

Отчёта по лабораторной работе

Лабораторная работа №8

Дикач Анна Олеговна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Задание для самостоятельной работы	9
4	Выводы	11

Список иллюстраций

2.1	создание каталога, переход в него, создание файла	6
2.2	пример работы программы	6
2.3	пример работы программы	7
2.4	пример работы программы	8
2.5	результат запуска программы	8

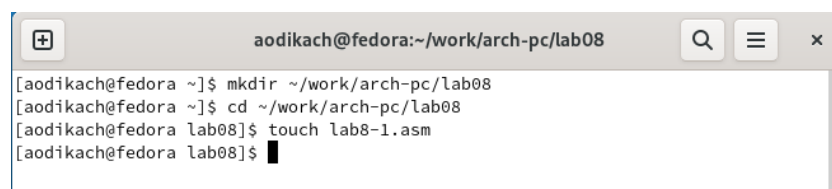
Список таблиц

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

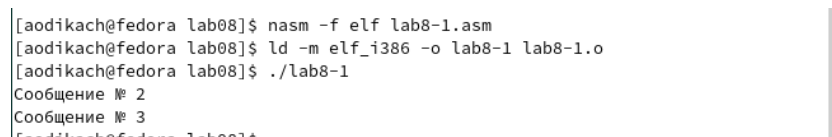
1. создаю каталог для программ лабораторной работы, перехожу в него и создаю файл (рис. 2.1)



```
aodikach@fedora:~/work/arch-pc/lab08
[aodikach@fedora ~]$ mkdir ~/work/arch-pc/lab08
[aodikach@fedora ~]$ cd ~/work/arch-pc/lab08
[aodikach@fedora lab08]$ touch lab8-1.asm
[aodikach@fedora lab08]$
```

Рис. 2.1: создание каталога, переход в него, создание файла

2. ввожу текст программы, создаю файл, запускаю его (рис. 2.2)



```
[aodikach@fedora lab08]$ nasm -f elf lab8-1.asm
[aodikach@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[aodikach@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 3
```

Рис. 2.2: пример работы программы

3. ввожу текст другой программы, создаю файл, запускаю его (рис. ??). вношу изменения чтобы изменить выводимое сообщение (рис. ??)(рис. ??)

```
[aodikach@fedora lab08]$ mc
[aodikach@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 1

[aodikach@fedora lab08]$ nasm -f elf lab8-1.asm
[aodikach@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[aodikach@fedora lab08]$ ./lab8-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
```

```
aodikach@fedora:~
GNU nano 6.0 /home/aodikach/work/arch-pc/lab8-1.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 _label3

_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
jmp _end

_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
jmp _label2

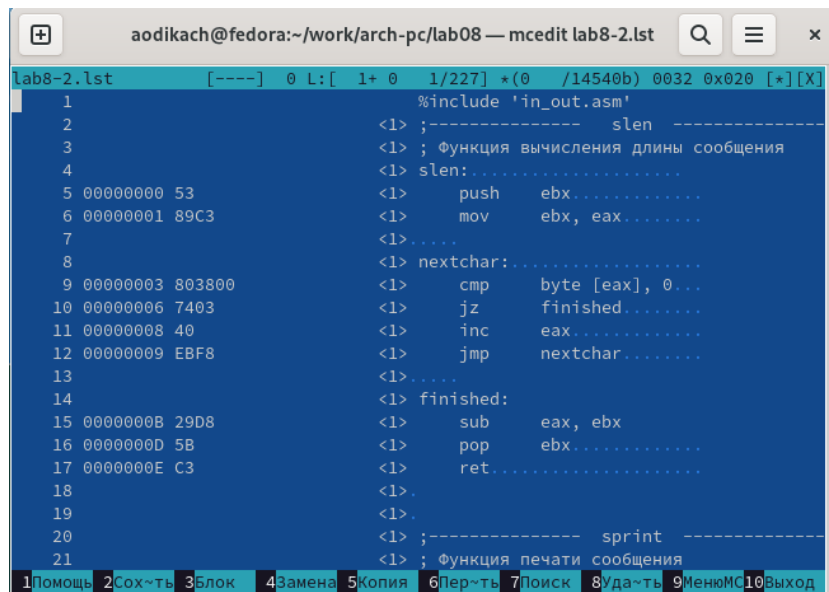
_end:
call quit ; вызов подпрограммы завершения
```

- создаю файл lab8-2.asm в каталоге ~/work/arch-pc/lab08. внимательно изучаю текст программы из листинга 8.3 и ввожу в lab8-2.asm. (рис. 2.3)

```
[aodikach@fedora lab08]$ nasm -f elf lab8-2.asm
[aodikach@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[aodikach@fedora lab08]$ ./lab8-2
Введите В: 3
Наибольшее число: 50
```

