

Шаблон отчёта по лабораторной работе

Простейший вариант

Дмитрий Сергеевич Кулябов

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выполнение лабораторной работы	7
3.1	Реализация переходов в NASM	7
4	Задание для самостоятельной работы	14
5	Выводы	17

Список иллюстраций

3.1	Создал каталог	7
3.2	Создал lab7-1.asm	7
3.3	Заполнил lab7-1	8
3.4	Запуск lab7-1	8
3.5	Изменил текст lab7-1	9
3.6	Запуск измененного lab7-1	9
3.7	Создание lab7-2.asm	9
3.8	Ввод кода в lab7-2.asm	10
3.9	Запуск lab7-2.asm	10
3.10	Компиляция с листингом	10
3.11	Листинг файл	11
3.12	Ошибка в lab7-2	12
3.13	Запуск lab7-2.lst	12
3.14	Измененный листинг	13
4.1	Создал lab7-3.asm	14
4.2	Программа hw1	15
4.3	Запуск hw1	15
4.4	Программма hw2	15
4.5	Запуск hw2	16

Список таблиц

1 Цель работы

Изучение команд для условных и безусловных переходов. Освоение навыков программирования с применением переходов. Ознакомление с назначением и структурой файла с листингом.

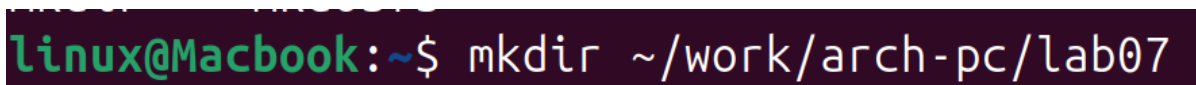
2 Выполнение лабораторной работы

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

3 Выполнение лабораторной работы

3.1 Реализация переходов в NASM

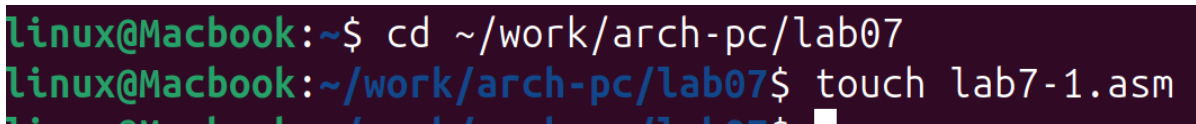
1. Создал каталог для программ перешел в него и создал файл lab7-1.asm (рис. 3.1)



```
linux@Macbook:~$ mkdir ~/work/arch-pc/lab07
```

Рис. 3.1: Создал каталог

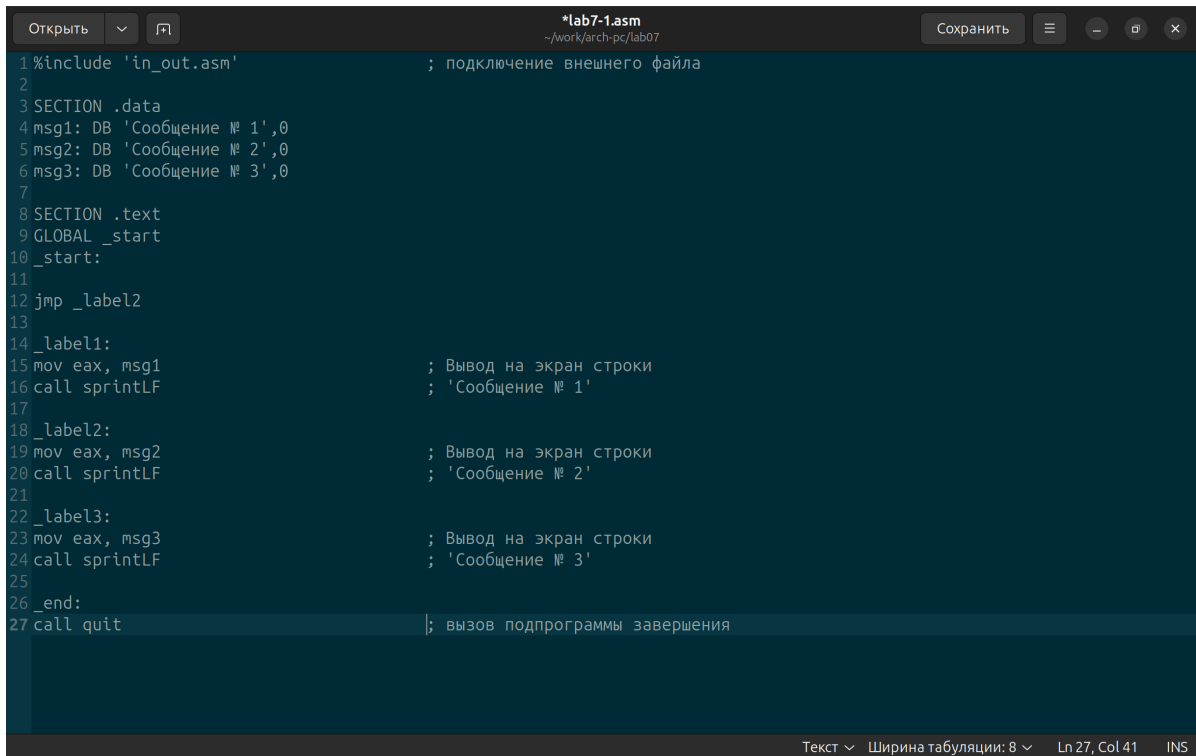
2. перешел в него и создал файл lab7-1.asm (рис. 3.2)



```
linux@Macbook:~$ cd ~/work/arch-pc/lab07
linux@Macbook:~/work/arch-pc/lab07$ touch lab7-1.asm
```

