

Отчет по лабораторной работе №8.

Дисциплина: Архитектура компьютера.

Лобанова Полина Иннокентьевна

Содержание

1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Выполнение самостоятельной работы	10
4	Вывод	13

1 Цель работы

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

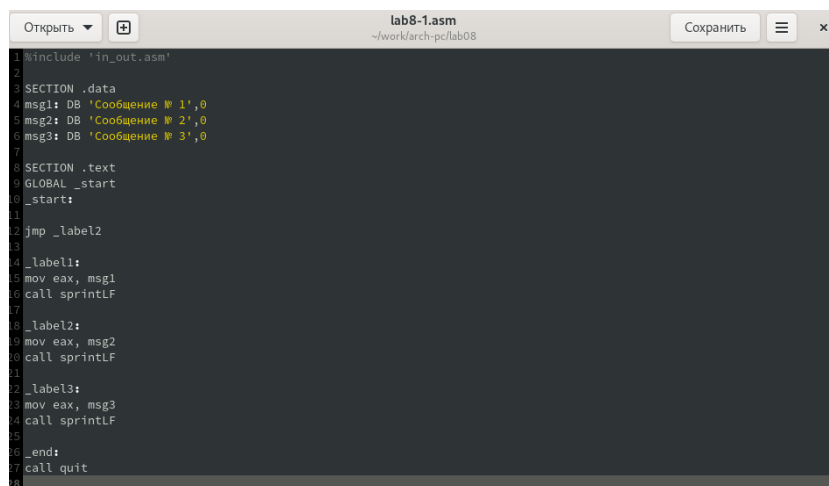
2 Выполнение лабораторной работы

1. Создадим каталог lab08 и текстовый файл lab8-1.asm.

```
[pilobanova@i0 ~]$ mkdir ~/work/arch-pc/lab08  
[pilobanova@i0 ~]$ cd ~/work/arch-pc/lab08  
[pilobanova@i0 lab08]$ touch lab8-1.asm
```

Рис. 2.1: Создание каталога lab08 и файла lab8-1.asm.

2. Введем текст программы в файл lab8-1.asm.



```
1 %include 'in_out.asm'  
2  
3 SECTION .data  
4 msg1: DB 'Сообщение № 1',0  
5 msg2: DB 'Сообщение № 2',0  
6 msg3: DB 'Сообщение № 3',0  
7  
8 SECTION .text  
9 GLOBAL _start  
10 _start:  
11  
12 jmp _label2  
13  
14 _label1:  
15 mov eax, msg1  
16 call sprintf  
17  
18 _label2:  
19 mov eax, msg2  
20 call sprintf  
21  
22 _label3:  
23 mov eax, msg3  
24 call sprintf  
25  
26 _end:  
27 call quit  
28
```

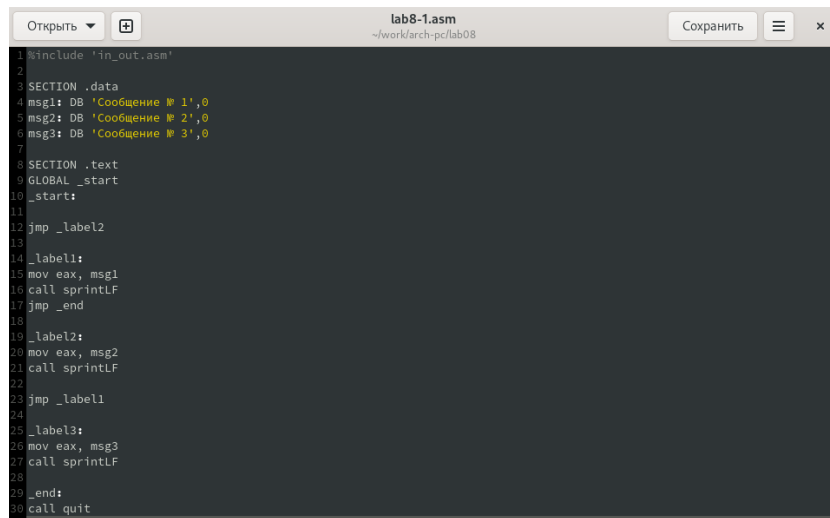
Рис. 2.2: Текст программы в файле lab8-1.asm.

