

Лабораторная работа №7

Отчет

Зубов Иван Александрович

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Самостоятельная работа	13
5	Выводы	16

Список иллюстраций

3.1	Заполняем файл	7
3.2	Запускаем файл и смотрим на его работу	8
3.3	Редактируем файл	8
3.4	Запускаем файл и смотрим на его работу	8
3.5	Редактируем файл	9
3.6	Проверяем, сошелся ли наш вывод с данным в условии выводом .	9
3.7	Заполняем файл	10
3.8	Смотрим, что получается	10
3.9	Создаем файл листинга	10
3.10	Изучаем файл	11
3.11	Удаляем операндум из файла	12
3.12	Транслируем файл	12
4.1	Пишем программу	13
4.2	Смотрим, что все получилось	14
4.3	Пишем программу	14
4.4	Смотрим, что все получилось	15

Список таблиц

1 Цель работы

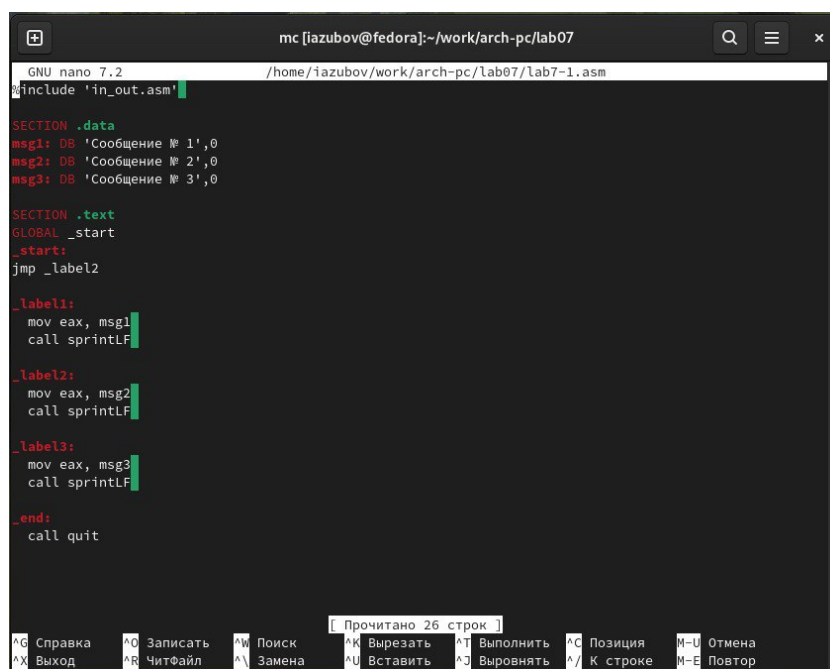
Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

2 Задание

Написать программы для решения выражений.

3 Выполнение лабораторной работы

Создаем каталог для программ лабораторной работы № 7 с помощью команды `mkdir`, перейдем в него и создадим файл `lab7-1.asm` с помощью команды `touch`. Откроем файл в Midnight Commander и заполняем его в соответствии с листингом 7.1



```
GNU nano 7.2 /home/iazubov/work/arch-pc/lab07/lab7-1.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 sprintf

_label2:
mov eax, msg2
call sprintf

_label3:
mov eax, msg3
call sprintf

_end:
call quit
```