Рис. 3.2: Создал lab7-1.asm

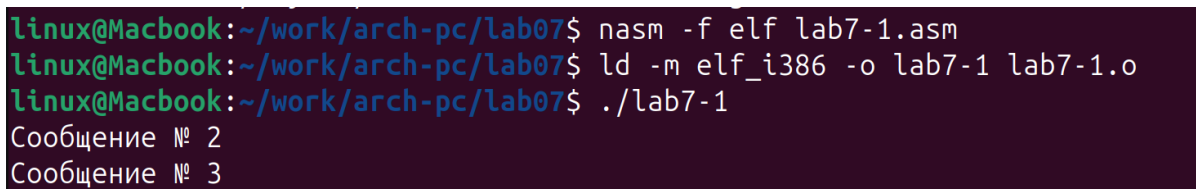
3. Заполнил lab7-1.asm (рис. 3.3)



```
1 %include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 1'
17
18 _label2:
19 mov eax, msg2 ; Вывод на экран строки
20 call sprintf ; 'Сообщение № 2'
21
22 _label3:
23 mov eax, msg3 ; Вывод на экран строки
24 call sprintf ; 'Сообщение № 3'
25
26 _end:
27 call quit ; вызов подпрограммы завершения
```

Рис. 3.3: Заполнил lab7-1

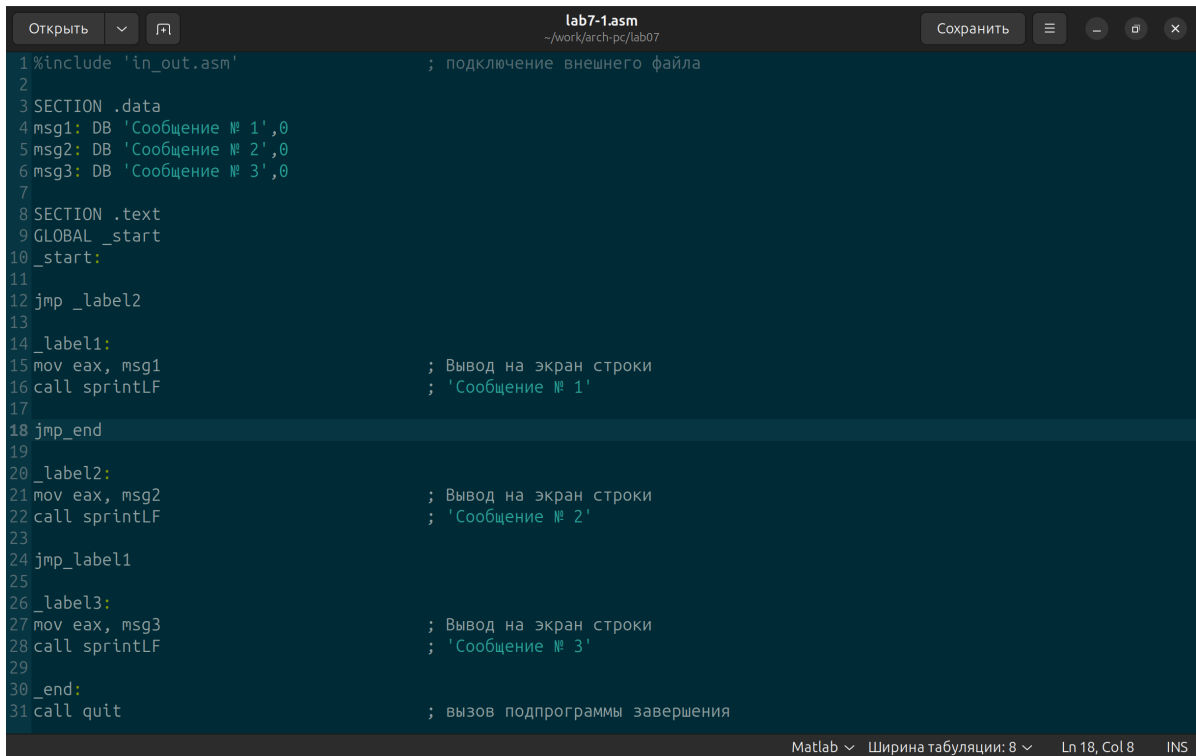
4. Создал исполняемый файл и запустил его (рис. 3.4)



