

Лабораторная работа 7

Команды безусловного и условного переходов в Nasm.
Программирование ветвлений.

Мазуркевич Анастасия

Содержание

Цель работы	5
Задание	6
Выполнение лабораторной работы	7
Задание для самостоятельной работы	14
Выводы	18
Список литературы	19

Список иллюстраций

1	Создаем каталог и файл	7
2	Вводим листинг	8
3	создаем и запускаем	8
4	Изменяем текст в соответствии с листингом	9
5	Создаем файл и проверяем, все верно	9
6	Изменяем	10
7	Проверяем	10
8	Создаем файл и вводим листинг	11
9	Вводим различные В	11
10	Создаем	12
11	Открываем	12
12	Открываем и удаляем	13
13	Транслируем	13
14	Изучаем	14
15	Создаем	14
16	Прописываем программу	15
17	Создаем файл, работает верно	15
18	Пишем программу	16
19	Вводим значения, ответ верный	16
20	Вводим значения, ответ верный	17

Список таблиц

Цель работы

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

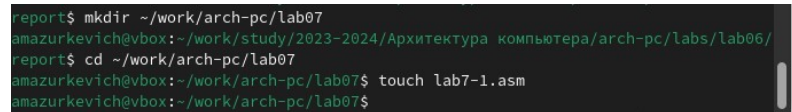
Задание

Написать программы для решения заданий

Выполнение лабораторной работы

##Реализация переходов в NASM

Создайте каталог для программ лабораторной работы № 7. Перейдите в него и создайте файл lab7-1.asm(рис. [-@fig:001]).



```
report$ mkdir ~/work/arch-pc/lab07
amazurkevich@vbox:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/
report$ cd ~/work/arch-pc/lab07
amazurkevich@vbox:~/work/arch-pc/lab07$ touch lab7-1.asm
amazurkevich@vbox:~/work/arch-pc/lab07$
```

Рис. 1: Создаем каталог и файл

Введите в файл lab7-1.asm текст программы из листинга 7.1.(рис. [-@fig:002]).

```

1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label2
10 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 _label2:
14 mov eax, msg2 ; Вывод на экран строки
15 call sprintf ; 'Сообщение № 2'
16 _label3:
17 mov eax, msg3 ; Вывод на экран строки
18 call sprintf ; 'Сообщение № 3'
19 _end:
20 call quit ; вызов подпрограммы завершени

```

Рис. 2: Вводим листинг

Создайте исполняемый файл и запустите его(рис. [-@fig:003]).

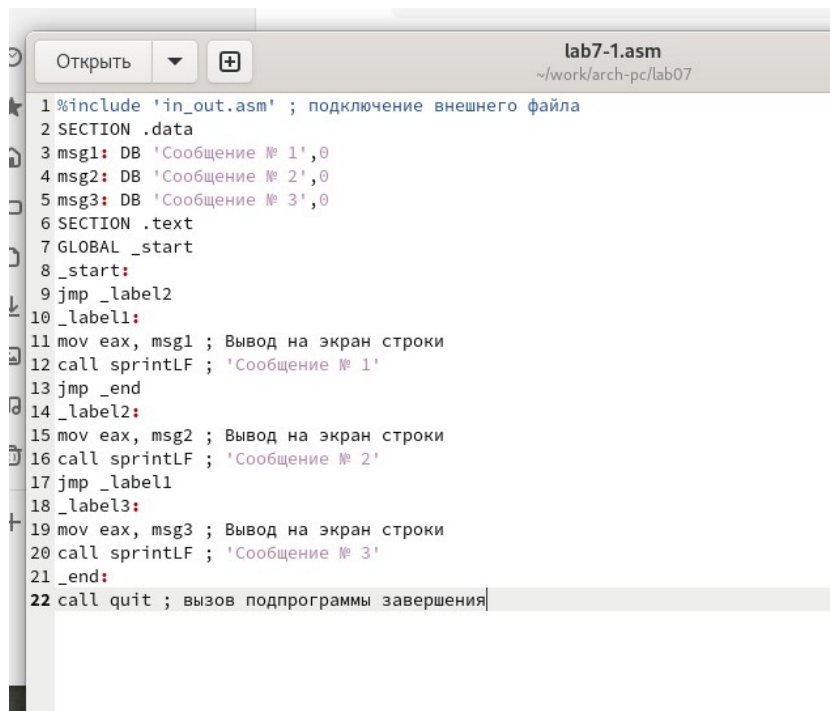
```

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