3. Создадим исполняемый файл и проверим его.

```
[pilobanova@i0 lab08]$ nasm -f elf lab8-1.asm  
[pilobanova@i0 lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o  
[pilobanova@i0 lab08]$ ./lab8-1  
Сообщение № 2  
Сообщение № 3
```

Рис. 2.3: Создание исполняемого файла и его запуск.

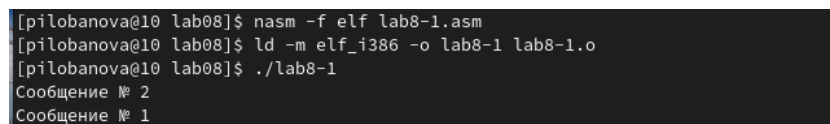
4. Изменим текст программы в файле lab8-1.asm так, чтобы она выводила сначала 'Сообщение № 2', потом 'Сообщение № 1' и завершала работу.



```
1 %include "in_out.asm"
2
3 SECTION .data
4 msg1i DB 'Сообщение № 1',0
5 msg2i DB 'Сообщение № 2',0
6 msg3i DB 'Сообщение № 3',0
7
8 SECTION .text
9 GLOBAL _start
10 _start:
11
12 jmp _label2
13
14 _label1:
15 mov eax, msg1
16 call sprintf
17 jmp _end
18
19 _label2:
20 mov eax, msg2
21 call sprintf
22
23 jmp _label1
24
25 _label3:
26 mov eax, msg3
27 call sprintf
28
29 _end:
30 call quit
```

Рис. 2.4: lab8-1.asm

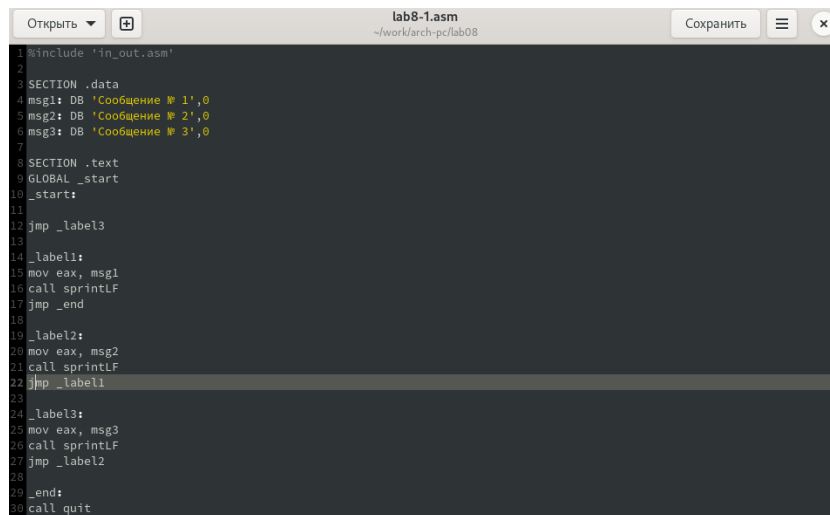
5. Создадим исполняемый файл и проверим его.



```
[pilobanova@10 lab08]$ nasm -f elf lab8-1.asm
[pilobanova@10 lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[pilobanova@10 lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 1
```

Рис. 2.5: Создание исполняемого файла и его запуск.

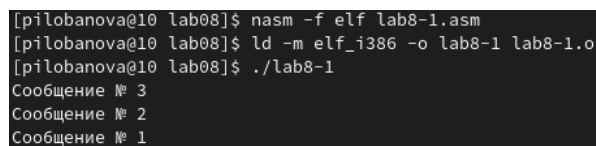
6. Изменим текст программы так, чтобы в результате получалось Сообщение № 3 Сообщение № 2 Сообщение № 1



```
1 %include "in_out.asm"
2
3 SECTION .data
4 msg1: DB "Сообщение № 1",0
5 msg2: DB "Сообщение № 2",0
6 msg3: DB "Сообщение № 3",0
7
8 SECTION .text
9 GLOBAL _start
10 _start:
11
12 jmp _label3
13
14 _label1:
15 mov eax, msg1
16 call sprintf
17 jmp _end
18
19 _label2:
20 mov eax, msg2
21 call sprintf
22 jmp _label1
23
24 _label3:
25 mov eax, msg3
26 call sprintf
27 jmp _label2
28
29 _end:
30 call quit
```

Рис. 2.6: Измененный текст программы в файле *lab8-1.asm*.