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

Рис. 3.4: Запуск lab7-1

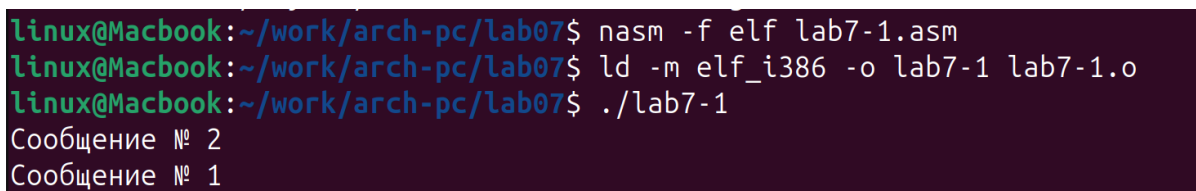
5. Изменил текст программы (рис. 3.5)



```
1 %include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
16 call sprintf ; 'Сообщение № 1'
17
18 jmp_end
19
20 _label2:
21 mov eax, msg2 ; Вывод на экран строки
22 call sprintf ; 'Сообщение № 2'
23
24 jmp_label1
25
26 _label3:
27 mov eax, msg3 ; Вывод на экран строки
28 call sprintf ; 'Сообщение № 3'
29
30 _end:
31 call quit ; вызов подпрограммы завершения
```

Рис. 3.5: Изменил текст lab7-1

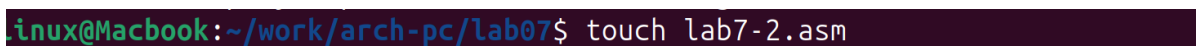
6. Создал исполняемый файл и запустил его (рис. 3.6)



