

Отчет по лабораторной работе No.7

Дисциплины: Архитектура компьютера

Нджову Нелиа

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	17
	Список литературы	18

Список иллюстраций

3.1	Рис 1	7
3.2	Рис 2	7
3.3	Рис 3	7
3.4	Рис 4	8
3.5	Рис 5	8
3.6	Рис 6	9
3.7	Рис 7	9
3.8	Рис 8	10
3.9	Рис 9	10
3.10	Рис 10	10
3.11	Рис 11	11
3.12	Рис 12	12
3.13	Рис 13	12
3.14	Рис 14	13
3.15	Рис 15	14
3.16	Рис 16	14
3.17	Рис 17	14
3.18	Рис 18	15
3.19	Рис 19	15

Список таблиц

1 Цель работы

Целью данной лабораторной работы является изучение команд условного и безусловного перехода. Также приобрести навыки написания программ с использованием переходов и понимания назначения и структуры листинга файлов.

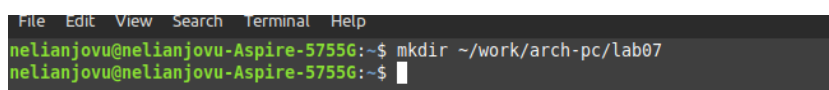
2 Задание

1. Реализация переходов в NASM
2. Изучение структуры файлы листинга
3. Выполнение заданий для самостоятельной работы

3 Выполнение лабораторной работы

1. Реализация переходов в NASM

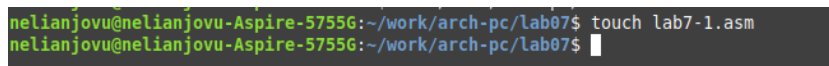
Я создам каталог для программ лабораторных работ 7 под названием lab07 в каталоге ~/work/arch-pc с помощью команды mkdir(рис 1)



```
File Edit View Search Terminal Help
nelianjovu@nelianjovu-Aspire-5755G:~$ mkdir ~/work/arch-pc/lab07
nelianjovu@nelianjovu-Aspire-5755G:~$
```

Рис. 3.1: Рис 1

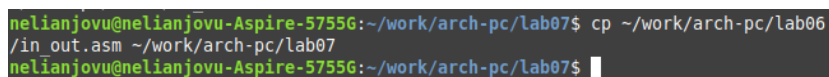
В созданном каталоге я создам файл с именем lab7-1.asm с помощью сенсорной команды(рис 2).



```
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ touch lab7-1.asm
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$
```

Рис. 3.2: Рис 2

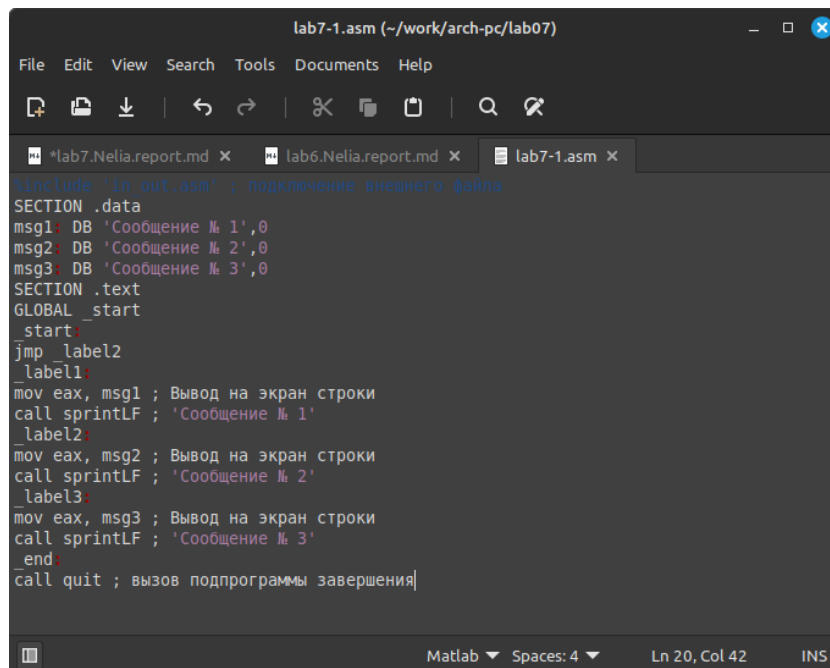
Я скопирую файл in_out.asm в текущий каталог с помощью команды cp, потому что буду использовать его в программах(рис 3)



```
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ cp ~/work/arch-pc/lab06
/in_out.asm ~/work/arch-pc/lab07
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$
```