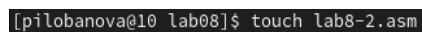
7. Создадим исполняемый файл и проверим его.



```
[pilobanova@i0 lab08]$ nasm -f elf lab8-1.asm
[pilobanova@i0 lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[pilobanova@i0 lab08]$ ./lab8-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
```

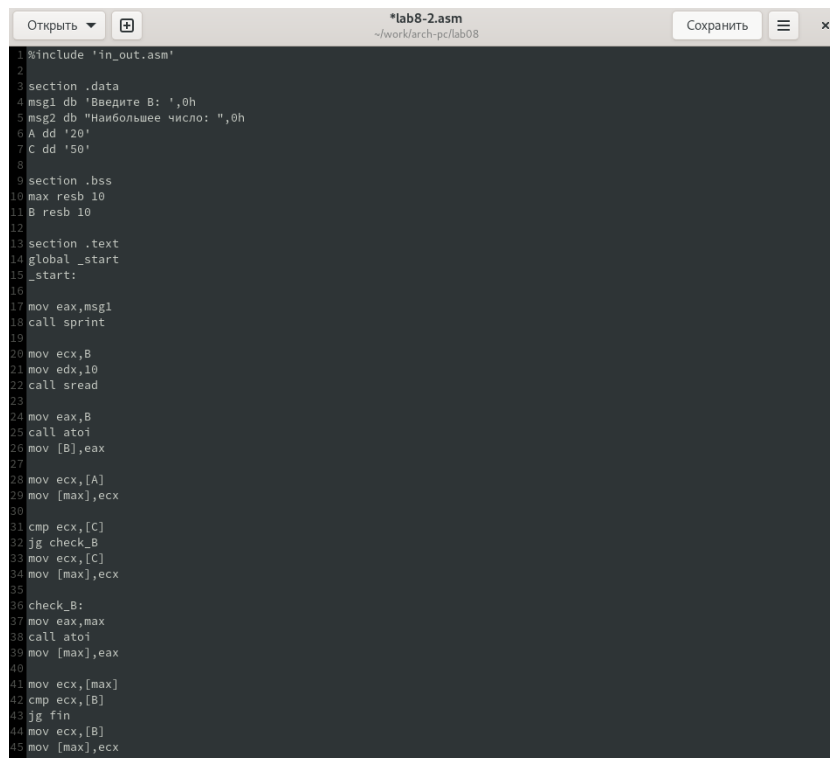
Рис. 2.7: Создание исполняемого файла и его запуск.

8. Создадим текстовый файл *lab8-2.asm* и заполним его.



```
[pilobanova@i0 lab08]$ touch lab8-2.asm
```

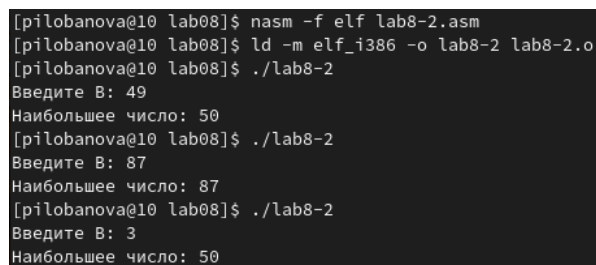
Рис. 2.8: Создание текстового файла *lab8-2.asm*.



```
1 %include 'in_out.asm'
2
3 section .data
4 msg1 db 'Введите B: ',0h
5 msg2 db "Наибольшее число: ",0h
6 A dd '20'
7 C dd '50'
8
9 section .bss
10 max resb 10
11 B resb 10
12
13 section .text
14 global _start
15 _start:
16
17 mov eax,msg1
18 call sprint
19
20 mov ecx,B
21 mov edx,10
22 call sread
23
24 mov eax,B
25 call atoi
26 mov [B],eax
27
28 mov ecx,[A]
29 mov [max],ecx
30
31 cmp ecx,[C]
32 jg check_B
33 mov ecx,[C]
34 mov [max],ecx
35
36 check_B:
37 mov eax,max
38 call atoi
39 mov [max],eax
40
41 mov ecx,[max]
42 cmp ecx,[B]
43 jg fin
44 mov ecx,[B]
45 mov [max],ecx
46
```

Рис. 2.9: Текст программы в файле *lab8-2.asm*.