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

Рис. 3.6: Запуск измененного lab7-1

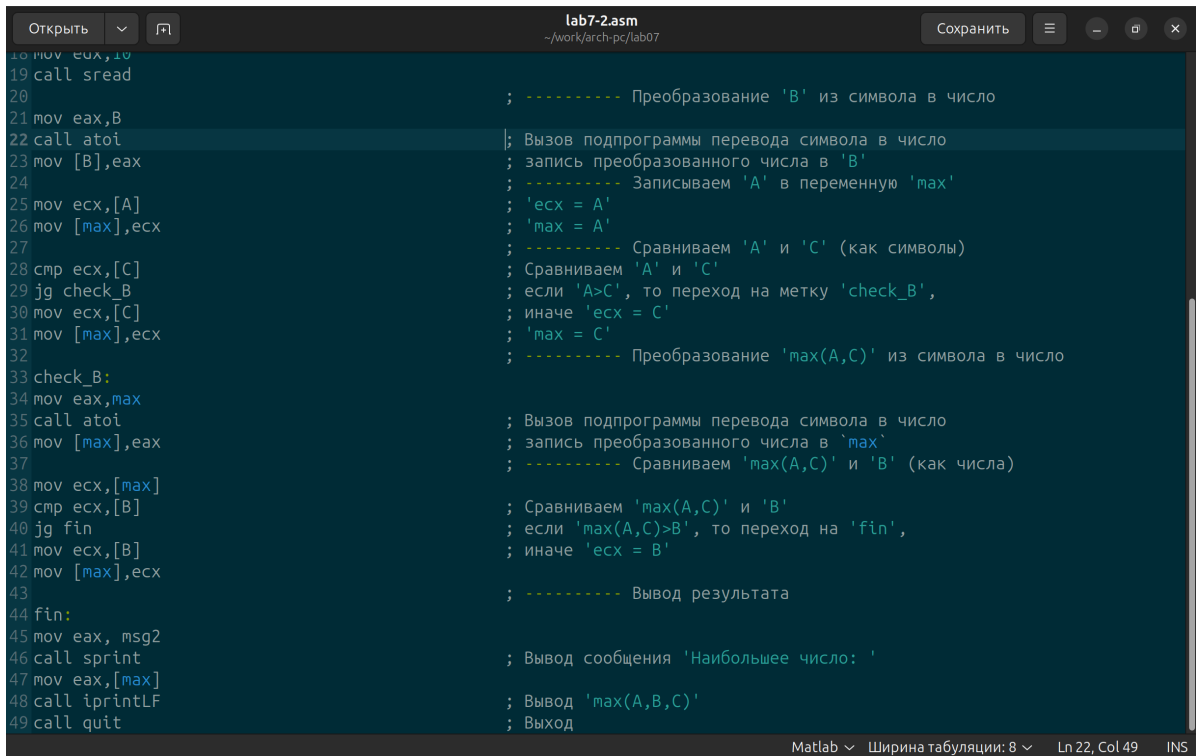
7. Создал файл lab7-2.asm (рис. 3.7)



```
linux@Macbook:~/work/arch-pc/lab07$ touch lab7-2.asm
```

Рис. 3.7: Создание lab7-2.asm

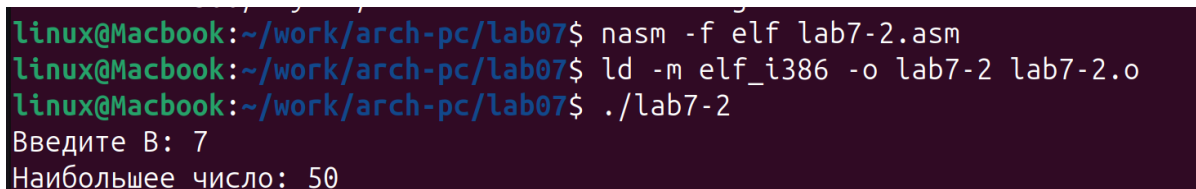
8. Изучил текст программы и ввел в lab7-2.asm (рис. 3.8)



```
10 mov ecx,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
```

Рис. 3.8: Ввод кода в lab7-2.asm

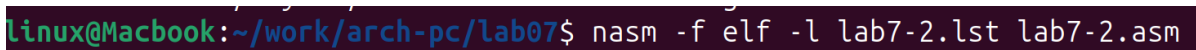
9. Создал исполняемый файл и запустил его (рис. 3.9)



```
linux@Macbook:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
linux@Macbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
linux@Macbook:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 7
Наибольшее число: 50
```

Рис. 3.9: Запуск lab7-2.asm

10. Создал файл листинга для программы из файла lab7-2.asm (рис. 3.10)



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

Рис. 3.10: Компиляция с листингом

11. Открыл файл с листингом (рис. 3.11)

```
lab7-2.lst
~/work/arch-pc/lab07
Сохранить

1 1 %include 'in_out.asm'
2 1 <1> ;----- slen | -----
3 2 <1> ; Функция вычисления длины сообщения
4 3 <1> slen:
5 4 00000000 53 <1> push ebx
6 5 00000001 89C3 <1> mov ebx, eax
7 6 <1>
8 7 <1> nextchar:
9 8 00000003 803800 <1> cmp byte [eax], 0
10 9 00000006 7403 <1> jz finished
11 10 00000008 40 <1> inc eax
12 11 00000009 EBF8 <1> jmp nextchar
13 12 <1>
14 13 <1> finished:
15 14 0000000B 29D8 <1> sub eax, ebx
16 15 0000000D 5B <1> pop ebx
17 16 0000000E C3 <1> ret
18 17 <1>
19 18 <1>
20 19 <1> ;----- sprint -----
21 20 <1> ; Функция печати сообщения
22 21 <1> ; входные данные: mov eax,<message>
23 22 <1> sprint:
24 23 0000000F 52 <1> push edx
25 24 00000010 51 <1> push ecx
26 25 00000011 53 <1> push ebx
27 26 00000012 50 <1> push eax
28 27 00000013 E8E8FFFFFF <1> call slen
29 28 <1>
30 29 00000018 89C2 <1> mov edx, eax
31 30 0000001A 58 <1> pop eax
32 31 <1>
```