Рис. 3.1: Заполняем файл

Создаем исполняемый файл и запускаем его

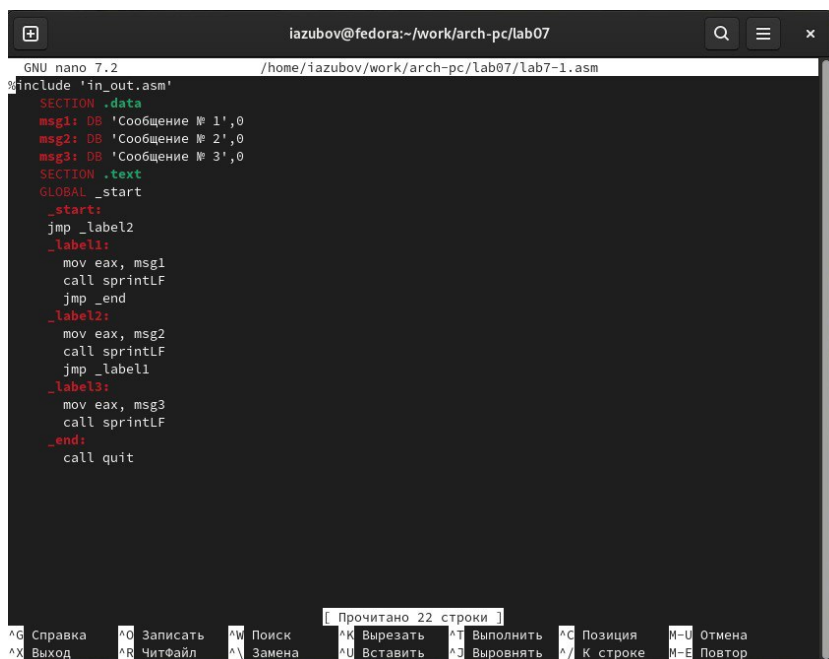
```

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

```

Рис. 3.2: Запускаем файл и смотрим на его работу

Снова открываем файл для редактирования и изменяем его в соответствии с листингом 7.2



```

GNU nano 7.2 /home/iazubov/work/arch-pc/lab07/lab7-1.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 sprintf
jmp _end
_label2:
mov eax, msg2
call sprintf
jmp _label1
_label3:
mov eax, msg3
call sprintf
_end:
call quit

```

Рис. 3.3: Редактируем файл

Создаем исполняемый файл и запускаем его

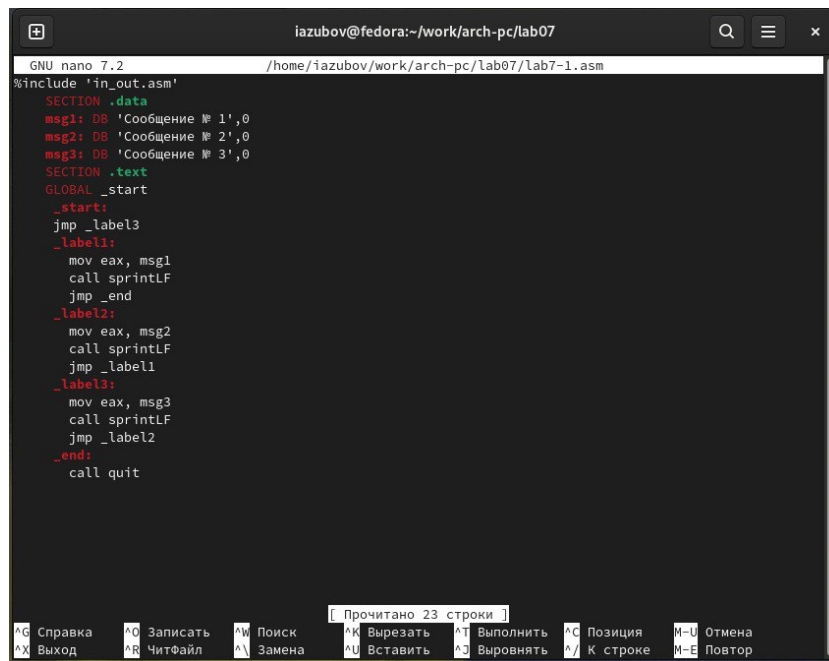
```

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

