

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7**

дисциплина: *Архитектура компьютера*

Студент: Кадыков Максим Владимирович

Группа: НКАбд-07-25

**МОСКВА**

2025 г.

# **Оглавление**

1. Цель работы.....	3
2. Задание.....	4
3. Теоретическое введение.....	5
4. Выполнение лабораторной работы.....	6
4.1 Реализация в NASM.....	6
4.2 Изучение структуры файла листинга.....	12
4.3 Задания для самостоятельной работы.....	15
5. Выводы.....	20
Список литературы.....	21

## **1 Цель работы**

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

## **2 Задание**

1. Реализация переходов в NASM
2. Изучение структуры файлов листинга
3. Самостоятельное написание программ по материалам лабораторной работы.

### **3 Теоретическое введение**

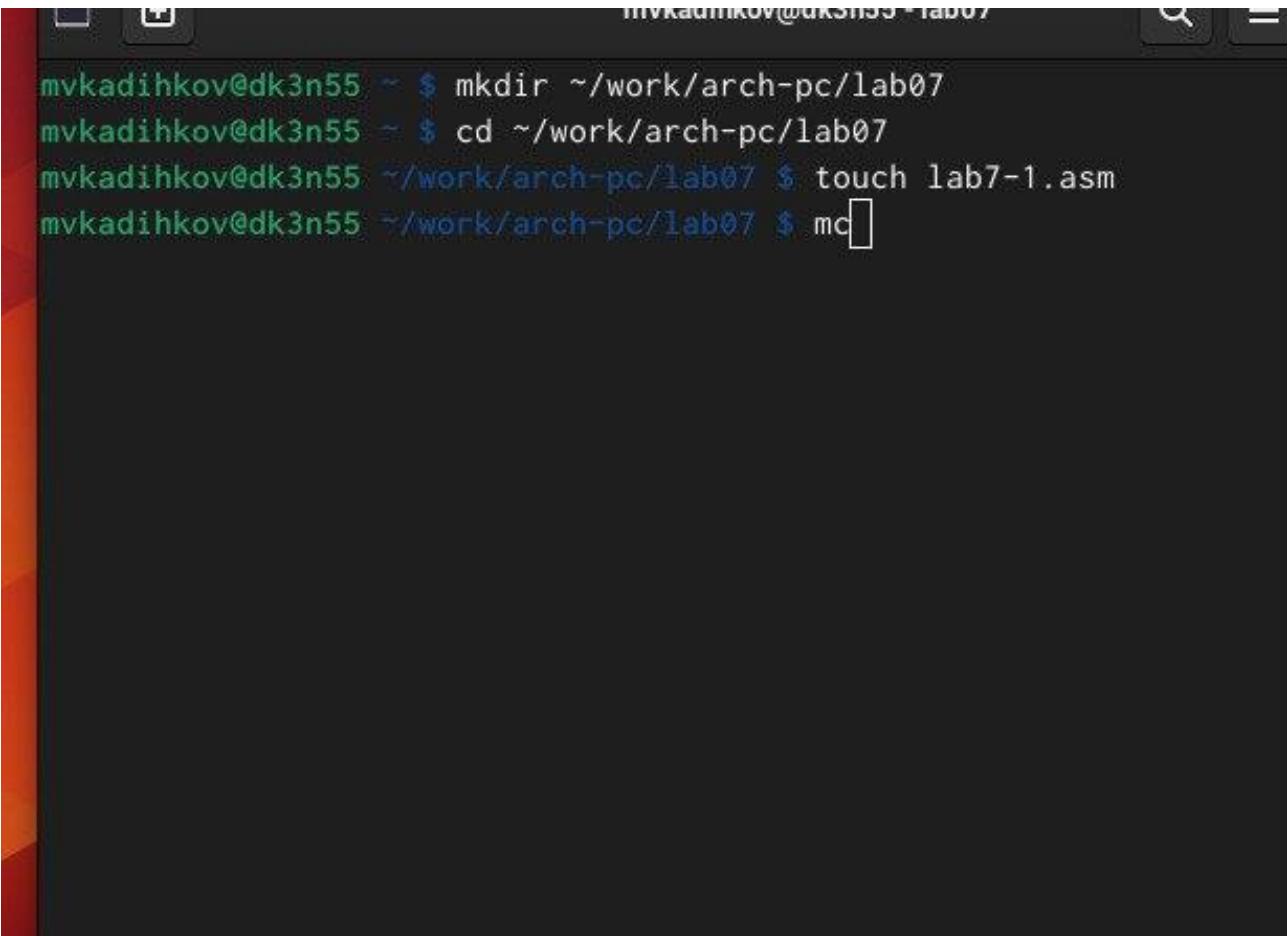
Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов:

- условный переход – выполнение или не выполнение перехода в определенную точку программы в зависимости от проверки условия.
- безусловный переход – выполнение передачи управления в определенную точку программы без каких-либо условий.

## 4 Выполнение лабораторной работы

### 4.1 Реализация переходов в NASM

Создаю каталог для программ лабораторной работы №7.