```

Рис. 3: создаем и запускаем

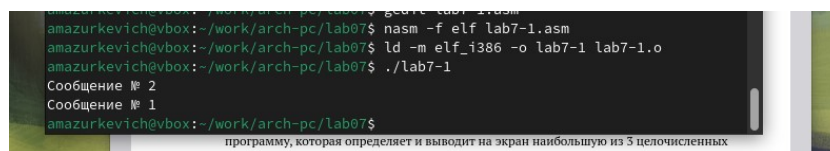
Измените текст программы в соответствии с листингом 7.2.(рис. [-@fig:004]).



```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label2
10 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 jmp _end
14 _label2:
15 mov eax, msg2 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 2'
17 jmp _label1
18 _label3:
19 mov eax, msg3 ; Вывод на экран строки
20 call sprintf ; 'Сообщение № 3'
21 _end:
22 call quit ; вызов подпрограммы завершения
```

Рис. 4: Изменяем текст в соответствии с листингом

Создайте исполняемый файл и проверьте его работу.(рис. [-@fig:005]).



```
amazurkevich@vbox: ~/work/arch-pc/lab07$ gcc -c lab7-1.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
amazurkevich@vbox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
amazurkevich@vbox:~/work/arch-pc/lab07$
```

Рис. 5: Создаем файл и проверяем, все верно

Измените текст программы добавив или изменив инструкции jmp(рис. [-@fig:006]).

```

1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label3
10 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 jmp _end
14 _label2:
15 mov eax, msg2 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 2'
17 jmp _label1
18 _label3:
19 mov eax, msg3 ; Вывод на экран строки
20 call sprintf ; 'Сообщение № 3'
21 jmp _label2
22 _end:
23 call quit ; вызов подпрограммы завершения

```

Рис. 6: Изменяем

Проверьте работу файла(рис. [-@fig:007]).

```

amazurkevich@vbox:~/work/arch-pc/lab07$ gedit lab7-1.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
amazurkevich@vbox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
amazurkevich@vbox:~/work/arch-pc/lab07$

```

Рис. 7: Проверяем

Создайте файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. Внимательно изучите текст программы из листинга 7.3 и введите в lab7-2.asm.(рис. [-@fig:008]).

```

13 ; ----- Вывод сообщения 'Введите B: '
14 mov eax,msg1
15 call sprint
16 ; ----- Ввод 'B'
17 mov ecx,B
18 mov edx,10
19 call sread
20 ; ----- Преобразование 'B' из символа в число
21 mov eax,B
22 call atoi ; Вызов подпрограммы перевода символа в число
23 mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
25 mov ecx,[A] ; 'ecx = A'
26 mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 jg check_B ; если 'A>C', то переход на метку 'check_B',
30 mov ecx,[C] ; иначе 'ecx = C'
31 mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 mov eax,max
35 call atoi ; Вызов подпрограммы перевода символа в число
36 mov [max],eax ; запись преобразованного числа в 'max'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 mov ecx,[max]
39 cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
40 jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 mov ecx,[B] ; иначе 'ecx = B'
42 mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 mov eax, msg2
46 call sprint ; Вывод сообщения 'Наибольшее число: '
47 mov eax,[max]
48 call iprintLF ; Вывод 'max(A,B,C)'
49 call quit ; Выход

```

Загрузка файла «~/work/arch-pc/lab07/lab7-2.asm»... Matlab ▾ Ширина табуляции: 8 ▾ Ln

Рис. 8: Создаем файл и вводим листинг

Создайте исполняемый файл и проверьте его работу для разных значений B(рис. [-@fig:009]).

```

Наибольшее число: 90
amazurkevich@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
amazurkevich@vbox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 10
Наибольшее число: 50
amazurkevich@vbox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 90
Наибольшее число: 90
amazurkevich@vbox:~/work/arch-pc/lab07$

```

Рис. 9: Вводим различные B

##Изучение структуры файлы листинга

Создайте файл листинга для программы из файла lab7-2.asm(рис. [-@fig:010]).

```
amazurkevich@vbox:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ mcedit lab7-2.lst
```

Рис. 10: Создаем

Откройте файл листинга lab7-2.lst с помощью любого текстового редактора(рис. [-@fig:011]).

```
lab7-2.lst [----] 0 L: [ 1+ 0 1/225] *(0 /14458b) 0032 0x020 [*][X]
1                                     %include 'in_out.asm'
1                                     <1> ;----- slen -----
2                                     <1> ; Функция вычисления длины сообщения
3                                     <1> slen:.....
4 00000000 53                         <1> push    ebx.....
5 00000001 89C3                       <1> mov     ebx, eax.....
6                                     <1> .....
7                                     <1> nextchar:.....
8 00000003 803800                     <1> cmp     byte [eax], 0...
9 00000006 7403                       <1> jz      finished.....
10 00000008 40                        <1> inc     eax.....
11 00000009 EBF8                      <1> jmp     nextchar.....
12                                     <1> .....
13                                     <1> finished:
14 0000000B 29D8                     <1> sub     eax, ebx
15 0000000D 5B                        <1> pop     ebx.....
16 0000000E C3                       <1> ret.....
17                                     <1> .
18                                     <1> .
19                                     <1> ;----- sprint -----
20                                     <1> ; Функция печати сообщения
21                                     <1> ; входные данные: mov eax,<message>
```

Рис. 11: Открываем

Внимательно ознакомиться с его форматом и содержимым. Подробно объяснить содержимое трёх строк файла листинга по выбору.