Рис. 3.11: Листинг файл

12. Изменил файл lab7-2.asm (рис. 3.12)

```

10 mov ecx,10
19 call sread
20 ; ----- Преобразование 'B' из символа в число
21 mov eax,B
22 call atoi
23 mov [B],eax
24 ; ----- Записываем 'A' в переменную 'max'
25 mov ecx,[A]
26 ; 'ecx = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C]
29 jg check_B
30 mov ecx,[C]
31 mov [max],ecx
32 ; 'max = C'
33 check_B:
34 mov eax,max
35 call atoi
36 mov [max],eax
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 mov ecx,[max]
39 cmp ecx,[B]
40 jg fin
41 mov ecx,[B]
42 mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 mov eax, msg2
46 call sprint
47 mov eax,[max]
48 call iprintLF
49 call quit
; Вывод сообщения 'Наибольшее число: '
; Вывод 'max(A,B,C)'
; Выход

```

Рис. 3.12: Ошибка в lab7-2

9. Создал исполняемый файл и запустил его (рис. 3.13)

```

linux@Macbook:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
linux@Macbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
linux@Macbook:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 7
Наибольшее число: 50

```

Рис. 3.13: Запуск lab7-2.lst

13. Создание файла листинга для измененной программы и нахождение изменений (рис. 3.14)

```

Открыть  ▾  lab7-2.lst  Сохранить  ▮  -  □  x
~/work/arch-pc/lab07

1  1  %include 'in_out.asm'
2  1  <1> ;----- slen | -----
3  2  <1> ; Функция вычисления длины сообщения
4  3  <1> slen:
5  4  00000000 53  <1> push ebx
6  5  00000001 89C3 <1> mov ebx, eax
7  6  <1>
8  7  <1> nextchar:
9  8  00000003 803800 <1> cmp byte [eax], 0
10 9  00000006 7403 <1> jz finished
11 10 00000008 40 <1> inc eax
12 11 00000009 EBF8 <1> jmp nextchar
13 12 <1>
14 13 <1> finished:
15 14 0000000B 29D8 <1> sub eax, ebx
16 15 0000000D 5B <1> pop ebx
17 16 0000000E C3 <1> ret
18 17 <1>
19 18 <1>
20 19 <1> ;----- sprint -----
21 20 <1> ; Функция печати сообщения
22 21 <1> ; входные данные: mov eax,<message>
23 22 <1> sprint:
24 23 0000000F 52 <1> push edx
25 24 00000010 51 <1> push ecx
26 25 00000011 53 <1> push ebx
27 26 00000012 50 <1> push eax
28 27 00000013 E8E8FFFFFF <1> call slen
29 28 <1>
30 29 00000018 89C2 <1> mov edx, eax
31 30 0000001A 58 <1> pop eax
32 31 <1>

Текст ▾  Ширина табуляции: 8 ▾  Ln 2, Col 65  INS