Рис. 3.3: Рис 3

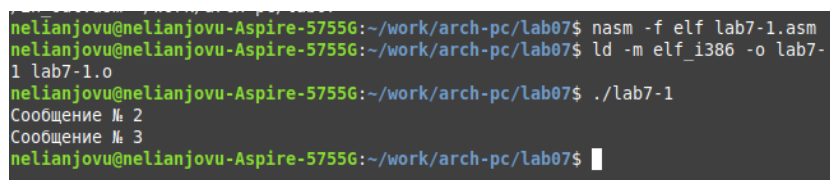
Я открою и введу заданный текст программы в созданный мною файл. Программа использует инструкцию jmp. Инструкция jmp в NASM используется для реализации безусловных переходов(рис 4)



```
lab7-1.asm (~/work/arch-pc/lab07)
File Edit View Search Tools Documents Help
*lab7.Nelia.report.md x lab6.Nelia.report.md x lab7-1.asm x
%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 sprintf ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 3.4: Рис 4

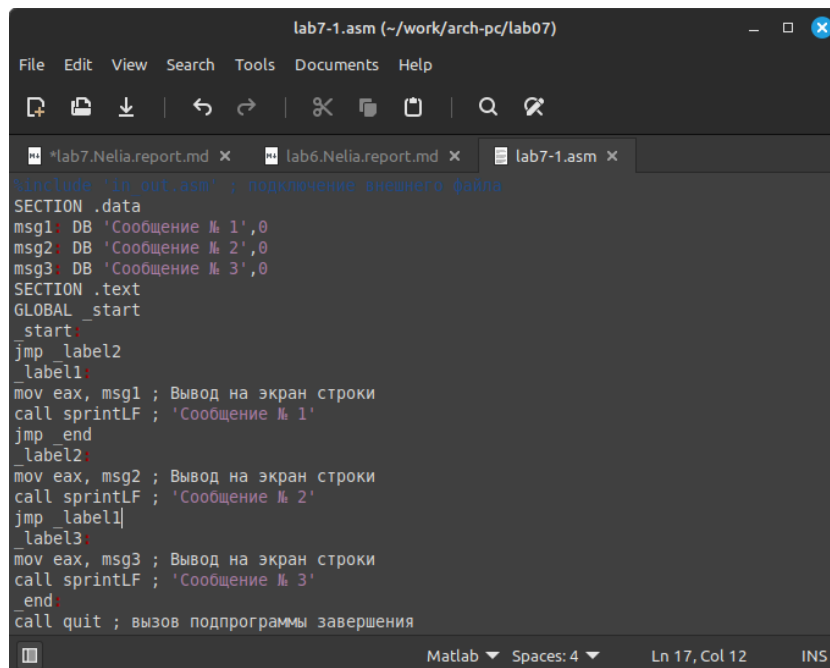
Я создам исполняемый файл и запущу его(рис 5).Как мы видим, использование инструкции `jmp _label2` меняет порядок выполнения инструкций и позволяет выполнять инструкции, начиная с метки `_label2`, пропуская вывод первого сообщения.



```
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$
```

Рис. 3.5: Рис 5

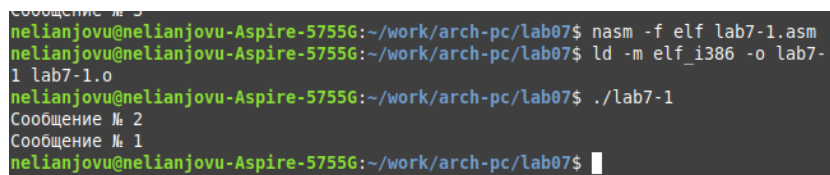
Инструкция `jmp` позволяет прыгать не только вперед, но и назад.Я изменю программу, добавив «`jmp _label1`» после `label2` и «`jmp _end`» после `label1`, чтобы она сначала отображала «Сообщение 2», затем «Сообщение 1» и завершала работу без отображения «Сообщения 3».(рис 6)