```

Рис. 3.4: Запускаем файл и смотрим на его работу

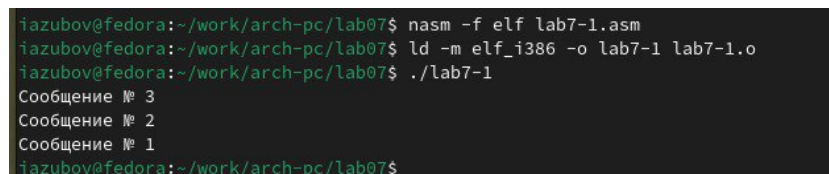
Снова открываем файл для редактирования и изменяем его, чтобы произошел данный вывод



```
GNU nano 7.2 /home/iazubov/work/arch-pc/lab07/lab7-1.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 _label3
_label1:
mov eax, msg1
call sprintf
jmp _end
_label2:
mov eax, msg2
call sprintf
jmp _label1
_label3:
mov eax, msg3
call sprintf
jmp _label2
_end:
call quit
```

Рис. 3.5: Редактируем файл

Создаем исполняемый файл и запускаем его



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

Рис. 3.6: Проверяем, сошелся ли наш вывод с данным в условии выводом

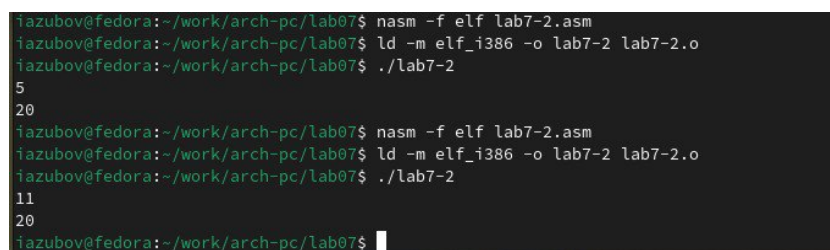
Создаем новый файл с помощью команды touch, открываем файл в Midnight Commander и заполняем его в соответствии с листингом 7.3



```
GNU nano 7.2 /home/iazubov/work/arch-pc/lab07/lab7-2.asm
#include 'in_out.asm'
section .data
    msg1 db '',0h
    msg2 db "",0h
    A dd '20'
    C dd '50'
section .bss
    max resb 10
    B resb 10
section .text
    global _start
_start:
    mov eax,msg1
    call sprint
    mov ecx,B
    mov edx,10
    call sread
    mov eax,B
    call atoi
    mov [B],eax
    mov ecx,[A]
    mov [max],ecx
check_B:
    mov eax,max
    call atoi
    mov [max],eax
    mov ecx,[max]
    cmp ecx,[B]
    jg fin
    mov ecx,[B]
```

Рис. 3.7: Заполняем файл

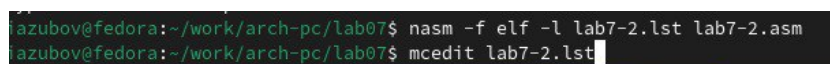
Создаем исполняемый файл и проверяем его работу, вводя разные значения В



```
iazubov@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
iazubov@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
iazubov@fedora:~/work/arch-pc/lab07$ ./lab7-2
5
20
iazubov@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
iazubov@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
iazubov@fedora:~/work/arch-pc/lab07$ ./lab7-2
11
20
iazubov@fedora:~/work/arch-pc/lab07$
```