The screenshot shows a terminal window with a dark background and light-colored text. The window title is "mvkadihkov@dk3n55:~/work/arch-pc/lab07". The command history is as follows:

```
mvkadihkov@dk3n55 ~ $ mkdir ~/work/arch-pc/lab07
mvkadihkov@dk3n55 ~ $ cd ~/work/arch-pc/lab07
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ touch lab7-1.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ mc
```

The terminal window has a red vertical decorative bar on the left and a search icon in the top right corner.

**Рисунок 1 Создание каталога и файла для программы**  
Копирую код из листинга в файл будущей программы.

```
%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:
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
_end:
call quit
```

Рисунок 2 Сохранение программы

При запуске программы я убедился в том, что неусловный переход действительно изменяет порядок выполнения инструкций.

```
mvkadihkoy@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
mvkadihkoy@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
mvkadihkoy@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-1
Сообщение № 2
Сообщение № 3
mvkadihkoy@dk3n55 ~/work/arch-pc/lab07 $ █
```

Рисунок 3 Запуск программы

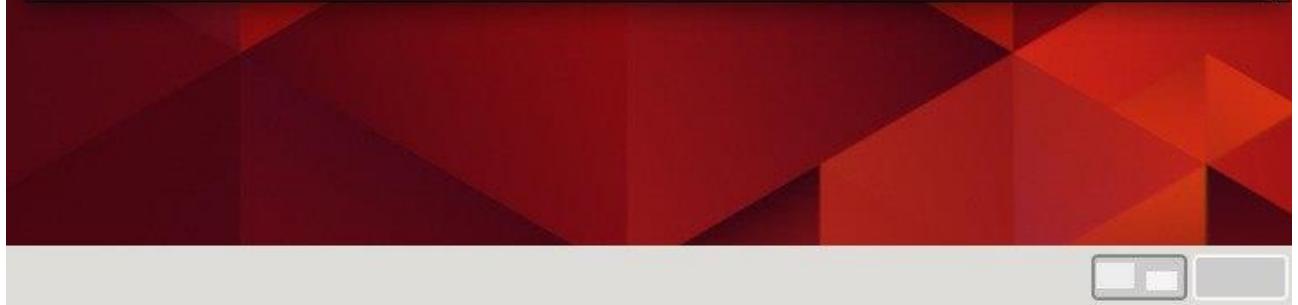
Теперь изменяю текст программы так, чтобы все три сообщения вывелись в обратном порядке.

```
%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:
_label1:
jmp _label2
_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'
_end:
call quit
```

Рисунок 4 Изменение программы

Работа выполнена корректно, программа в нужном мне порядке выводит сообщения.

```
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-1.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-1 lab7-1.o
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $
```



**Рисунок 5 Проверка изменений**

Создаю новый рабочий файл и вставляю в него код из следующего листинга.

```
%include 'in_out.asm'
section .data
msg1 db 'Введите В: ',0h
msg2 db "Наибольшее число: ",0h
A dd '20'
C dd '50'
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
; ----- Вывод сообщения 'Введите В: '
mov eax,msg1
call sprint
; ----- Ввод 'В'
mov ecx,B
mov edx,10
call sread
; ----- Преобразование 'В' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'В'
; ----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в `max`
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
```

**Рисунок 6 Сохранение новой программы**

Программа выводит значение переменной с максимальным значением, проверяю работу программы с разными входными данными.

```
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ touch lab7-2.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ mc

mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-2
bash: ./lab7-2: Нет такого файла или каталога
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-2.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-2 lab7-2.o
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-2
Введите В: 5
Наибольшее число: 50
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-2
Введите В: 100
Наибольшее число: 100
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-2
Введите В: 78
Наибольшее число: 78
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $
```

Рисунок 7 Проверка программы из листинга

## 4.2 Изучение структуры файла листинга

Создаю файл листинга с помощью флага -l команды nasm и открываю его с помощью текстового редактора mcedit.

```
lab7-2.lst      [---] 0 L:[ 1+ 0 1/225] <(0 /14458b) 0032 0x020
1          %include 'in_out.asm'
1          ;----- slen -----
2          ; функция вычисления длины сообщения
3
4 00000000 53    ;<1> push  ebx
5 00000001 89C3  ;<1> mov   ebx, eax
6
7          ;<1>
8 00000002 893000 ;<1> cmp   byte [eax], 0
9 00000006 7403  ;<1> jz    finished
10 00000008 40   ;<1> inc   eax
11 00000009 EBF8  ;<1> jmp   nextchar
12
13          ;<1>
14 0000000B 29D8  ;<1> sub   eax, ebx
15 0000000D 58   ;<1> pop   ebx
16 0000000E C3   ;<1> ret
17
18          ;<1>
19          ;----- sprint -----
20          ; функция печати сообщения
21          ; входные данные: mov eax,<message>
22          ;<1> sprint:
23 0000000F 52   ;<1> push  edx
24 00000010 51   ;<1> push  ecx
25 00000011 53   ;<1> push  ebx
26 00000012 50   ;<1> push  eax
27 00000013 E8E8FFFF  ;<1> call   slen
28
29 00000018 89C2  ;<1> mov   edx, eax
30 0000001A 58   ;<1> pop   eax
31
32 0000001B 89C1  ;<1> mov   ecx, eax
33 0000001D B801000000  ;<1> mov   ebx, 1
34 00000022 B804000000  ;<1> mov   eax, 4
35 00000027 CD80  ;<1> int   80h
36
37 00000029 58   ;<1> pop   ebx
38 0000002A 59   ;<1> pop   ecx
39 0000002B 5A   ;<1> pop   edx
40 0000002C C3   ;<1> ret
```