```
lab7-1.asm (~/work/arch-pc/lab07)
File Edit View Search Tools Documents Help
*lab7.Nelia.report.md x lab6.Nelia.report.md x lab7-1.asm x
%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 sprintf ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
Matlab Spaces: 4 Ln 17, Col 12 INS
```

Рис. 3.6: Рис 6

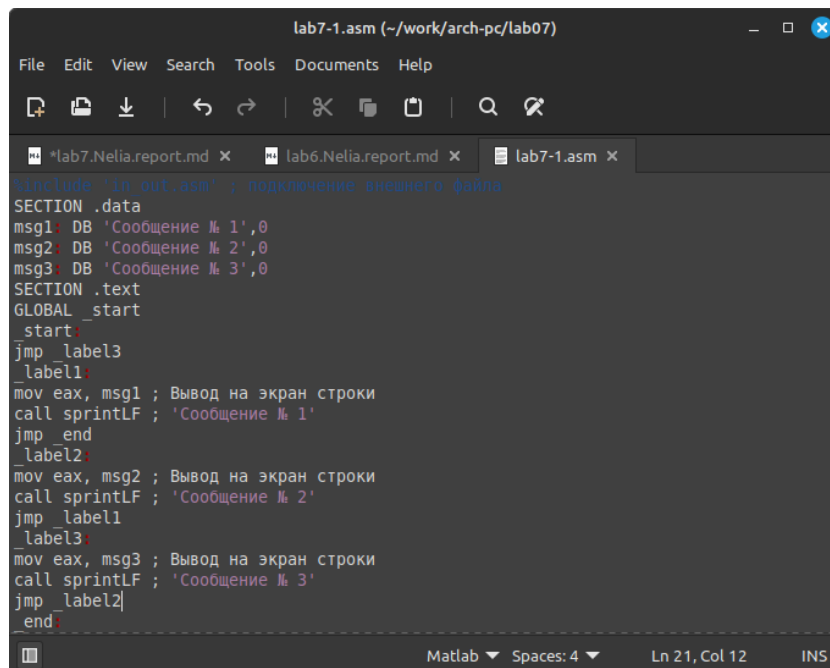
Я создам исполняемый файл, запущу его и проверю, работает ли программа корректно(рис 7)



```
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$
```

Рис. 3.7: Рис 7

Теперь я изменю программу, добавив «jmp_label3» перед label1, «jmp_label1» после label2, «jmp_label2» после label3 и «jmp_end» после label1, чтобы она сначала отображала «Сообщение 3», затем «Сообщение 2» и, наконец, «Сообщение 1»(рис 8)



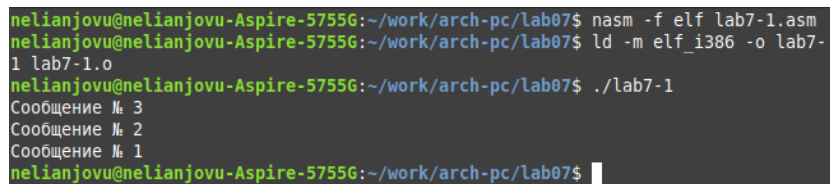
The screenshot shows a text editor window titled 'lab7-1.asm (~/.work/arch-pc/lab07)'. The menu bar includes File, Edit, View, Search, Tools, Documents, and Help. The toolbar contains icons for file operations and editing. The code is as follows:

```
%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 _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
jmp _label2
_end:
```

The status bar at the bottom indicates 'Matlab', 'Spaces: 4', 'Ln 21, Col 12', and 'INS'.

Рис. 3.8: Рис 8

Я создам исполняемый файл, запущу его и проверю, работает ли программа корректно(рис 9)

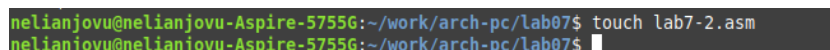


The screenshot shows a terminal window with the following commands and output:

```
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$
```

Рис. 3.9: Рис 9

Я создам новый файл с именем lab7-2.asm в каталоге ~/.work/arch-pc/lab07(рис 10)



The screenshot shows a terminal window with the following commands:

```
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$ touch lab7-2.asm
nelianjovu@nelianjovu-Aspire-57556:~/work/arch-pc/lab07$
```

Рис. 3.10: Рис 10

Я скопирую данную текстовую программу в только что созданный файл.При


```

nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 4
Наибольшее число: 50
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 60
Наибольшее число: 60
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$

```

Рис. 3.12: Рис 12

2. Изучение структуры файлы листинга

Указав ключ `-l` и имя файла листинга в командной строке. Файл листинга программы я создам из файла `lab7-2.asm`(рис 13)

```

наибольшее число: 60
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$

```

Рис. 3.13: Рис 13

Теперь я открою файл листинга `lab7-2.lst` с помощью текстового редактора(рис 14)