Рис. 3.8: Смотрим, что получается

Создаем файл листинга для программы lab7-2.asm



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

Рис. 3.9: Создаем файл листинга

Открываем файл листинга с помощью команды mcedit и изучаем его

```

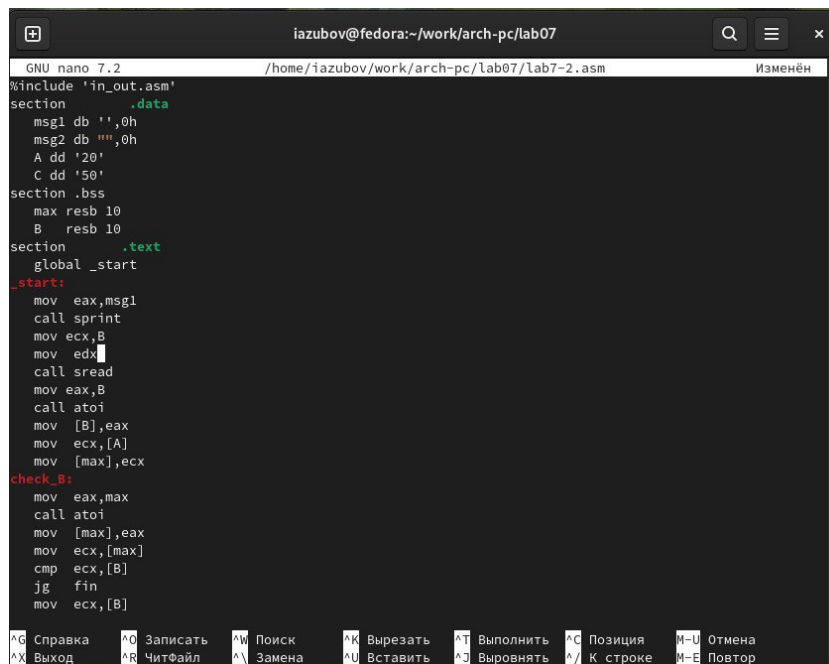
lab7-2.lst [----] 61 L: [ 1+ 0 1/208] *(61 /12617b) 0010 0x00A [X]
1      %include 'in_out.asm'
2      <1> ;----- slen -----
3      <1> ; Функция вычисления длины сообщения
4      00000000 53      <1> push    ebx
5      00000001 89C3    <1> mov     ebx, eax
6      <1>
7      <1> nextchar:
8      00000003 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>
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 E8E8FFFFFF <1> call    slen
28     <1>
29     00000018 89C2    <1> mov     edx, eax
30     0000001A 58      <1> pop     eax
31     <1>

```

Рис. 3.10: Изучаем файл

Строка 5 - 00000001 89C3 mov ebx, eax 00000001 - адрес в сегменте кода 89C3 - машинный код для инструкции mov ebx,eax - присваивание переменной ebx значения, хранящееся в регистре eax
 Строка 26 - 00000012 50 push eax 00000012 - адрес в сегменте кода 50 - машинный код для инструкции push eax - значение из регистра eax помещается в стек
 Строка 53 - 0000003B E8CFFFFFFF call sprint 0000003B - адрес в сегменте кода E8CFFFFFFF - машинный код для инструкции call sprint - вызов функции sprint, которая выводит данные на экран

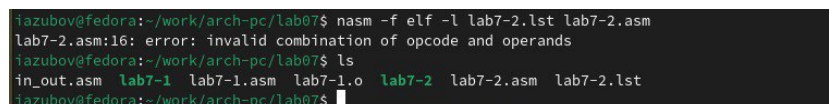
Открываем файл и удаляем один операндум



```
GNU nano 7.2 /home/iazubov/work/arch-pc/lab07/lab7-2.asm
#include 'in_out.asm'
section .data
msg1 db '',0h
msg2 db '""',0h
A dd '20'
C dd '50'
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx,B
mov edx,
call sread
mov eax,B
call atoi
mov [B],eax
mov ecx,[A]
mov [max],ecx
check_B:
mov eax,max
call atoi
mov [max],eax
mov ecx,[max]
cmp ecx,[B]
jg fin
mov ecx,[B]
```

Рис. 3.11: Удаляем операндум из файла

Транслируем с получением файла листинга



```
iazubov@fedora:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:16: error: invalid combination of opcode and operands
iazubov@fedora:~/work/arch-pc/lab07$ ls
in_out.asm lab7-1 lab7-1.asm lab7-1.o lab7-2 lab7-2.asm lab7-2.lst
iazubov@fedora:~/work/arch-pc/lab07$
```

