

Отчёт по лабораторной работе №7

Дисциплина: архитектура компьютера

Юлдошев Давлатджон Шухратович

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Самостоятельная работа	12
4	Вывод	19

Список иллюстраций

2.1	Создание директории	5
2.2	Редактирование файла	5
2.3	Запуск исполняемого файла	6
2.4	Редактирование файла	6
2.5	Запуск исполняемого файла	6
2.6	Редактирование программы	7
2.7	Создание исполняемого файла	7
2.8	Создание файла	8
2.9	Вставляю текст в файл	8
2.10	Запуск исполняемого файла	8
2.11	Редактирование файла	9
2.12	Файл листинга	9
2.13	Файл листинга	11
2.14	Файл листинга	11
3.1	Создание файла	12
3.2	Редактирование файла	12
3.3	Запуск исполняемого файла	13
3.4	создание файла	15
3.5	ввод программы в файл	16
3.6	запуск исполняемого файла	16

1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга

2 Выполнение лабораторной работы

1

С помощью утилиты `mkdir` создаю директорию `lab07`, перехожу в нее и создаю файл для работы. (рис. [2.1])



```
dsyuldoshev@fedora:~/work/arch-pc/lab07
[dsyuldoshev@fedora:~/work/arch-pc/lab07]$ mkdir -p /work/arch-pc/lab07
[dsyuldoshev@fedora:~/work/arch-pc/lab07]$ cd /work/arch-pc/lab07
[dsyuldoshev@fedora:~/work/arch-pc/lab07]$ touch lab7-1.asm
```

Рис. 2.1: Создание директории

2

Открываю созданный файл `lab7-1.asm`, вставляю в него программу реализации безусловных переходов(рис. [2.2]).



```
GNU nano 7.2 lab7-1.asm
#include "in_out.asm" ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение # 1',0
msg2: DB 'Сообщение # 2',0
msg3: DB 'Сообщение # 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение # 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение # 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение # 3'
end
call quit ; вызов подпрограммы завершения
```

Рис. 2.2: Редактирование файла

3

Создаю исполняемый файл программы и запускаю его (рис. [2.3]). Инструкции `jmp _label2` меняет порядок исполнения инструкций и позволяет выполнить инструкции начиная с метки `_label2`.

```
[dsyuldoshev@fedora lab07]$ nasm -f elf lab7-1.asm
[dsyuldoshev@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[dsyuldoshev@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 3
[dsyuldoshev@fedora lab07]$
```

Рис. 2.3: Запуск исполняемого файла

4

Изменяю текст программы так, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу (рис. [2.4]).

```
GNU nano 7.2 dsyuldoshev@fedora:~/wo
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.4: Редактирование файла

5

Создаю новый исполняемый файл программы и запускаю его (рис. [2.5]). Убеждаюсь в том, программа работает верно.

```
[dsyuldoshev@fedora lab07]$ nasm -f elf lab7-1.asm
[dsyuldoshev@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[dsyuldoshev@fedora lab07]$ nano lab7-1.asm
[dsyuldoshev@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 1
[dsyuldoshev@fedora lab07]$
```

Рис. 2.5: Запуск исполняемого файла

6

Изменяю текст программы, так чтобы вывод происходил в обратном порядке (рис. [2.6]).

```
GNU nano 7.2
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.6: Редактирование программы

7

Создаю исполняемый файл и проверяю работу программы (рис. [2.7]). Программа отработало верно.

```
[dsyuldoshev@fedora lab07]$ nasm -f elf lab7-1.asm
[dsyuldoshev@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[dsyuldoshev@fedora lab07]$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
[dsyuldoshev@fedora lab07]$
```

Рис. 2.7: Создание исполняемого файла

8

Создаю новый файл lab7-2.asm для программы с условным оператором. (рис. [2.8]).

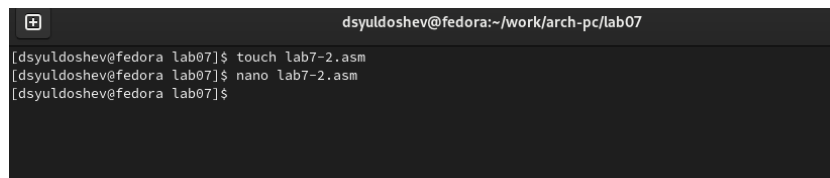


Рис. 2.8: Создание файла

9

Вставляю программу, которая определяет и выводит на экран наибольшее число (рис.[2.9]).

