

Шаблон отчёта по лабораторной работе №7

Дисциплина: архитектура компьютера

Нурьеева Гулсoltан

Содержание

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 файлы листинга

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 Изучение структуры файлы листинга

Для начала я создала каталог для программ Лабораторной работы. потом перешла в него и создала файл lab07-1.asm (рис. 1).

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs$ cd lab07
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ touch lab7-1.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ ls
lab7-1.asm presentation_report
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$
```

ЬС

Рис. 1: Создания каталога и файла

Потом зашла на МС и через него скопировала файл in_out.asm в созданный каталог

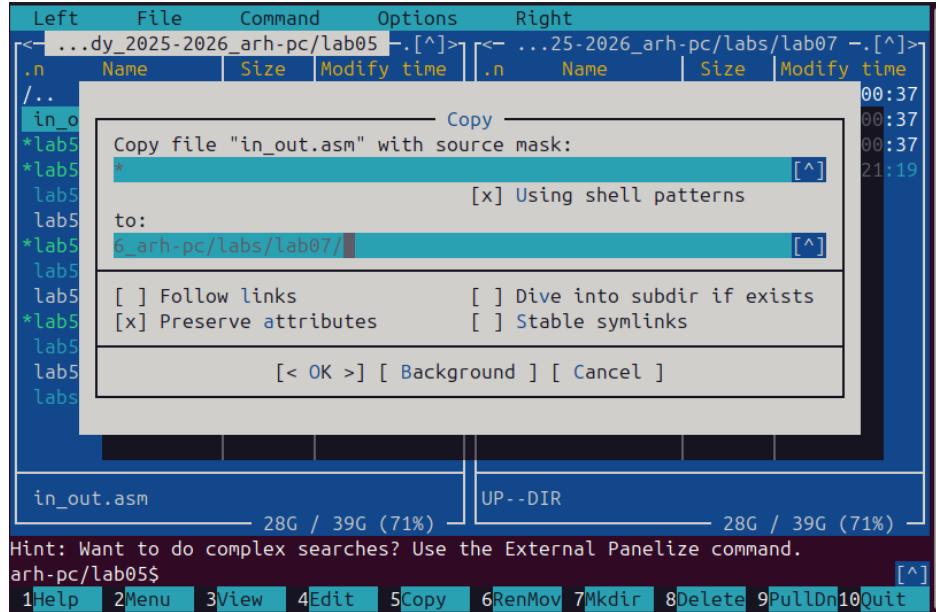


Рис. 2: Скопированиия файла in_out.asm в нужный каталог

После этого я открыла созданной мною файл с помощью клавиши F4 и ввел туда программу с использованием инструкции jmp

```
...26/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07/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 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

^G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line
```

Рис. 3: Программа с использованием инструкции jmp

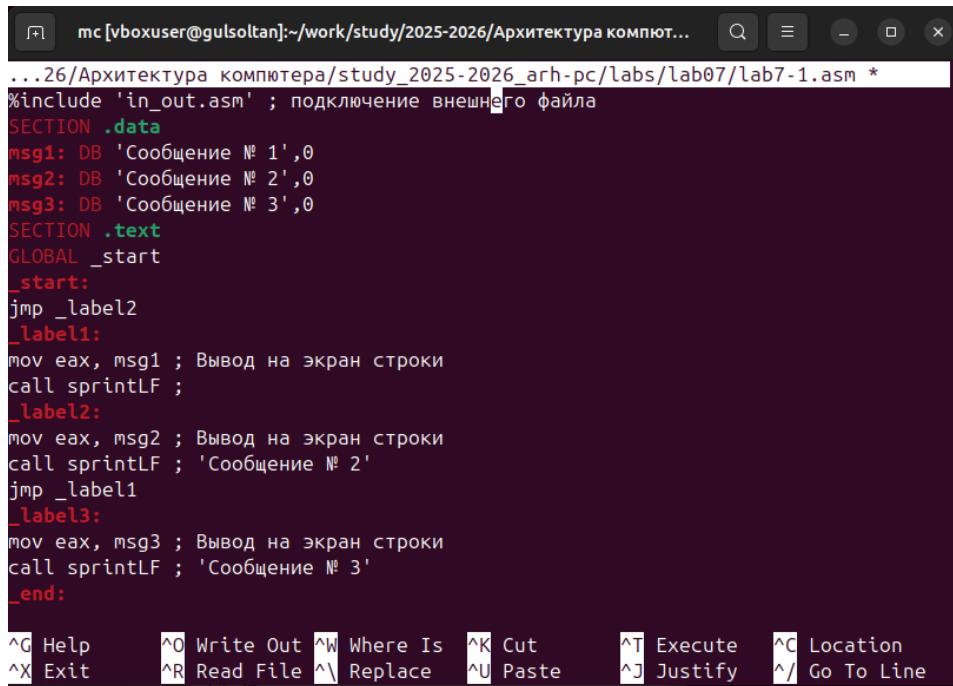
Потом я создала исполняемый файл и запустила его

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ nasm -f elf lab7-1.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ .lab7-1
.lab7-1: command not found
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$
```