Строка 5: 00000001 адрес в сегменте кода, 89C3 машинный код, mov ebx,eax присвоить eax значение ebx
 Строка 10: 00000008 адрес в сегменте кода, 40 машинный код, inc eax считывает значение eax и добавляет к нему 1, записывая обратно в eax
 Строка 14: 0000000B адрес в сегменте кода, 29D8 машинный код, sub eax,ebx вычитает (eax-ebx) и записывает значение в eax

Откройте файл с программой lab7-2.asm и в любой инструкции с двумя операндами удалите один операнд(рис. [-@fig:012]).


```

1 #include "in_out.asm"
2 section .data
3 msg1 db 'Введите B: ',0h
4 msg2 db "Наибольшее число: ",0h
5 A dd '20'
6 C dd '50'
7 section .bss
8 max resb 10
9 B resb 10
10 section .text
11 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 mov eax,msg1
15 call sprint
16 ; ----- Ввод 'B'
17 mov ecx,B
18 mov edx|
19 call sread
20 ; ----- Преобразование 'B' из символа в число
21 mov eax,B
22 call atoi ; Вызов подпрограммы перевода символа в число
23 mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
25 mov ecx,[A] ; 'ecx = A'
26 mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 jg check_B ; если 'A>C', то переход на метку 'check_B',
30 mov ecx,[C] ; иначе 'ecx = C'
31 mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 mov eax,max
35 call atoi ; Вызов подпрограммы перевода символа в число
36 mov [max],eax ; запись преобразованного числа в 'max'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)

```

Рис. 12: Открываем и удаляем

Выполните трансляцию с получением файла листинга:(рис. [-@fig:013]).



```

amazurkevich@vbox:~/work/arch-pc/lab07$ gedit lab7-2.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:18: error: invalid combination of opcode and operands
amazurkevich@vbox:~/work/arch-pc/lab07$ ls
in_out.asm lab7-1 lab7-1.asm lab7-1.o lab7-2 lab7-2.asm lab7-2.lst
amazurkevich@vbox:~/work/arch-pc/lab07$

```

Архитектура ЭВМ

Рис. 13: Транслируем

Выдается ошибка, но через ls видим, что создаются lab7-2 и lab7-2.asm
Смотрим файл(рис. [-@fig:014]).

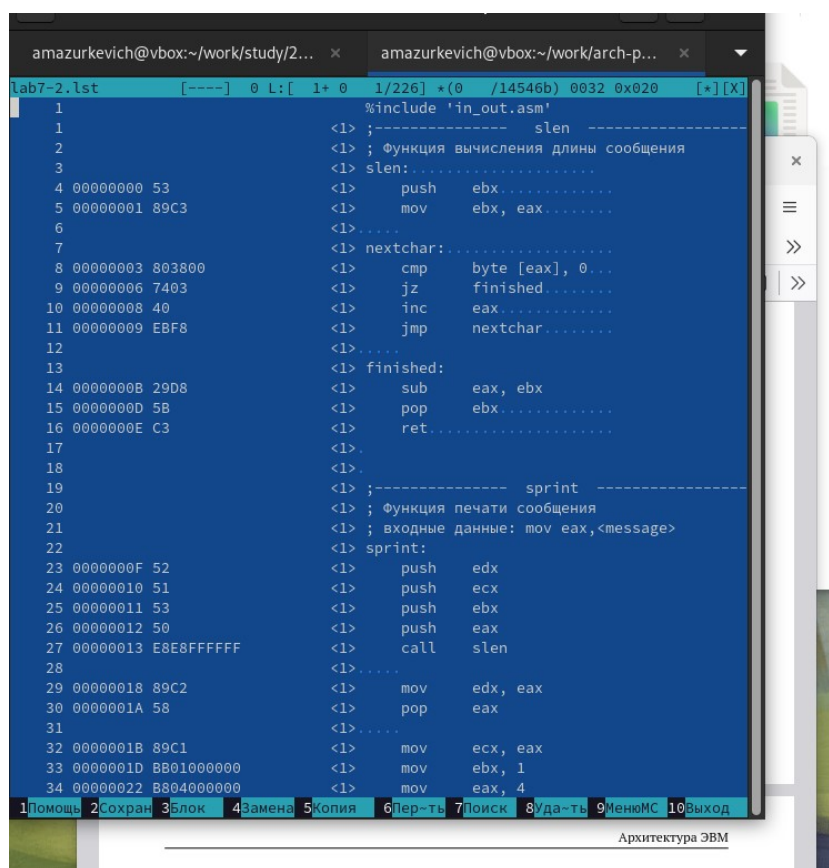


Рис. 14: Изучаем

Задание для самостоятельной работы

ВАРИАНТ 4

Напишите программу нахождения наименьшей из 3 целочисленных переменных a , b и c . Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу

Создадим для этого lab7-3.asm(рис. [-@fig:015]).