Рис. 2.3: пример работы программы

- открываю файл листинга lab8-2.lst с помощью любого текстового редактора, например mcedit (рис. 2.4)



```
lab8-2.lst [----] 0 L: [ 1+ 0 1/227] *(0 /14540b) 0032 0x020 [*][X]
1                                     %include 'in_out.asm'
2                                     <1> ;----- slen -----
3                                     <1> ; Функция вычисления длины сообщения
4                                     <1> slen:.....
5 00000000 53                         <1> push    ebx.....
6 00000001 89C3                       <1> mov     ebx, eax.....
7                                     <1>.....
8                                     <1> nextchar:.....
9 00000003 803800                     <1> cmp     byte [eax], 0...
10 00000006 7403                      <1> jz      finished.....
11 00000008 40                        <1> inc     eax.....
12 00000009 EBF8                      <1> jmp     nextchar.....
13                                     <1>.....
14                                     <1> finished:
15 0000000B 29D8                      <1> sub     eax, ebx
16 0000000D 5B                        <1> pop     ebx.....
17 0000000E C3                       <1> ret.....
18                                     <1>.
19                                     <1>.
20                                     <1> ;----- sprint -----
21                                     <1> ; Функция печати сообщения
```

Рис. 2.4: пример работы программы

5. открываю файл с программой lab8-2.asm и в любой инструкции с двумя операндами удаляю один операнд. выполняю трансляцию с получением файла ластинга (рис. 2.5). создаётся файл lab8-2.lst, в терминале добавляется сообщение об ошибке в тексте программы

```
[aodikach@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:30: error: invalid combination of opcode and operands
[aodikach@fedora lab08]$
```

Рис. 2.5: результат запуска программы

3 Задание для самостоятельной работы

1. пишу программу для вычисления наименьшего числа и проверяю с помощью чисел из 10 варианта (рис. ??) (рис. ??)

```
mc [aodikach@fedora]:~/work/arch-pc/lab08
GNU nano 6.0 /home/aodikach/work/arch-pc/lab08/lab8-3.asm

section .data
msg2 db "Наименьшее число: ",0h
A dd 41
B dd 62
C dd 35

section .bss
min resb 1000
section .text
global _start
_start:
; ----- Записываем 'A' в переменную 'min'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'min = A'
; ----- Сравниваем A и C (как числа)
cmp ecx,[C] ; Сравниваем A и C
jl check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'min = C'

check_B:
mov eax,min

; ----- Сравниваем 'min(A,C)' и 'B' (как числа)

mov ecx,[min]
cmp ecx,[B] ; Сравниваем 'min(A,C)' и 'B'
jl fin ; если 'min(A,C) > B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [min],ecx

; ----- Вывод результата
fin:
mov eax, msg2
call sprint ; Вывод сообщения 'Наименьшее число: '
mov eax,[min]
call iprintLF ; Вывод 'min(A,B,C)'
call quit ; Выход
```

```
[aodikach@fedora lab08]$ ./lab8-3
Наименьшее число: 35
[aodikach@fedora lab08]$
```

2. пишу программу, которая вычисляет значение функций для введенных с клавиатуры данных из варианта 10. создаю исполняемый файл и проверяю его работу (рис. ??) (рис. ??)

```
aodikach@fedora:~/work/arch-pc/lab08
GNU nano 6.0 /home/aodikach/work/arch-pc/lab08/lab8-4.asm
msg2 db 'Введите a: ',0h

section .bss
x resb 10
a resb 10

section .text
global _start
_start:

mov eax, msg1
call sprint
mov ecx, x
mov edx, 20
call sread
mov eax, x
call atoi
mov [x], eax

mov eax, msg2
call sprint
mov ecx, a
mov edx, 20
call sread
mov eax, a
call atoi

mov ecx, 2

cmp [x], ecx
jg more

cmp [x], ecx
jle less

more:
mov eax, [x]
sub eax, 2
call fin

less:
mov ebx, 3;
```

```
[aodikach@fedora lab08]$ nasm -f elf lab8-4.asm
[aodikach@fedora lab08]$ ld -m elf_i386 -o lab8-4 lab8-4.o
[aodikach@fedora lab08]$ ./lab8-4
Введите x: 3
Введите a: 0
1
[aodikach@fedora lab08]$ ./lab8-4
Введите x: 1
Введите a: 2
6
[aodikach@fedora lab08]$ mc
[aodikach@fedora lab08]$
```

4 Выводы

изучила команды условного и безусловного перехода. приобрела навыки написания программ с использованием переходов. познакомилась с назначением и структурой файла листинга.