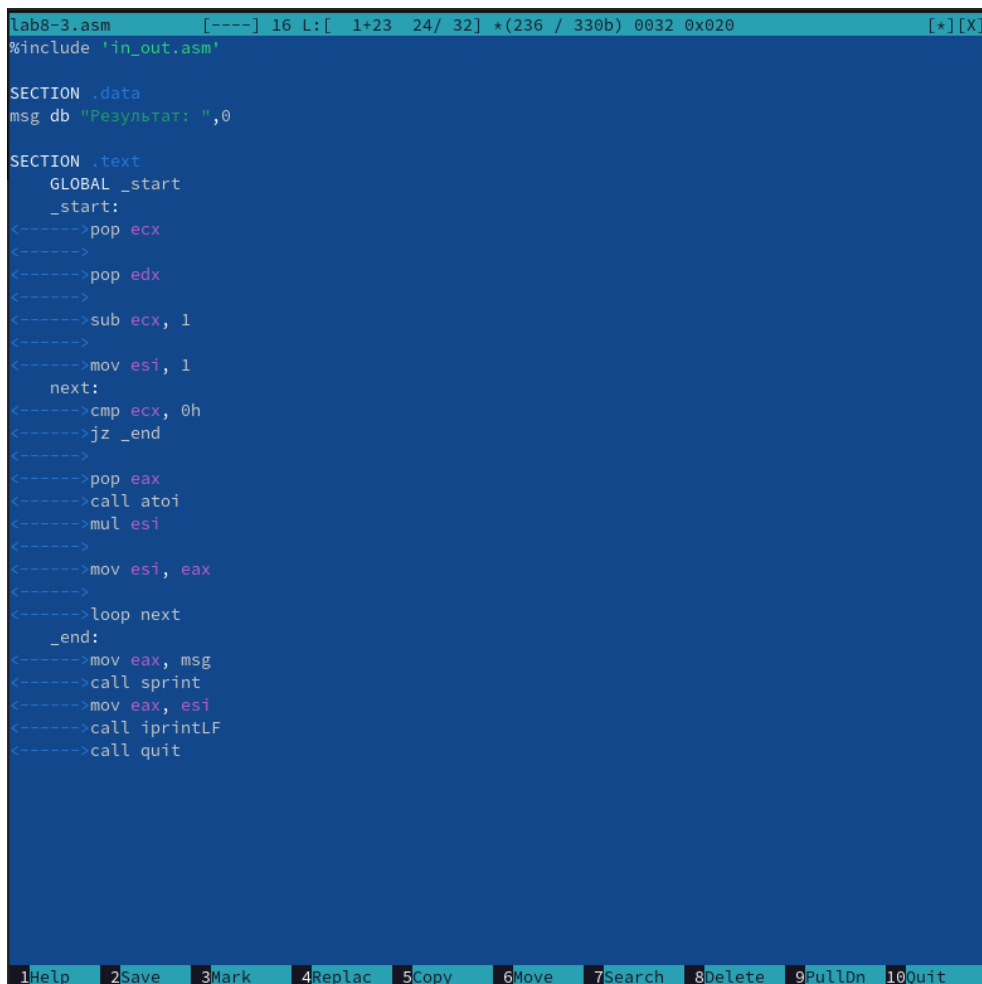
```

frigatzero@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-3
Результат: 0
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
frigatzero@fedora:~/work/arch-pc/lab08$

```

Рис. 2.14: Проверка файла lab8-3

9. Изменим текст программы для вычисления произведения аргументов командной строки.



```

lab8-3.asm [----] 16 L: [ 1+23 24/ 32] *(236 / 330b) 0032 0x020 [X]
#include 'in_out.asm'

SECTION .data
msg db "Результат: ",0

SECTION .text
GLOBAL _start
_start:
<----->pop ecx
<----->
<----->pop edx
<----->
<----->sub ecx, 1
<----->
<----->mov esi, 1
next:
<----->cmp ecx, 0h
<----->jz _end
<----->
<----->pop eax
<----->call atoi
<----->mul esi
<----->
<----->mov esi, eax
<----->
<----->loop next
_end:
<----->mov eax, msg
<----->call sprint
<----->mov eax, esi
<----->call iprintLF
<----->call quit