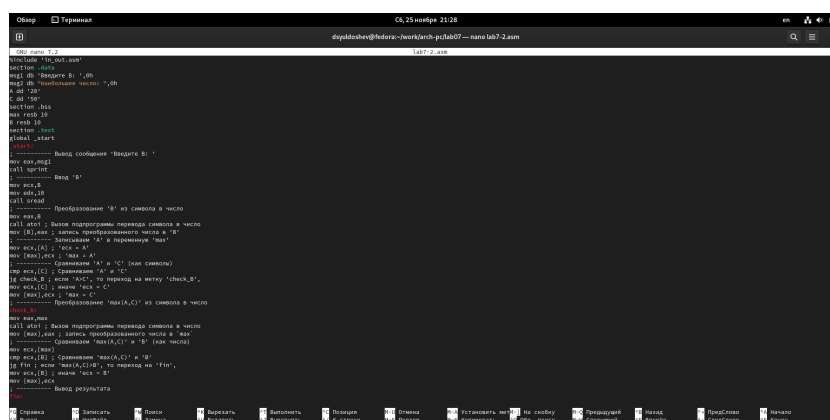


Рис. 2.9: Вставляю текст в файл

10

Создаю и запускаю новый исполняемый файл, проверяю работу программы для разных В, при А=20 и С=50 (рис. [2.10]).

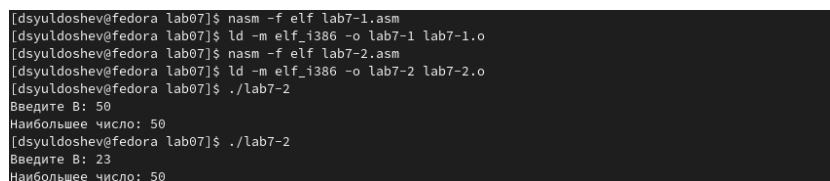
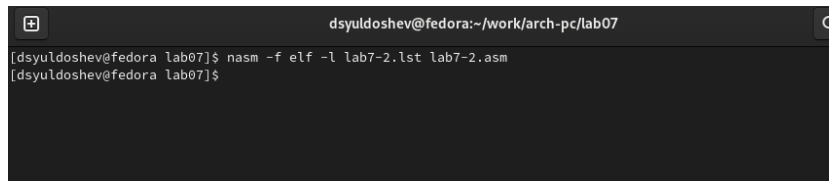


Рис. 2.10: Запуск исполняемого файла

11

Создаю файл листинга для программы в файле lab7-2.asm и открываю его в редакторе mcedit (рис. [2.11]).

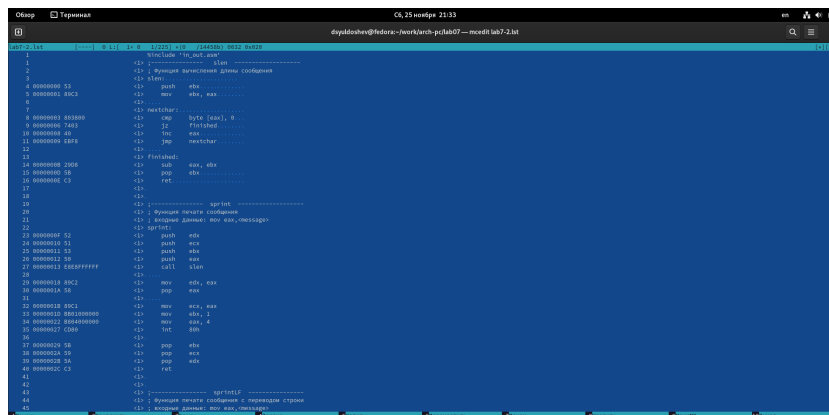


```
dsyuldoshev@fedora:~/work/arch-pc/lab07
[dsyuldoshev@fedora lab07]$ nasm -f elf -l lab7-2.lst lab7-2.asm
[dsyuldoshev@fedora lab07]$
```