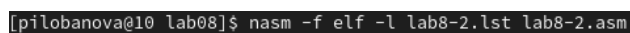
9. Создадим исполняемый файл и запустим его.



```
[pilobanova@i0 lab08]$ nasm -f elf lab8-2.asm
[pilobanova@i0 lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o
[pilobanova@i0 lab08]$ ./lab8-2
Введите B: 49
Наибольшее число: 50
[pilobanova@i0 lab08]$ ./lab8-2
Введите B: 87
Наибольшее число: 87
[pilobanova@i0 lab08]$ ./lab8-2
Введите B: 3
Наибольшее число: 50
```

Рис. 2.10: Создание исполняемого файла и его запуск.

10. Создадим файл листинга для программы из файла *lab8-2.asm*.



```
[pilobanova@i0 lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
```

Рис. 2.11: Создание файла листинга *lab8-2.lst*.

11. Откроем файл листинга с помощью текстового редактора *mcedit*.

```

lab8-2.lst      [----]  0 L: [ 1+ 0  1/230] *(0  /13472b) 0032 0x020 [*][X]
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>
1Помощь 2Сох-ть 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюMC10Выход

```

Рис. 2.12: Текст файла листинга lab8-2.lst.

“5” - номер строки “00000000” - адрес строки “53” - машинный код “push ebx” - исходный текст программы

```

5 00000000 53      <1> push    ebx.....

```

Рис. 2.13: 5 строка листинга.

“6” - номер строки “00000001” - адрес строки “89C3” - машинный код “mov ebx,eax” - исходный текст программы

```

6 00000001 89C3    <1> mov     ebx, eax.....

```

Рис. 2.14: 6 строка листинга.

“15” - номер строки “0000000B” - адрес строки “29DB” - машинный код “sub eax,ebx” - исходный текст программы

```

15 0000000B 29D8    <1> sub     eax, ebx

```

Рис. 2.15: 15 строка листинга

12. Изменим файл lab8-2.asm, удалив один операнд.


```

1 %include 'in_out.asm'
2
3 section .data
4 msg1 db 'Введите B: ',0h
5 msg2 db "Наибольшее число: ",0h
6 A dd '20'
7 C dd '50'
8
9 section .bss
10 max resb 10
11 B resb 10
12
13 section .text
14 global _start
15 _start:
16
17 mov eax,msg1
18 call sprint
19
20 mov ecx,B
21 mov edx,10
22 call sread
23
24 mov eax,B
25 call atoi
26 mov [B],eax
27
28 mov ecx,[A]
29 mov [max],ecx
30
31 cmp ecx,
32 jg check_B
33 mov ecx,[C]
34 mov [max],ecx
35
36 check_B:
37 mov eax,max
38 call atoi
39 mov [max],eax
40
41 mov ecx,[max]
42 cmp ecx,[B]
43 jg fin
44 mov ecx,[B]
45 mov [max],ecx
46
47 fin:

```

Рис. 2.16: Измененный текст программы в файле lab8-2.asm.

13. Выполним трансляцию с получением файла листинга. Нам выдает ошибку, потому что для выполнения команды необходимо два операнда.

```

[pilobanova@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:31: error: invalid combination of opcode and operands

```

Рис. 2.17: Создание файла листинга.

```

lab8-2.lst      [----]  0 L:[202+17 219/231] *(12973/13559b) 0032 0x020[*][X]
27 .....
28 00000110 8B0D[35000000]      mov ecx,[A]
29 00000116 890D[00000000]      mov [max],ecx
30 .....
31                               cmp ecx
31                               ***** error: invalid combination of opcode an
32 0000011C 7F0C                               jg check_B
33 0000011E 8B0D[39000000]      mov ecx,[C]
34 00000124 890D[00000000]      mov [max],ecx
35 .....
36                               check_B:
37 0000012A B8[00000000]      mov eax,max
38 0000012F E868FFFFFF      call atoi
39 00000134 A3[00000000]      mov [max],eax
40 .....
41 00000139 8B0D[00000000]      mov ecx,[max]
42 0000013F 3B0D[0A000000]      cmp ecx,[B]
43 00000145 7F0C                               jg fin
44 00000147 8B0D[0A000000]      mov ecx,[B]
45 0000014D 890D[00000000]      mov [max],ecx
46 .....
47                               fin:

```

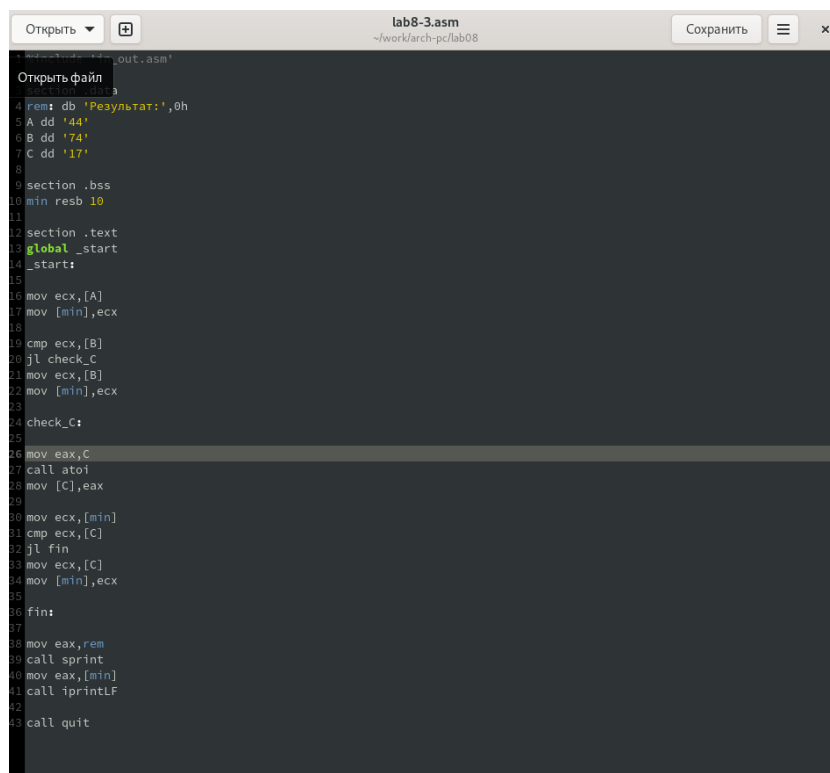
Рис. 2.18: Текст файла листинга с объяснением ошибки.

3 Выполнение самостоятельной работы

1. Создадим текстовый файл lab8-3.asm и напомним программу нахождения наименьшей из 3 целочисленных переменных a, b, c (в соответствии с вариантом 16, полученным при выполнении лабораторной работы №7, значение переменных - 44,74,17).

```
[pilobanova@10 lab08]$ touch lab8-3.asm
```

Рис. 3.1: Создание текстового файла lab8-3.asm.



```
lab8-3.asm
~/work/arch-pc/lab08

4 rem: db 'Результат:',0h
5 A dd '44'
6 B dd '74'
7 C dd '17'
8
9 section .bss
10 min resb 10
11
12 section .text
13 global _start
14 _start:
15
16 mov ecx,[A]
17 mov [min],ecx
18
19 cmp ecx,[B]
20 jl check_C
21 mov ecx,[B]
22 mov [min],ecx
23
24 check_C:
25
26 mov eax,C
27 call atoi
28 mov [C],eax
29
30 mov ecx,[min]
31 cmp ecx,[C]
32 jl fin
33 mov ecx,[C]
34 mov [min],ecx
35
36 fin:
37
38 mov eax,rem
39 call sprint
40 mov eax,[min]
41 call iprintf
42
43 call quit
```

Рис. 3.2: Текст программы в файле lab8-3.asm.

2. Создадим исполняемый файл и запустим его.