```

Рис. 3.14: Измененный листинг

4 Задание для самостоятельной работы

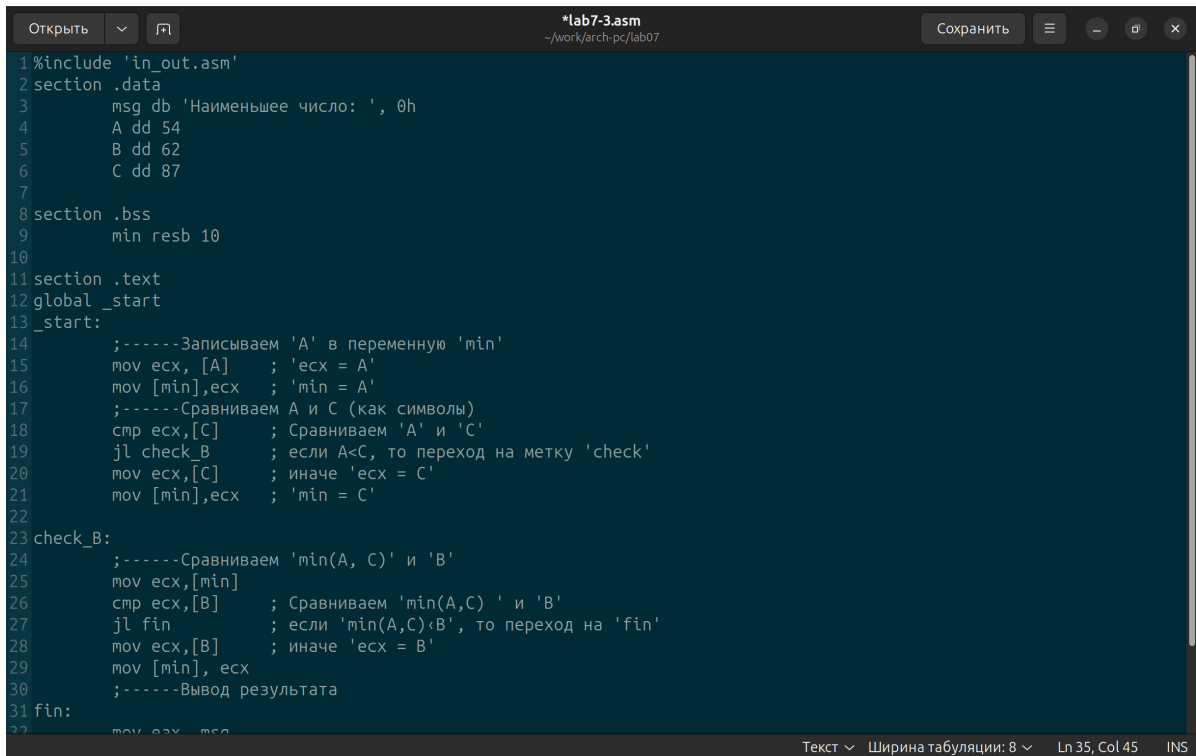
создал файл lab7-3.asm (рис. ??)



```
linux@Macbook:~/work/arch-pc/lab07$ touch lab7-3.asm
```

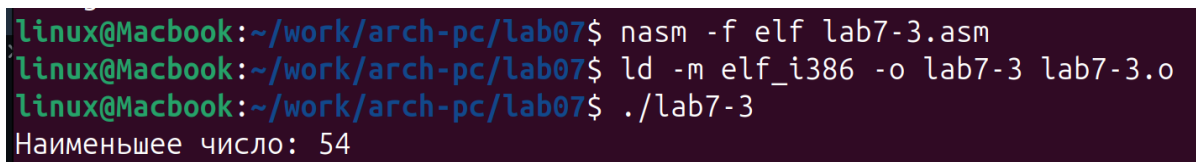
Рис. 4.1: Создал lab7-3.asm

2. Программа для нахождения наименьшего из 3 целочисленных переменных (рис. 4.2) (рис. 4.3)



```
1 %include 'in_out.asm'
2 section .data
3     msg db 'Наименьшее число: ', 0h
4     A dd 54
5     B dd 62
6     C dd 87
7
8 section .bss
9     min resb 10
10
11 section .text
12 global _start
13 _start:
14     ;-----Записываем 'A' в переменную 'min'
15     mov ecx, [A] ; 'ecx = A'
16     mov [min],ecx ; 'min = A'
17     ;-----Сравниваем A и C (как символы)
18     cmp ecx,[C] ; Сравниваем 'A' и 'C'
19     jl check_B ; если A<C, то переход на метку 'check'
20     mov ecx,[C] ; иначе 'ecx = C'
21     mov [min],ecx ; 'min = C'
22
23 check_B:
24     ;-----Сравниваем 'min(A, C)' и 'B'
25     mov ecx,[min]
26     cmp ecx,[B] ; Сравниваем 'min(A,C)' и 'B'
27     jl fin ; если 'min(A,C)<B', то переход на 'fin'
28     mov ecx,[B] ; иначе 'ecx = B'
29     mov [min],ecx
30     ;-----Вывод результата
31 fin:
32     mov eax, msg
```

Рис. 4.2: Программа hw1



```
linux@Macbook:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
linux@Macbook:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
linux@Macbook:~/work/arch-pc/lab07$ ./lab7-3
Наименьшее число: 54
```