Рис. 15: Создаем

Напишите программу нахождения наименьшей из 3 целочисленных переменных `A`, `B` и `C`. (рис. [-@fig:016]).

```
1 %include 'in_out.asm'
2 section .data
3     msg1 db 'Введите B: ',0h
4     msg2 db 'Наименьшее число: ',0h
5     A dd '8'
6     C dd '88'
7 section .bss
8     min resb 10
9     B resb 10
10 section .text
11     global _start
12 _start:
13     mov eax,msg1
14     call sprint
15     mov ecx,B
16     mov edx,10
17     call sread
18     mov eax,B
19     call atoi
20     mov [B],eax
21     mov ecx,[A]
22     mov [min],ecx
23     cmp ecx,[C]
24     jl check_B
25     mov ecx,[C]
26     mov [min],ecx
27 check_B:
28     mov eax,min
29     call atoi
30     mov [min],eax
31     mov ecx,[min]
32     cmp ecx,[B]
33     jl fin
34     mov ecx,[B]
35     mov [min],ecx
36 fin:
37     mov eax, msg2
```

Рис. 16: Прописываем программу

Создадим файл и проверим его работу(рис. [-@fig:017]).

```
amazurkevich@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
amazurkevich@vbox:~/work/arch-pc/lab07$ ./lab7-3
Введите B: 68
Наименьшее число: 8
amazurkevich@vbox:~/work/arch-pc/lab07$
```

Рис. 17: Создаем файл, работает верно

Напишите программу, которая для введенных с клавиатуры значений `A` и `B` вычисляет значение заданной функции $f(A, B)$ и выводит результат вычислений. Вид функции $f(A, B)$ выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений `A` и `B` из 7.6.

Создадим для этого lab7-4.asm и откроем его, пропишем программу для нахождения функций(рис. [-@fig:018]).

```
6 otv: DB 'F(x) = ', 0h
7
8 SECTION .bss
9 x: RESB 80
10 a: RESB 80
11 res: RESB 80
12
13 SECTION .text
14 GLOBAL _start
15
16 _start:
17     mov eax, msg1
18     call sprint
19     mov ecx, x
20     mov edx, 80
21     call sread
22     mov eax, x
23     call atoi
24     mov [x], eax
25
26     mov eax, msg2
27     call sprint
28     mov ecx, a
29     mov edx, 80
30     call sread
31     mov eax, a
32     call atoi
33     mov [a], eax
34
35     mov eax, [a]
36     cmp eax, 0
37     je a_zero
38
39 a_not_zero:
40     mov eax, [x]
41     shl eax, 1
42     add eax, [a]
43     mov [res], eax
44     jmp fin
45
46 a_zero:
47     mov eax, [x]
48     shl eax, 1
49     add eax, 1
50     mov [res], eax
51
52 fin:
53     mov eax, otv
54     call sprint
55     mov eax, [res]
56     call iprintLF
57     call quit
```

Matlab ▾ Ширина табуляции: 8 ▾ Ln 37, C

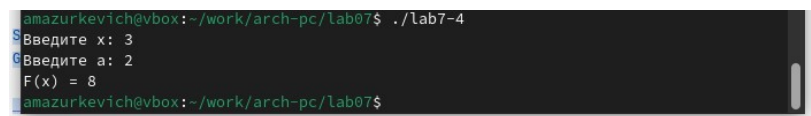
Рис. 18: Пишем программу

Создаем файл и проверяем для $x=3$ $a=0$ (рис. [-@fig:019]).

```
amazurkevich@vbox:~/work/arch-pc/lab07$ gedit lab7-4.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
amazurkevich@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
amazurkevich@vbox:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 0
F(x) = 7
amazurkevich@vbox:~/work/arch-pc/lab07$
```

Рис. 19: Вводим значения, ответ верный

Проверяем для $x=3$ $a=2$ (рис. [-@fig:020]).



```
amazurkevich@vbox:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 2
F(x) = 8
amazurkevich@vbox:~/work/arch-pc/lab07$
```

Рис. 20: Вводим значения, ответ верный

Выводы

Изучили команды условного и безусловного переходов. Приобрели навыки написания программ с использованием переходов. Познакомились с назначением и структурой файла листинга.

Список литературы