Рис. 2.11: Редактирование файла

12

Открываю файл листинга с помощью редактора mcedit. Рассмотрим 9-11 строки: (рис. [2.12]).



```
9 00000006 push ebx
10 00000008 mov ebx, eax
11 00000009 mul ebp
```

Рис. 2.12: Файл листинга

9 строка:

- Первая цифра [9] - это номер строки файла листинга.
- Следующие цифры [00000006] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел.
- следующие числа [7403] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтоу и появляются буквы латинского алфавита.

- следующее [jz finished] - исходный текст программы, которая просто состоит из строк исходной программы вместе с комментариями.

10 строка:

- Первое число [10] - это номер строки файла листинга.
- Следующие цифры [00000008] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел.
- следующие числа [40] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтому и появляются буквы латинского алфавита.
- следующее [inc eax] - исходный текст программы, которая просто состоит из строк исходной программы вместе с комментариями

11 строка:

- Первое число [11] - это номер строки файла листинга.
- Следующие цифры [00000009] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел.
- следующие числа [EBF8] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтому и появляются буквы латинского алфавита.
- следующее [jmp nextchar] - исходный текст программы, которая просто состоит из строк исходной программы вместе с комментариями

13

Открываю файл lab7-2.asm с помощью редактора и Удаляю один операнд в инструкции str. (рис. [2.13]).

```

; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,B; Сравниваем 'max(A,C)' и 'B'
jg fin ; если 'max(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [max],ecx
; ----- Вывод результата
fin:
mov eax, msg2

```

Рис. 2.13: Файл листинга

14

Открываю файл листинга с помощью редактора mscedit и замечая, что в файле листинга появляется ошибка. (рис. [2.14]).

```

36 0000013A 43[00000000] mov [max],eax ; запись преобразованного числа в 'max'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 0000013F 8B0D[00000000] mov ecx,[max]
39 cmp ecx, ; Сравниваем 'max(A,C)' и 'B'
39 ***** error: invalid combination of opcode and operands
40 00000145 7F0C jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 00000147 8B0D[00000000] mov ecx,[B] ; иначе 'ecx = B'
42 0000014D 890D[00000000] mov [max],ecx
43 ; ----- Вывод результата
44 fin:

```

Рис. 2.14: Файл листинга

Отсюда можно сделать вывод, что, если в коде появляется ошибка, то ее описание появится в файле листинга

3 Самостоятельная работа

1

Создаю файл lab7-3.asm с помощью утилиты touch (рис. [3.1]).

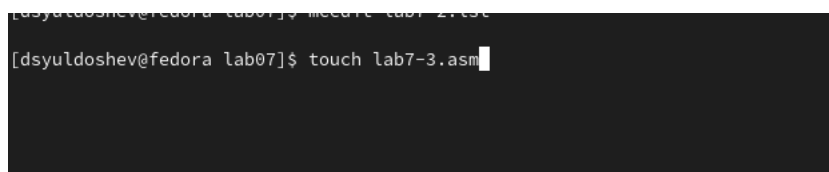


Рис. 3.1: Создание файла

2

Ввожу в созданный файл текст программы для вычисления наибольшего из 3 чисел. Числа беру, учитывая свой вариант из прошлой лабораторной работы. 20 вариант (рис. [3.2]).



Рис. 3.2: Редактирование файла

3

Создаю исполняемый файл и запускаю его (рис. [3.3]).

```
[dsyuldoshev@fedora lab07]$ nasm -f elf lab7-3.asm
[dsyuldoshev@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[dsyuldoshev@fedora lab07]$ ./lab7-3
a = 95
b = 2
c = 61
Наименьшее число: 2
[dsyuldoshev@fedora lab07]$
```