Рисунок 8 Проверка файла листинга

Первое значение в файле листинга - номер строки, и он может вовсе не совпадать с номером строки изначального файла. Второе вхождение - адрес, смещение машинного кода относительно начала текущего сегмента, затем непосредственно идет сам машинный код, а заключает строку исходный текст программы с комментариями.

Удаляю один operand из случайной инструкции, чтобы проверить поведение файла листинга в дальнейшем.

```
/afs/.dk.sci.pfu.edu.ru/home/m/v/mvkadihkov/work/arch-pc/lab07/lab7-2.asm
%include 'in_out.asm'
section .data
msg1 db 'Введите В: ',0h
msg2 db "Наибольшее число: ",0h
A dd '20'
C dd '50'
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
; ----- Вывод сообщения 'Введите В: '
mov eax,msg1
call sprint
; ----- Ввод ''
mov ecx,
mov edx,10
call sread
; ----- Преобразование 'В' из символа в число
mov ebx
```

Рисунок 9 Удаление операнда из программы

В новом файле листинга показывает ошибку, которая возникла при попытке трансляции файла. Никакие выходные файлы при этом помимо файла листинга не создаются.

```
mvkadihkov@dk3n55 ~/work/arch-pe/lab07 $ nasm -f elf -l lab7-2.asm
nasm: fatal: no input file specified
Type nasm -h for help.
mvkadihkov@dk3n55 ~/work/arch-pe/lab07 $
```

Рисунок 10 Просмотр ошибки в файле листинга

#### 4.3 Задания для самостоятельной работы

Буду выполнять вариант 20, так как в 6 лабораторной мне выдали 20 вариант. Возвращаю операнд к функции в программе и изменяю ее так, чтобы она выводила переменную с наименьшим значением.

```
/afs/.dk.sci.pfu.edu.ru/home/m/v/mvkadihkov/work/arch-pc/lab07/lab7-sm.asm
%include 'in_out.asm'      ; Подключение внешнего файла

SECTION .data
    a DD 95                ; Переменная a = 95
    b DD 2                  ; Переменная b = 2
    c DD 61                 ; Переменная c = 61

    msg_min DB 'Наименьшее число: ',0
    newline DB 10,0

SECTION .text
GLOBAL _start
_start:
    ; Находим минимум среди a, b, c
    mov eax, [a]            ; Предполагаем, что a - минимальное

    ; Сравниваем с b
    mov ebx, [b]
    cmp eax, ebx
    jl check_c              ; Если a < b, проверяем c
```

Рисунок 11 Первая программа самостоятельной работы

Проверяю корректность написания первой программы.

The screenshot shows a terminal window with the following session:

```
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ touch lab7-sm.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ mc
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-sm.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-sm lab7-sm.o
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-sm
Наименьшее число: 2
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $
```

**Рисунок 12 Проверка работы первой программы**

Пишу программу, которая будет вычислять значение заданной функции согласно моему варианту для введенных с клавиатурных переменных а и х.

```
/afs/.dk.sci.pfu.edu.ru/home/m/v/mvkadihkov/work/arch-pc/lab07/lab7-sm2.asm
%include 'in_out.asm'

SECTION .data
msg_x db "Введите x: ",0
msg_a db "Введите a: ",0
msg_result db "f(x) = ",0
newline db 0xA,0

SECTION .bss
x resd 1
a resd 1
result resd 1

SECTION .text
global _start

_start:
; Ввод переменной x
mov eax, msg_x
call sprint
mov ecx, x
```

Рисунок 13 Вторая программа самостоятельной работы

Транслирую и компоную файл, запускаю и проверяю работу программы для различных значений а и х.

```
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf -l lab7-sm2.asm
nasm: fatal: no input file specified
Type nasm -h for help.
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-sm2.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-sm2 lab7-sm2.o
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-sm2
Введите x: 1
Введите a: 2
f(x) = 5
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ nasm -f elf lab7-sm2.asm
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ld -m elf_i386 -o lab7-sm2 lab7-sm2.o
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ ./lab7-sm2
Введите x: 2
Введите a: 1
f(x) = 1
mvkadihkov@dk3n55 ~/work/arch-pc/lab07 $ 
```

Рисунок 14 Проверка работы второй программы

## **5 Выводы**

При выполнении лабораторной работы я изучил команды условных и безусловных переходов, а также приобрел навыки написания программ с использованием переходов, познакомился с назначением и структурой файлов листинга.

## **Список литературы**

- 1.Курс на ТУИС
- 2.Лабораторная работа №7
- 3.Программирование на языке ассемблера NASM Столяров А. В.