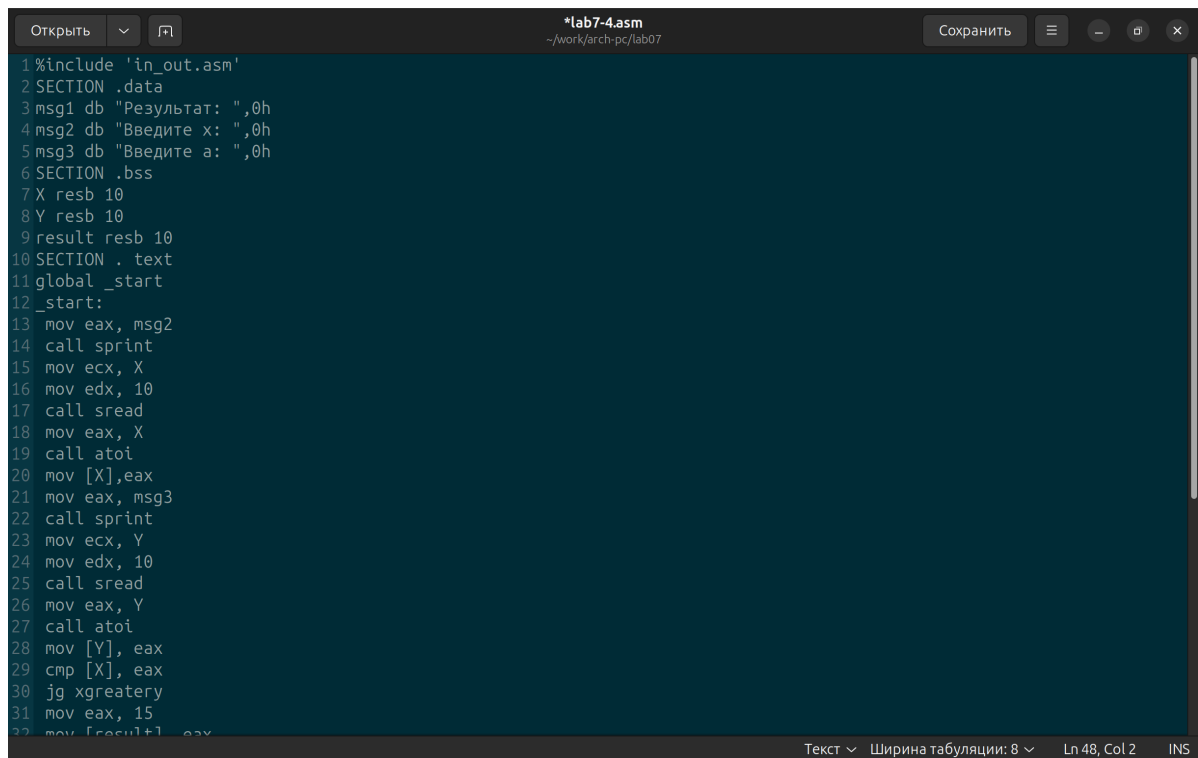
Рис. 4.3: Запуск hw1

3. Программа для вычислений по формулам $2(x-a)$ при $x>a$ и 15, при $x\leq a$ (Вариант №5) (рис. 4.4) (рис. 4.5)



```
linux@Macbook:~/work/arch-pc/lab07$ touch lab7-4.asm
```

Рис. 4.4: Программма hw2



```
1 %include 'in_out.asm'
2 SECTION .data
3 msg1 db "Результат: ",0h
4 msg2 db "Введите x: ",0h
5 msg3 db "Введите a: ",0h
6 SECTION .bss
7 X resb 10
8 Y resb 10
9 result resb 10
10 SECTION .text
11 global _start
12 _start:
13 mov eax, msg2
14 call sprint
15 mov ecx, X
16 mov edx, 10
17 call sread
18 mov eax, X
19 call atoi
20 mov [X],eax
21 mov eax, msg3
22 call sprint
23 mov ecx, Y
24 mov edx, 10
25 call sread
26 mov eax, Y
27 call atoi
28 mov [Y], eax
29 cmp [X], eax
30 jg xgreatery
31 mov eax, 15
32 mov [result], eax
```

Текст ▾ Ширина табуляции: 8 ▾ Ln 48, Col 2 INS

Рис. 4.5: Запуск hw2

5 Выводы

Я изучил команды для условных и безусловных переходов. Я освоил навыки программирования с применением переходов и ознакомился с назначением и структурой файла с листингом.