

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: Волчкова Елизавета Дмитриевна

Группа: НКАбд-01-24

МОСКВА

2024 г.

1.	Цель работы	3
2.	Теоретическое введение	4
3.	Задания для самостоятельной работы	5
4.	Выполнение лабораторной и самостоятельной работ	6
5.	Вывод	14
6.	Список литературы.	15
.		

Цель работы.

Изучить команды условного и безусловного переходов и приобрести навыки написания программ с использованием переходов, а также знакомство с назначением и структурой файла листинга.

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.

Задание для самостоятельной работы.

1. 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.

2. Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ 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 x и a из 7.6.

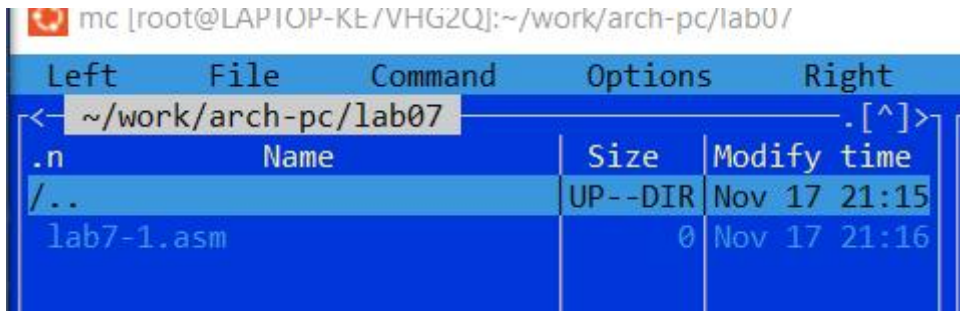
Порядок выполнения лабораторной работы.

1. Сначала создала каталог для программ лабораторной работы № 7, перешла в него и создайте файл lab7-1.asm:

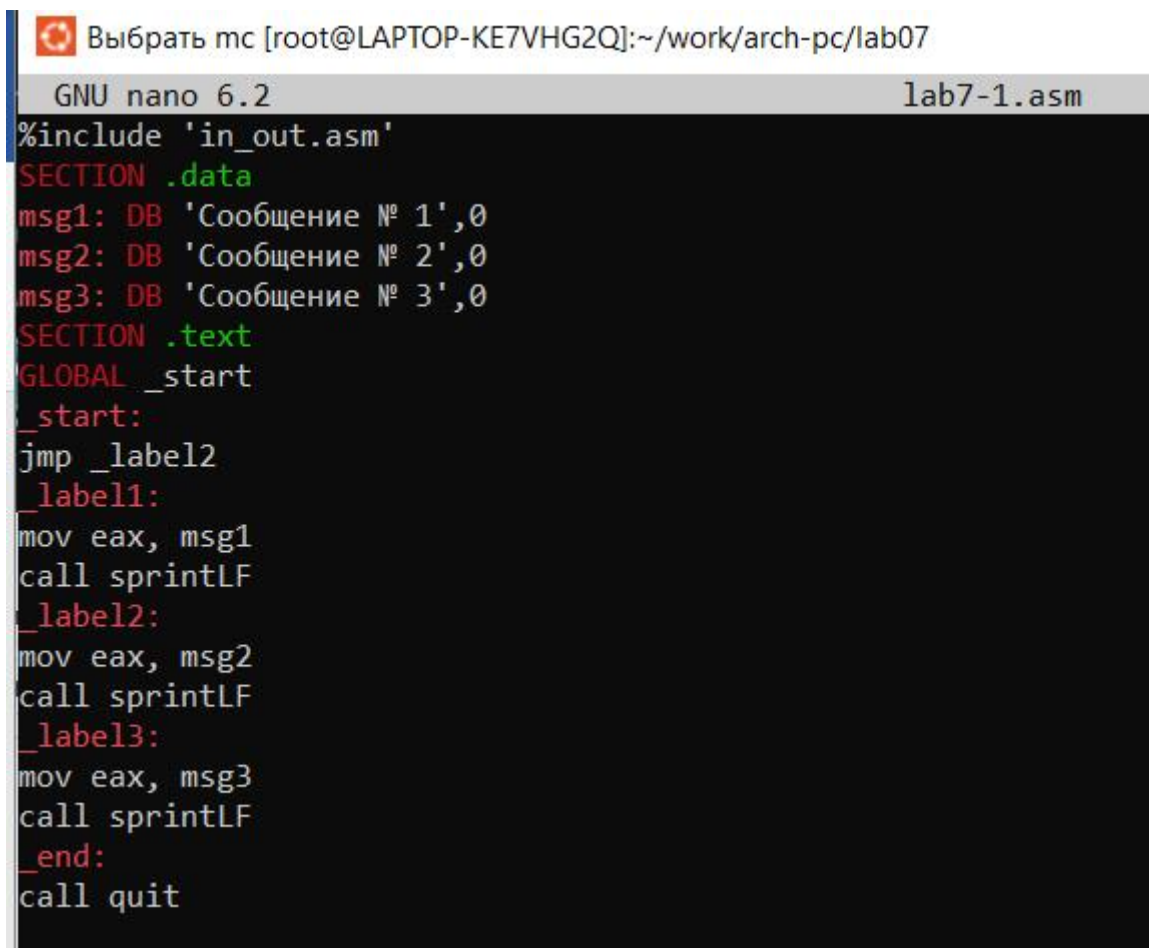
```
mkdir ~/work/arch-pc/lab07
```

```
cd ~/work/arch-pc/lab07
```

```
touch lab7-1.asm
```



Ввела в файл lab7-1.asm текст программы из листинга 7.1.