```
lab7-2.lst (~/.work/arch-pc/lab07)
File Edit View Search Tools Documents Help
lab7-2.asm x *lab7.Nelia.report.md x lab7-2.lst 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
Plain Text Spaces: 4 Ln 1, Col 1 INS
```

Рис. 3.14: Рис 14

строка 19 «вызов *atoi*» меняет *B* с арифметического символа на число
строка 174 «*msg2 db "Наибольшее число: ",0h*» отображает текст «Наибольшее
число:» на экране
строка 173 «*msg1 db 'Введите B:',0h*» отображает текст «Введите *B*:» на экране
Я открою файл *lab7-2.asm* и удалю один из операндов, затем выполню широко-
вещательную рассылку, чтобы получить файл листинга(рис 15 и 16)

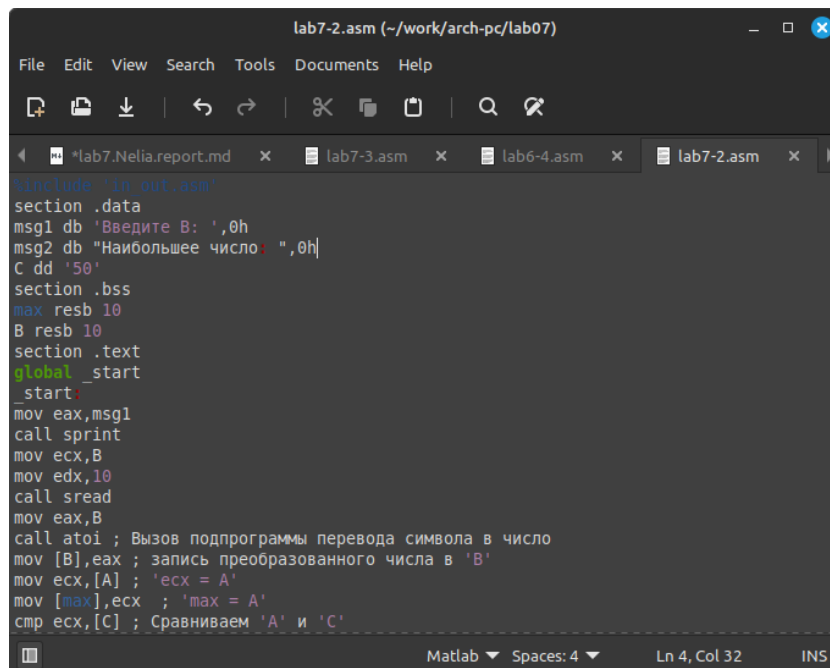


Рис. 3.15: Рис 15

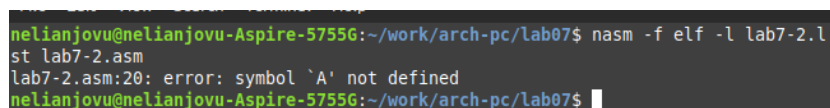


Рис. 3.16: Рис 16

При создании файла листинга выдала ошибку, так как в файле lab7-2.asm программа неверна

3. Выполнение заданий для самостоятельной работы

С помощью touch команды я создам новый файл lab7-3.asm(рис 17)

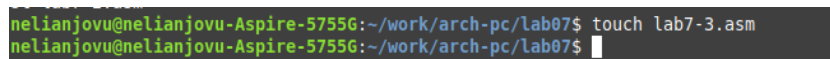


Рис. 3.17: Рис 17

В созданном мной файле я напишу программу, которая будет находить минимальное значение среди трех чисел a, b и c. Я получу значения a, b и c из варианта, который я получил при выполнении лабораторной работы 6(рис 18)

```

lab7-3.asm (~/work/arch-pc/lab07)
File Edit View Search Tools Documents Help
*lab7.Nelia.report.md x lab7-3.asm x lab.asm x
msg1 db "Наибольшее число: ",0h
A dd '81'
B dd '22'
C dd '72'
section .bss
min resb 10
section .text
global _start
_start:
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'max = A'
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jl check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'max = C'
check_B:
mov eax,min
call atoi ; Вызов подпрограммы перевода символа в число
mov [min],eax ; запись преобразованного числа в 'max'

```

Рис. 3.18: Рис 18

Теперь я создам исполняемый файл и запущу его, чтобы посмотреть, даст ли он правильный ответ(рис 19)

```

nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$ ./lab7-3
Наименьше число: 22
nelianjovu@nelianjovu-Aspire-5755G:~/work/arch-pc/lab07$

```

Рис. 3.19: Рис 19

```

#include 'in_out.asm'

section .data
msg1 db "Наименьше число: ",0h
A dd '81'
B dd '22'
C dd '72'

section .bss

```

```

min resb 10
section .text
global _start
_start:
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'max = A'
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jl check_B ; если 'A<C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'max = C'
check_B:
mov eax,min
call atoi ; Вызов подпрограммы перевода символа в число
mov [min],eax ; запись преобразованного числа в 'max'
mov ecx,[min]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jl fin ; если 'max(A,C)<B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [min],ecx
fin:
mov eax, msg1
call sprintf ; Вывод сообщения 'Наибольшее число: '
mov eax,[min]
call fprintf ; Вывод 'min(A,B,C)'
call quit ; Выход

```


4 Выводы

Выполняя эту лабораторную работу, я узнал об условных и безусловных командах перехода. Также приобрел навыки написания программ с использованием переходов и понял назначение и структуру листингов файлов.

Список литературы

Архитектура ЭВМ