БС

Рис. 4: Создания исполняемого файла

Я изменила текст файла чтобы осуществить переход назад в инструкции jmp. Для этого в текст программы после вывода сообщения № 2 добавила инструкцию jmp с меткой _label1, и после вывода сообщения № 1 добавила инструкцию jmp с меткой _end



```
mc [vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_арх-pc/labs/lab07/lab7-1.asm *]
...26/Архитектура компьютера/study_2025-2026_арх-pc/labs/lab07/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 sprintLF ;
_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:
```

Рис. 5: Изменения текста файла

Создала исполняемый файл и запустила его ещё раз но уже изменённого.

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ nasm -f elf lab7-1.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_
арх-pc/labs/lab07$
```

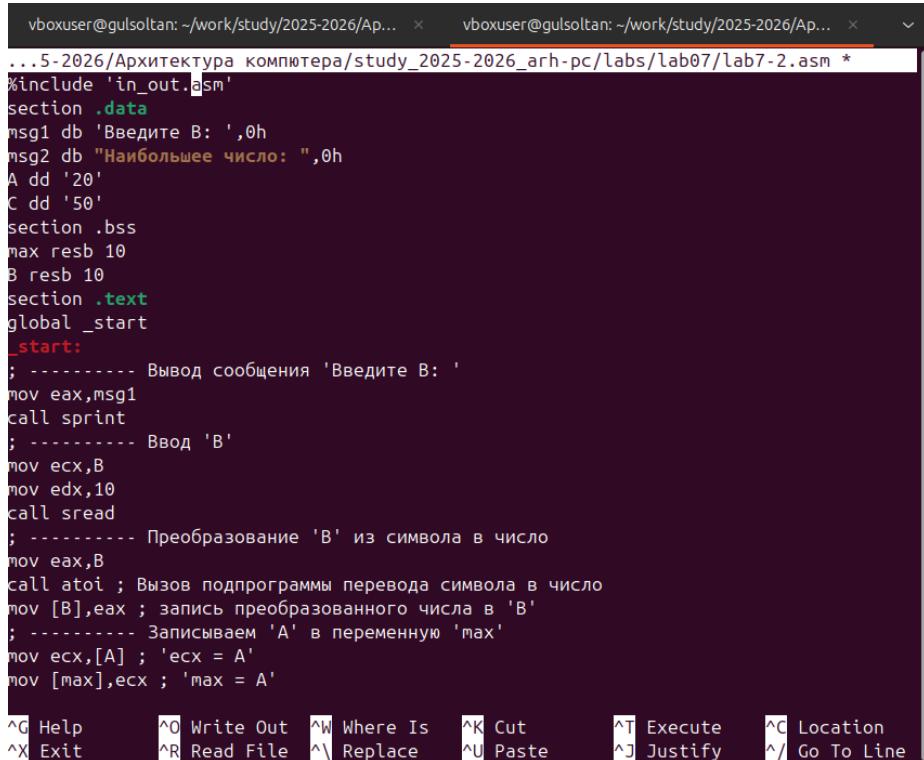
Рис. 6: Создания(изменённого) исполняемого файла

Потом я создала новый файл в том же каталоге lab7-2.asm

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07$ touch lab7-2.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07$
```

Рис. 7: Создания файла lab7-2.asm

После создания я открыла файл и ввела туда программу, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: A, B и C



The screenshot shows two terminal windows side-by-side. The left window contains the assembly code for the program, and the right window shows the assembly output.

```
%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 число
mov [B],eax ; запись преобразованного числа в 'B'
; ----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
```

Рис. 8: Программа, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: A, B и C