Рис. 3.3: Запуск исполняемого файла

Текст программы

```
%include 'in_out.asm'

section .data
msg1 db ' a = ',0h
msg2 db ' b = ',0h
msg3 db ' c = ',0h
msg4 db "Наименьшее число: ",0h
a dd '62'
b dd '41'
c dd '35'

section .bss
max resb 10

section .text
global _start
_start:
; ----- Вывод всех чисел:
mov eax,msg1
call sprint
mov eax,a
```

```
call atoi
call iprintLF
```

```
mov eax,msg2
call sprint
mov eax,b
call atoi
call iprintLF
```

```
mov eax,msg3
call sprint
mov eax,c
call atoi
call iprintLF
```

```
;-----сравнивание чисел
mov eax,a
call atoi ;перевод символа в число
mov [a],eax ; запись преобразованного числа в b
;----- запись b в переменную max
mov eax,b
call atoi ;перевод символа в число
mov [b],eax ; запись преобразованного числа в b

mov eax,c
call atoi ;перевод символа в число
mov [c],eax ; запись преобразованного числа в b

mov ecx,[a] ;
```

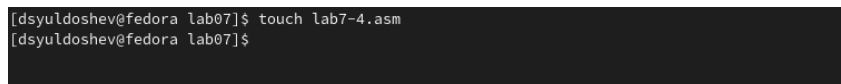
```

mov [max],ecx ;
;-----сравнивание чисел а с
cmp ecx,[c]; if a<c
jl check_b ; то перход на метку
mov ecx,[c] ; else ecx=c
mov [max],ecx ; max=c
;-----метка check_b
check_b:
;-----
mov ecx,[max] ;
cmp ecx,[b] ; ecx<b
jl check_c ;
mov ecx,[b] ;
mov [max],ecx ;
;-----
check_c:
mov eax,msg4 ;
call sprint ;
mov eax,[max];
call iprintLF ;
call quit2

```

4

Создаю новый файл lab7-4 для написания программы второго задания. (рис. [3.4]).



```

[dsyuldoshev@fedora lab07]$ touch lab7-4.asm
[dsyuldoshev@fedora lab07]$

```

Рис. 3.4: создание файла

5

Ввожу в него программу, (рис. [3.5]). в которую ввожу 2 значения x и a , и которая выводит значения функции. Функцию беру из таблицы в соответствии со своим вариантом (Вариант №20)



Рис. 3.5: ввод программы в файл

6

Создаю исполняемый файл и проверяю её выполнение (рис. [3.6]). Программа отработала верно!

```
[dsyuldoshev@fedora lab07]$ nasm -f elf lab7-4.asm
[dsyuldoshev@fedora lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[dsyuldoshev@fedora lab07]$ ./lab7-4
P'PIPuPrPeC,Pu x: 1
P'PIPuPrPeC,Pu a: 2
f(x) = 5
[dsyuldoshev@fedora lab07]$ ./lab7-4
P'PIPuPrPeC,Pu x: 2
P'PIPuPrPeC,Pu a: 1
f(x) = 1
[dsyuldoshev@fedora lab07]$
```

Рис. 3.6: запуск исполняемого файла

Текст программы

```
%include 'in_out.asm'

section .data

msg1 db 'Введите x: ',0h
msg2 db 'Введите a: ',0h
```



```
msg3 db 'f(x) = ',0h
```

```
section .bss
```

```
x resb 10
```

```
a resb 10
```

```
section .text
```

```
global _start
```

```
_start:
```

```
mov eax,msg1
```

```
call sprint
```

```
mov ecx,x
```

```
mov edx,10
```

```
call sread
```

```
mov eax,x
```

```
;-----
```

```
call atoi
```

```
mov [x],eax
```

```
;-----
```

```
mov eax,msg2
```

```
call sprint
```

```
mov ecx,a
```

```
mov edx,10
```

```
call sread
```

```
mov eax,a ;
```

```
call atoi
```

```
mov [a],eax ;
```

```
;-----
```

```
mov ecx,[x]
cmp ecx,[a] ;x<a
jl _check ;
mov ecx,[x]
add ecx,10
jmp _end
_check:
mov ecx,[a];
add ecx,10
_end:
mov eax,msg3 ;
call sprint ;
mov eax,ecx ;
call iprintLF;
call quit ;
```

4 Вывод

При выполнении данной лабораторной работы я освоил инструкции условного и безусловного вывода и ознакомился с структурой файла листинга.ы