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: Метвалли Ахмед Фарг Набеех

Группа: НПИбд-02-23

МОСКВА 2023г

Содержание

Список иллюстраций

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. Создадим файл lab7-1.asm, запишем код программы и проверим его работу:

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 l
ab7-1.o
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.1: Работа программы lab7-1

2. Создадим файл lab7-2.asm, запишем код программы и также проверим его работу:

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 5
Наибольшее число: 50
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 60
Наибольшее число: 60
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.2: Работа программы lab7-2

3. Изменим текст программы, изменив инструкции jmp и получим следующее:

```
 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
22                                  <1> ; входные данные: mov eax,<message>
23                                  <1> sprint:
24 0000000F 52                      <1>     push    edx
25 00000010 51                      <1>     push    ecx
26 00000011 53                      <1>     push    ebx
27 00000012 50                      <1>     push    eax
28 00000013 E8E8FFFFFF              <1>     call    slen
29                                  <1>
30 00000018 89C2                    <1>     mov     edx, eax
31 0000001A 58                      <1>     pop     eax
32                                  <1>
33 0000001B 89C1                    <1>     mov     ecx, eax
```

Рис. 2.3: Измененный код

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ls
in_out.asm  lab7-1  lab7-1.asm  lab7-1.o  lab7-2.asm  lab7-2.lst
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.4: Работа измененной программы

4. Откроем файл с программой lab7-1.asm и в любой инструкции с двумя операндами удалим один, выполним трансляцию с получением файла листинга (nasm -f elf -l lab7-2.lst lab7-2.asm):

```
 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  ------------------
```

Рис. 2.5: Ошибка в программе

5. Создадим файл lab7-3.asm, запишем код программы и проверим его работу на разных значениях B:

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-3
Введите B: 5
Наибольшее число: 58
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.6: Результат выполнения программы

3.Самостоятельная работа

1. Напишем программу нахождения наименьшей из 3 целочисленных переменных и запустим её:

```
%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 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 ; вызов подпрограммы завершения
```

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 4
F(x)=5
```

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 4
F(x)=5
```

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 1
Введите a: 4
F(x)=5
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 1
Введите a: 4
F(x)=5
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
```

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
```

```
ahmad-farg@ahmadfarg-VirtualBox:~$ mkdir ~/work/arch-pc/lab07
ahmad-farg@ahmadfarg-VirtualBox:~$ cd ~/work/arch-pc/lab07
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ touch lab7-1.asm
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

```
ahmad-farg@ahmadfarg-VirtualBox:~$ mkdir ~/work/arch-pc/lab07
ahmad-farg@ahmadfarg-VirtualBox:~$ cd ~/work/arch-pc/lab07
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ touch lab7-1.asm
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$
```