Создала исполняемый файл и запустила его.

Результат работы данной программы будет следующим:

root@LAPTOP-KE7VHG2Q:~# ./lab7

Сообщение № 2

Сообщение № 3

root@LAPTOP-KE7VHG2Q:~#

```
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# nasm -f elf32 lab7-1.asm -o obj.o
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ld -m elf_i386 -o lab7 obj.o
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ./lab7
Сообщение № 2
Сообщение № 3
```

Далее в текст программы после вывода сообщения № 2 добавила инструкцию `jmp` с меткой `_label1` и после вывода сообщения № 1 добавила инструкцию `jmp` с меткой `_end`.

Выбрать mc [root@LAPTOP-KE7VHG2Q]:~/work/arch-pc/lab07

```
GNU nano 6.2 lab7-2.asm
#include 'in_out.asm'
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
call sprintLF
jmp _end
_label2:
mov eax, msg2
call sprintLF
jmp _label1
_label3:
mov eax, msg3
call sprintLF
_end:
call quit
```

```
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# nasm -f elf32 lab7-2.asm -o obj.o
```

```
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ld -m elf_i386 -o lab7 obj.o
```

```
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ./lab7
```

```
Сообщение № 2
```

```
Сообщение № 1
```

Затем создала исполняемый файл и проверила его работу. Потом изменила текст программы добавив или изменив инструкции `jmp`, чтобы вывод программы был следующим:

```
root@LAPTOP-KE7VHG2Q:~# ./lab7
```

```
Сообщение № 3
```

```
Сообщение № 2
```

```
Сообщение № 1
```

```
root@LAPTOP-KE7VHG2Q:~#
```



```
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ./lab7
Сообщение № 3
Сообщение № 2
Сообщение № 1
root@LAPTOP-KE7VHG2Q:~#
```

После создала файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. И я внимательно изучила текст программы из листинга 7.3 и ввела в lab7-2.asm

Далее я создала исполняемый файл и проверила его работу для разных значений В.

```
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# nasm -f elf32 lab7-2.asm -o obj.o
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ld -m elf_i386 -o lab7 obj.o
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ./lab7
Введите В: 3
Наибольшее число: 50
root@LAPTOP-KE7VHG2Q:~#
```

Потом создала файл листинга для программы из файла lab7-2.asm

```
nasm -f elf -l list.lst lab7-2.asm
```

Открыла файл листинга list.lst с помощью текстового редактора nano:

```
nano list.lst
```

```
mc [root@LAPTOP-KE7VHG2Q]:~/work/arch-pc/lab07
GNU nano 6.2 list.lst
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
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 E8E8FFFF <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
34     0000001D BB01000000 <1>      mov     ebx, 1
35     00000022 B804000000 <1>      mov     eax, 4
36     00000027 CD80    <1>      int     80h
37     <1>
38     00000029 5B      <1>      pop     ebx
39     0000002A 59      <1>      pop     ecx
40     0000002B 5A      <1>      pop     edx
41     0000002C C3      <1>      ret
42     <1>
43     <1>
44     <1> ;----- sprintf -----
45     <1> ; Функция печати сообщения с переводом строки
```

Я открыла файл с программой lab7-2.asm и в любой инструкции с двумя операндами удалить один операнд. Выполнила трансляцию с получением файла листинга: `nasm -f elf -l list.lst lab7-2.asm -o obj.o`.

Какие выходные файлы создаются в этом случае?

list.lst

obj.o

Что добавляется в листинге?

Файл листинга позволяет увидеть работу компилятора FASM: что генерирует каждая строка исходного кода, сколько байт занимают машинные команды, какие значения присваиваются переменным. Листинг состоит из трех колонок: первая содержит адреса (точнее, смещения в секции), вторая — сгенерированный машинный код и третья — соответствующие строки исходного кода программы.

1. Написала 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.

Создала исполняемый файл и проверила его работу.

2. Написала программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ выбрала из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создала исполняемый файл и проверила его работу для значений x и a из 7.6.

```
mc [root@LAPTOP-KE7VHG2Q]:~/work/arch-pc/lab07
GNU nano 6.2 lab7-3.asm
#include 'in_out.asm'
section .data
    A db 95
    B db 2
    C db 61
    msg db "Наибольшее число: ", 0

section .bss
    max resb 1

section .text
    global _start

_start:
    ; Сравниваем A и B
    mov al, [A]
    cmp al, [B]
    jge check_C
    mov al, [B]

check_C:
    cmp al, [C]
    jge store_max
    mov al, [C]

store_max:
    mov [max], al

    ; Выводим сообщение
    call iprintLF

    ; Завершаем программу
    call quit
```

```
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# nasm -f elf32 lab7-3.asm -o obj.o
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ld -m elf_i386 -o lab7 obj.o
root@LAPTOP-KE7VHG2Q:~/work/arch-pc/lab07# ./lab7
95
root@LAPTOP-KE7VHG2Q:~#
```

Вывод.

Целью было изучить команды условного и безусловного переходов и приобрести навыки написания программ с использованием переходов, а также знакомство с назначением и структурой файла листинга, сделав задания, я смога разобраться в данной теме.

Список литературы.

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
11. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
13. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
14. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix.
15. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
16. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).