Рис. 3.12: Транслируем файл

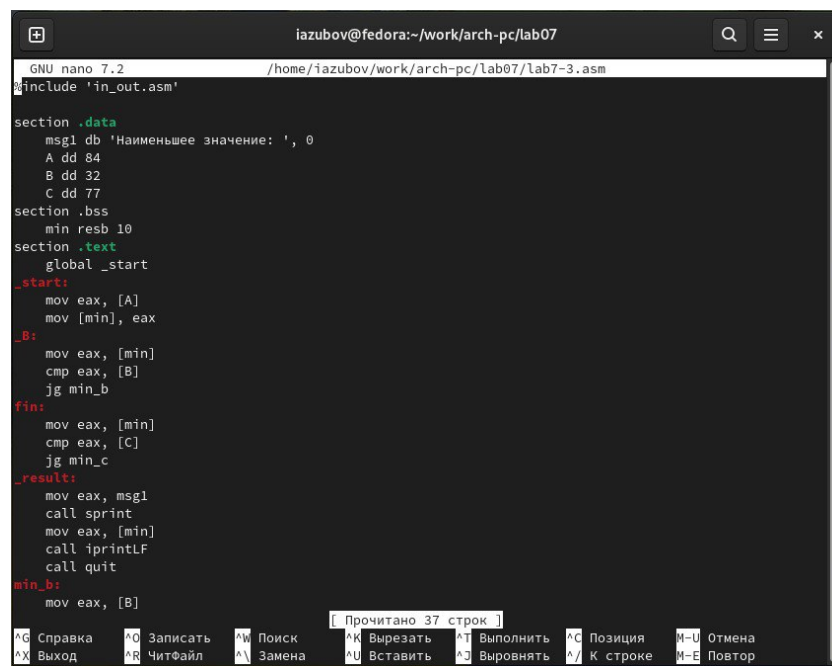
При трансляции файла, выдается ошибка, но создаются исполнительный файл lab7-2 и lab7-2.lst

4 Самостоятельная работа

Вариант 13

- 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.

Создаем новый файл,открываем его и пишем программу, которая выберет наименьшее число из трех



```
GNU nano 7.2 /home/iazubov/work/arch-pc/lab07/lab7-3.asm
#include 'in_out.asm'

section .data
    msg1 db 'Наименьшее значение: ', 0
    A dd 84
    B dd 32
    C dd 77
section .bss
    min resb 10
section .text
    global _start
_start:
    mov eax, [A]
    mov [min], eax
_B:
    mov eax, [min]
    cmp eax, [B]
    jg min_b
_fin:
    mov eax, [min]
    cmp eax, [C]
    jg min_c
_result:
    mov eax, msg1
    call sprint
    mov eax, [min]
    call iprintLF
    call quit
min_b:
    mov eax, [B]
```

Рис. 4.1: Пишем программу

Транслируем файл и смотрим на работу программы

```

iazubov@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
iazubov@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
iazubov@fedora:~/work/arch-pc/lab07$ ./lab7-3
Наименьшее значение: 32

```

Рис. 4.2: Смотрим, что все получилось

- 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.

Создаем новый файл в каталоге, открываем его и пишем программу, которая решит систему уравнений, при известных данных

```

GNU nano 7.2 /home/iazubov/work/arch-pc/lab07/lab7-4.asm
#include 'in_out.asm'

SECTION .data
msg1: DB 'Введите x: ', 0h
msg2: DB 'Введите a: ', 0h
ans: DB 'Результат системы: ', 0h

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

```

Рис. 4.3: Пишем программу

Транслируем файл и проверяем его работу при $x=3$ и $a=9$, при $x=6$ и $a=4$

```
iazubov@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
iazubov@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
iazubov@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 3
Введите a: 9
Результат системы: 2
iazubov@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
iazubov@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
iazubov@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите x: 6
Введите a: 4
Результат системы: 24
```

Рис. 4.4: Смотрим, что все получилось

5 Выводы

Мы познакомились с структурой файла листинга, изучили команды условного и безусловного перехода.