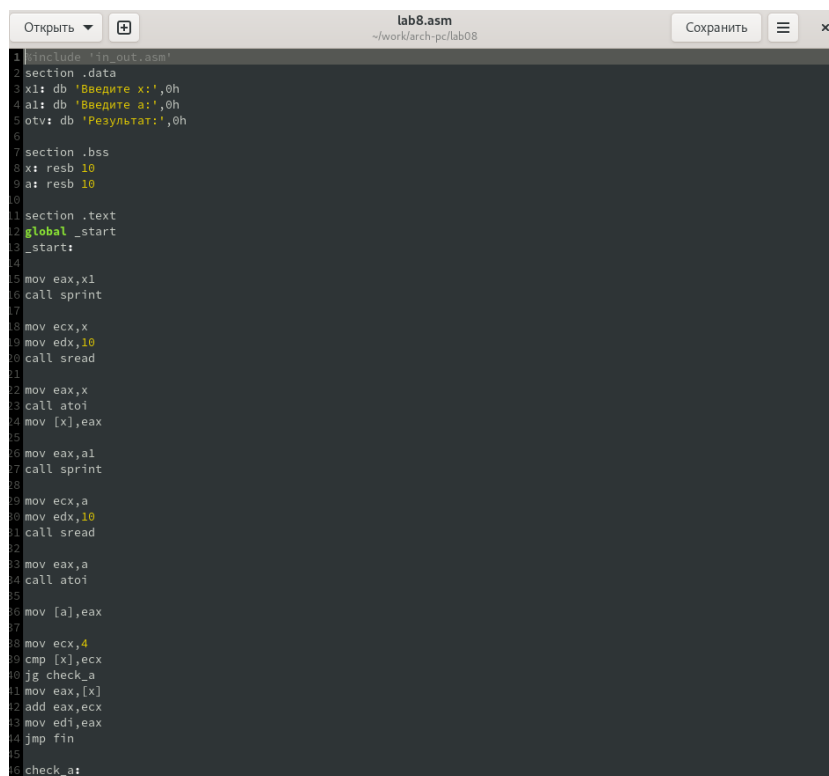
```
[pilobanova@fedora lab08]$ nasm -f elf lab8-3.asm
[pilobanova@fedora lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o
[pilobanova@fedora lab08]$ ./lab8-3
Результат:17
```

Рис. 3.3: Создание исполняемого файла и его запуск.

3. Создадим текстовый файл lab8.asm и напишем программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений (в соответствии с вариантом 16 функцией является система уравнений $x + 4$, $x < 4$ ax , $x \geq 4$).

```
[pilobanova@10 lab08]$ touch lab8-4.asm
```

Рис. 3.4: Создание текстового файла lab8.asm.



```
1 include "in_out.asm"
2 section .data
3 x1: db "Введите x:",0h
4 a1: db "Введите a:",0h
5 otv: db "Результат:",0h
6
7 section .bss
8 x: resb 10
9 a: resb 10
10
11 section .text
12 global _start
13 _start:
14
15 mov eax,x1
16 call sprint
17
18 mov ecx,x
19 mov edx,10
20 call sread
21
22 mov eax,x
23 call atoi
24 mov [x],eax
25
26 mov eax,a1
27 call sprint
28
29 mov ecx,a
30 mov edx,10
31 call sread
32
33 mov eax,a
34 call atoi
35
36 mov [a],eax
37
38 mov ecx,4
39 cmp [x],ecx
40 jg check_a
41 mov eax,[x]
42 add eax,ecx
43 mov edi,eax
44 jmp fin
45
46 check_a:
```

Рис. 3.5: *Текст программы в файле lab8.asm

4. Создадим исполняемый файл и запустим его.

```
[pilobanova@fedora lab08]$ nasm -f elf lab8.asm  
[pilobanova@fedora lab08]$ ld -m elf_i386 -o lab8 lab8.o  
[pilobanova@fedora lab08]$ ./lab8  
Введите x:7  
Введите a:1  
Результат:7  
[pilobanova@fedora lab08]$ ./lab8  
Введите x:1  
Введите a:1  
Результат:5  
[pilobanova@fedora lab08]$
```

Рис. 3.6: Создание исполняемого файла и его запуск.

4 Вывод

Я научилась работать с командами условного и безусловного перехода, писать программы с использованием переходов, а также узнала назначение и структуру файла листинга.