```
%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 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 ; вызов подпрограммы завершения
```

Рис. 3.1: Код программы

Рис. 3.2:Результат выполнения программы

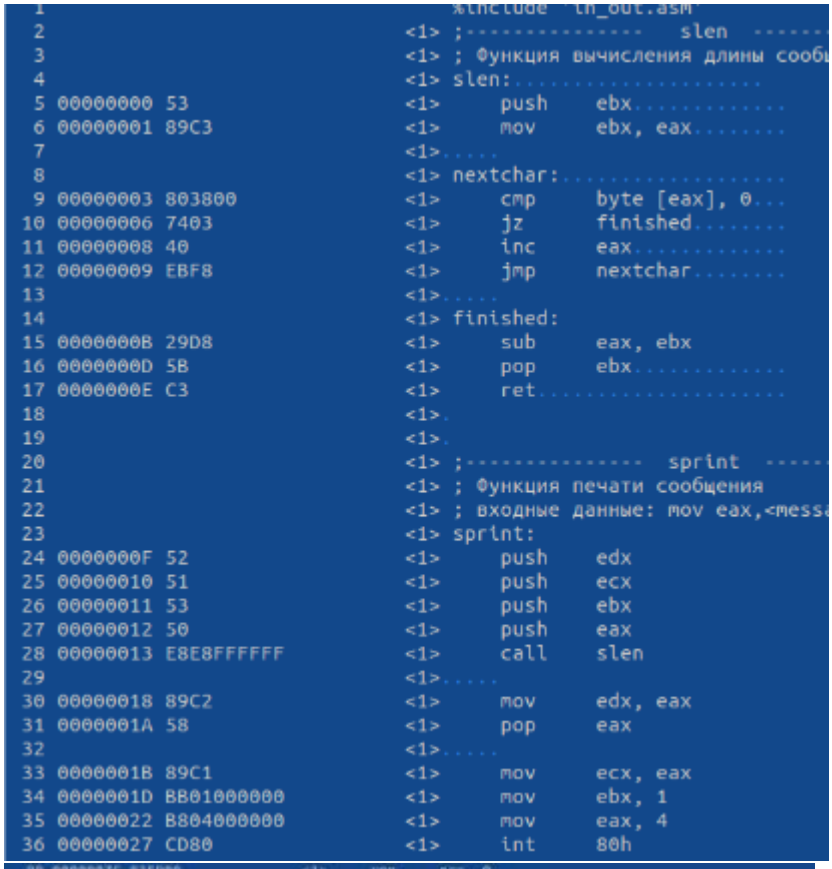
```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ touch lab7-4.asm
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1 ; Вывод на 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
```

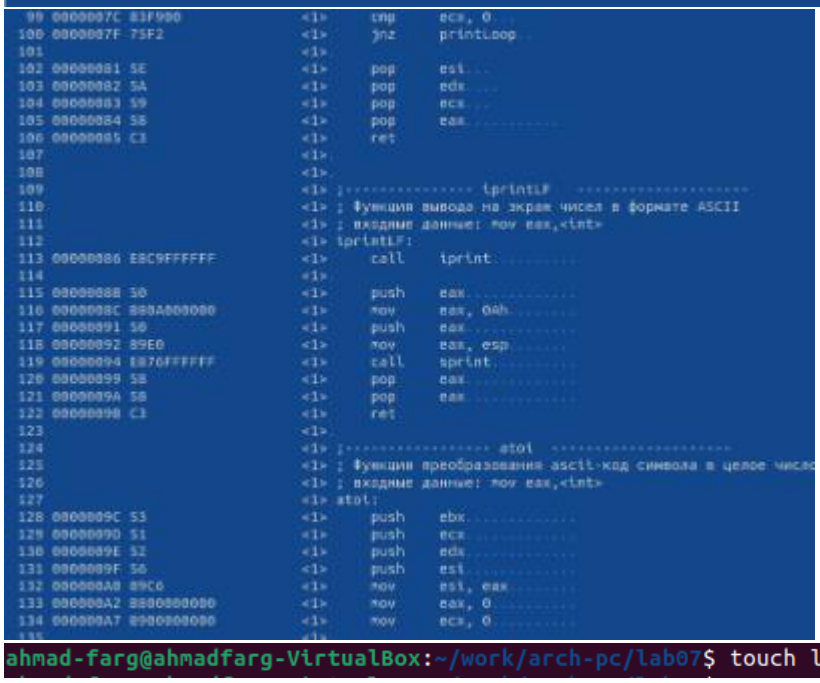
```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, 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
```

```
ahmad-farg@ahmadfarg-VirtualBox:~/work/arch-pc/lab07$ touch lab7-2.asm
```

```
  GNU nano 6.2                                    /home
%include 'in_out.asm'
section .data
msg1 db 'Введите B: ',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 B: '
mov eax,msg1
call sprint
; ---------- Ввод 'B'
mov ecx,B
mov edx,10
call sread
; ---------- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в чис
mov [B],eax ; запись преобразованного числа в 'B'
; ---------- 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'
```

```
; ---------- Ввод 'B'
mov ecx,B
mov edx,
call sread
```





2. Напишем программу для нахождения значения заданной функции из введенных с клавиатуры значений

Рис. 3.3: Код программы

Рис. 3.4: Результат выполнения программы

```
 GNU nano 6.2
%include 'in_out.asm'
section .data
msg1 DB 'Введите x: ',0
msg2 DB "Введите a: ",0
otv: DB 'F(x)=',oh
section .bss
x:RESB 80
a:RESB 80
res:RESB 80
section .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx,x
mov edx,80
call sread
mov eax,x
call atoi
mov[x],eax
mov eax,msg2
call sprint
mov ecx,a
mov edx,80
call sread
mov eax,a
call atoi
mov[a],eax
mov eax, [x]
cmp eax, 3
je x_is_3
mov eax, [a]
add eax, 1
jmp calc_res
x_is_3:
mov eax, [x]
```

```
 GNU nano 6.2
%include 'in_out.asm'
section .data
msg1 DB 'Введите x: ',0
msg2 DB "Введите a: ",0
otv: DB 'F(x)=',oh
section .bss
x:RESB 80
a:RESB 80
res:RESB 80
section .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx,x
mov edx,80
call sread
mov eax,x
call atoi
mov[x],eax
mov eax,msg2
call sprint
mov ecx,a
mov edx,80
call sread
mov eax,a
call atoi
mov[a],eax
mov eax, [x]
cmp eax, 3
je x_is_3
mov eax, [a]
add eax, 1
jmp calc_res
x_is_3:
mov eax, [x]
```

4 . Выводы

Я изучил команды условного и безусловного переходов и научился писать программы с использованием этих переходов .