Потом создала исполняемый файл и запустил его. И ещё я проверила его работу

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07$ nasm -f elf lab7-2.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07$ ./lab7-2
Введите B: 20
Наибольшее число: 50
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07$ ./lab7-2
Введите B: 70
Наибольшее число: 70
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab07$
```

Рис. 9: Создания исполняемого файла lab7-2.asm

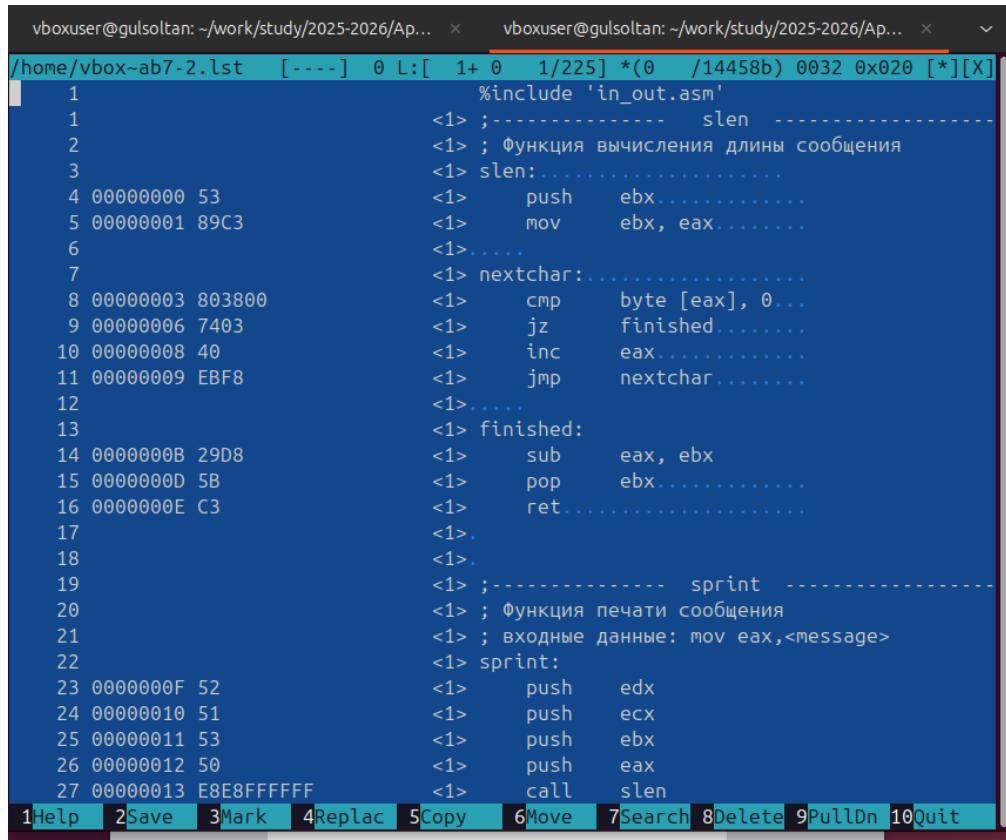
4.2 Изучение структуры файлы листинга

Я создала файл листинга с помощью nasm указав ключ -l и задала имя листинга в командной строке

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$
```

Рис. 10: Создания листинга

Потом открыла файл листинга с помощью mcedit и изучила содержимое



```
/home/vbox-ab7-2.lst [---] 0 L:[ 1+ 0 1/225] *(0 /14458b) 0032 0x020 [*][X]
1           %include 'in_out.asm'
1           ;----- slen -----
2           ; Функция вычисления длины сообщения
3           slen:
4 00000000 53      push    ebx.....
5 00000001 89C3    mov     ebx, eax.....
6           .....
7           nextchar:
8 00000003 803800  cmp     byte [eax], 0...
9 00000006 7403    jz      finished.....
10 00000008 40     inc     eax.....
11 00000009 EBF8   jmp     nextchar.....
12           .....
13           finished:
14 0000000B 29D8   sub     eax, ebx
15 0000000D 5B     pop     ebx.....
16 0000000E C3     ret.....
17           .....
18           .....
19           sprint:
20           ;----- sprint -----
21           ; Функция печати сообщения
22           ; входные данные: mov eax,<message>
23           sprint:
24 0000000F 52     push    edx
25 00000010 51     push    ecx
26 00000011 53     push    ebx
27 00000012 50     push    eax
28 00000013 E8E8FFFF  call    slen
1Help 2Save 3Mark 4Replace 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
```

Рис. 11: Открытие листинга

Выбрала первую строку и это 112. В строке которая показана в картинке снизу обозначается “00000086” — адрес в памяти, “E8C9FFFFFF” — машинный код для инструкции call a “call imprint” — обозначает вызов функции imprint.

```
112 00000086 E8C9FFFFFF <1> call imprint....
```

Рис. 12: 112 строка для объяснения

Выбрала вторую строку и это 14. В строке которая показана в картинке снизу обозначается “0000000B” — адрес в памяти, где расположена эта инструкция, 29D8 — машинный код для инструкции sub a “sub eax, ebx” — обозначает операцию, которая вычитает значение регистра ebx из значения регистра eax и сохраняет результат в eax.