```

Рис. 2.15: Изменение кода программы lab8-3.asm

```
frigatzero@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 54600
frigatzero@fedora:~/work/arch-pc/lab08$
```

Рис. 2.16: Проверка работы измененного файла lab8-3

3 Ход выполнения заданий для самостоятельной работы

1. Напишем программу для нахождения суммы значений функции для введенных в качестве аргумента программы значений x . Выберем значение функции из №18, табл. 7.5. Создадим исполняемый файл и проверим его работу

```
frigatzero@fedora:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1.asm lab8-2 lab8-2.o lab8-3.asm
lab8-1 lab8-1.o lab8-2.asm lab8-3 lab8-3.o
frigatzero@fedora:~/work/arch-pc/lab08$ touch lab8-4.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ls
in_out.asm lab8-1.asm lab8-2 lab8-2.o lab8-3.asm lab8-4.asm
lab8-1 lab8-1.o lab8-2.asm lab8-3 lab8-3.o
frigatzero@fedora:~/work/arch-pc/lab08$
```

Рис. 3.1: Создаем файл lab8-4.asm

Листинг 3.1 Программа для нахождения суммы значений функции для введенных x .

```
%include 'in_out.asm'
```

```
SECTION .data
```

```
fun db 'Функция:  $f(x) = 17 + 5x$ ',0
res db 'Результат: ',0
```

```
SECTION .bss
```

```
x resb 10
```

```
SECTION .text
```

```
GLOBAL _start
```

```
_start:
```

```
    ;=[Выводим сообщение с функцией]
```

```
    mov eax, fun
```

```
    call sprintf
```

```
    ;
```

```
    ;=[Извлекаем количество аргументов и имя программы с стека]
```

```
    pop ecx      ; Количество аргументов в <ecx>
```

```
    pop edx      ; Имя программы в <edx>
```

```
    sub ecx, 1   ; <ecx - 1> => кол-во аргументов без названия программы
```

```
    ;
```

```
main:
```

```
    ;=[Сравниваем ecx с нулем]
```

```
    cmp ecx, 0   ; Если <ecx = 0>
```

```
    je _exit     ; Завершаем программу
```

```
    ;
```

```
    ;=[Забираем n-ый аргумент с стека]
```

```
    pop eax
```

```

call atoi      ; конвертируем из кода символа в число
;

;=[Находим значение функции]
mov edi, 5     ; <edi = 5>
mul edi        ; <eax = 5x>
add eax, 17    ; <eax = 17+5x>
mov edi, eax   ; <edi = eax>
add [x], edi   ; <x = x + edi>
;

loop main ; <ecx - 1>
_exit:
;=[Выводим сообщение 'Результат: <x>']
mov eax, res
call sprint
mov eax, [x]
call iprintLF
;
call quit

```



```
frigatzero@fedora:~/work/arch-pc/lab08$ nasm -f elf lab8-4.asm
frigatzero@fedora:~/work/arch-pc/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-4 1 1 1 1
Функция:  $f(x) = 17 + 5x$ 
Результат: 88
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-4 1 2 3 4
Функция:  $f(x) = 17 + 5x$ 
Результат: 118
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-4 2 1 10 8
Функция:  $f(x) = 17 + 5x$ 
Результат: 173
frigatzero@fedora:~/work/arch-pc/lab08$ ./lab8-4 5 17 9 100
Функция:  $f(x) = 17 + 5x$ 
Результат: 723
frigatzero@fedora:~/work/arch-pc/lab08$
```

Рис. 3.2: Результат работы программы lab8-4

4 Вывод

После выполнения заданий лабораторной работы и заданий для самостоятельной работы я приобрел навыки написания программ с использованием циклов и обработкой аргументов командой строки, научился работать с структурой данных стек и использовать регистр `esp` для работы с циклами.