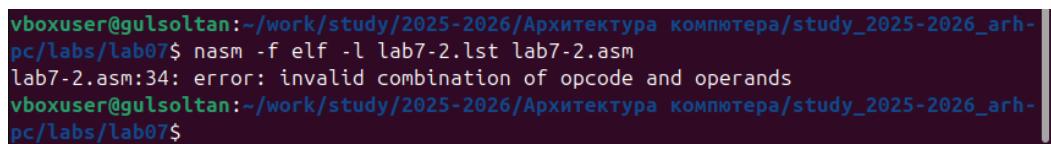
Рис. 13: 14 строка для объяснения

Выбрала третью строку и это 42. В строке которая показана в картинке снизу обозначается “00000153” — адрес в памяти, где расположена эта инструкция, 890D — машинный код для инструкции mov a “mov [max], ecx”— Обозначает операцию, которая копирует значение из регистра ecx в память по адресу, соответствующему метке или переменной max.



Рис. 14: 42 строка для объяснения

Потом в строке mov eax,max я убрала max и попробовала создать файл. Выдало ошибку, так как для программы нужно два операнда.

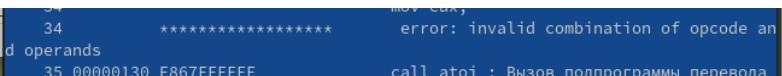


```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-  
pc/labs/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm  
lab7-2.asm:34: error: invalid combination of opcode and operands  
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-  
pc/labs/lab07$
```

ьс

Рис. 15: Ошибка в программе

В файле листинга показывает где ошибка и с чем оно связана



```
34      *****          mov eax,  
d operands           error: invalid combination of opcode an  
35 00000130 E867FFFFFFFFFF          call atoi ; Вызов подпрограммы перевода
```

Рис. 16: Осмотр листинга

4.3 Самостоятельная работа.

4.3.1 Напишите программу нахождения наименьшей из 3 целочисленных переменных a,b и c.

Для начала я создала файл и в него я написала программу нахождения наименьшей из 3 целочисленных переменных a,b и c.

```
/home/vboxuser/work/study-pc/labs/lab07/lab7-3.asm      853/1748  48%
%include 'in_out.asm'
SECTION .data
msg1 DB 'Введите В: ',0h
msg2 DB "Наименьшее число: ",0h
A dd '8'
C dd '68'
SECTION .bss
min 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 'А' в переменную 'min'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'min = A'
; ----- Сравниваем 'А' и 'С' (как символы)
cmp ecx,C ; Сравниваем 'A' и 'C'
1Help 2UnWrap 3Quit 4Hex 5Goto 6 7Search 8Raw 9Format 10Quit
```

Рис. 17: Внесения программы в файл

Потом создала исполняемый файл и запустила его и проверил все ли работает

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ nasm -f elf lab7-3.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ ./lab7-3
Введите В: 77
Наименьшее число: 8
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ ./lab7-3
Введите В: 1
Наименьшее число: 1
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$
```

Рис. 18: Создания исполняемого файла lab7-3.asm

4.3.2 Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции f(x) и выводит результат вычислений.

Для начала я создала файл и в него я написала программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции f(x) и выводит результат вычислений.

```
vboxuser@gulsoltan:~/work/study/2025-2026/Ap...      mc [vboxuser@gulsoltan]:~/work/study/2025-202...      ~
/home/vboxuser/work/study~pc/labs/lab07/lab7-4.asm    391/738    52%
%include 'in_out.asm'

SECTION .data
prim1 DB '2a-x ,x<a' ,0
prim2 DB '8, x=>a',0
X1 DB 'Введите значение X:',0
A1 DB 'Введите значение a:',0
otv DB 'Ответ: ',0

SECTION .bss
X RESB 20
A RESB 20
F RESB 20
SECTION .text
GLOBAL _start
_start:

mov eax,prim1
call sprintLF
mov eax,prim2
call sprintLF

mov eax,X1
call sprint

mov ecx,X
mov edx,10
call sread

1 Help 2 UnWrap 3 Quit 4 Hex 5 Goto 6 7 Search 8 Raw 9 Format 10 Quit
```

Рис. 19: Внесения программы в файл lab7-4.asm

После этого я создала исполняемый файл и запустила его. потом я написала цифры которые были таблице на Х и на А

```
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ nasm -f elf lab7-4.asm
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ ./lab7-4
2a-x ,x<a
8, x=>a
Введите значение X:4
Введите значение a:5
Ответ: 6
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$ ./lab7-4
2a-x ,x<a
8, x=>a
Введите значение X:5
Введите значение a:4
Ответ: 8
vboxuser@gulsoltan:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-
pc/labs/lab07$
```

Рис. 20: Создания исполняемого файла lab7-4.asm

Все готова!

5 Выводы

Я изучила команды условного и безусловного перехода. Приобрела навыки написания программ с переходами.

Список литературы

(<https://esystem.rudn.ru>) Архитектура компьютеров, Лабораторная работа №7