

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЁТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №7

Дисциплина: архитектура компьютера

Студент: Ванюшкина Т.В.

Группа: НКАбд-01-24

Студ.билет: 1132246713

МОСКВА

2024г

СОДЕРЖАНИЕ

1. Цель работы.....	3
2. Теоретическое введение.....	4
3. Выполнение лабораторной работы.....	5-11
3.1 Реализация переходов в NASM	5-7
3.2 Изучение структуры файлы листинга.....	8-9
3.3 Задание для самостоятельной работы.....	10-11
4. Выводы.....	12
5. Список литературы.....	13

1.Цель работы

Целью работы является изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

2.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.

3.Выполнение лабораторной работы

3.1 Реализация переходов в NASM

Создаю каталог для программ лабораторной работы № 7:

```
tatyana@vbox:~$ mkdir -p ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/labs/lab04
tatyana@vbox:~$
```

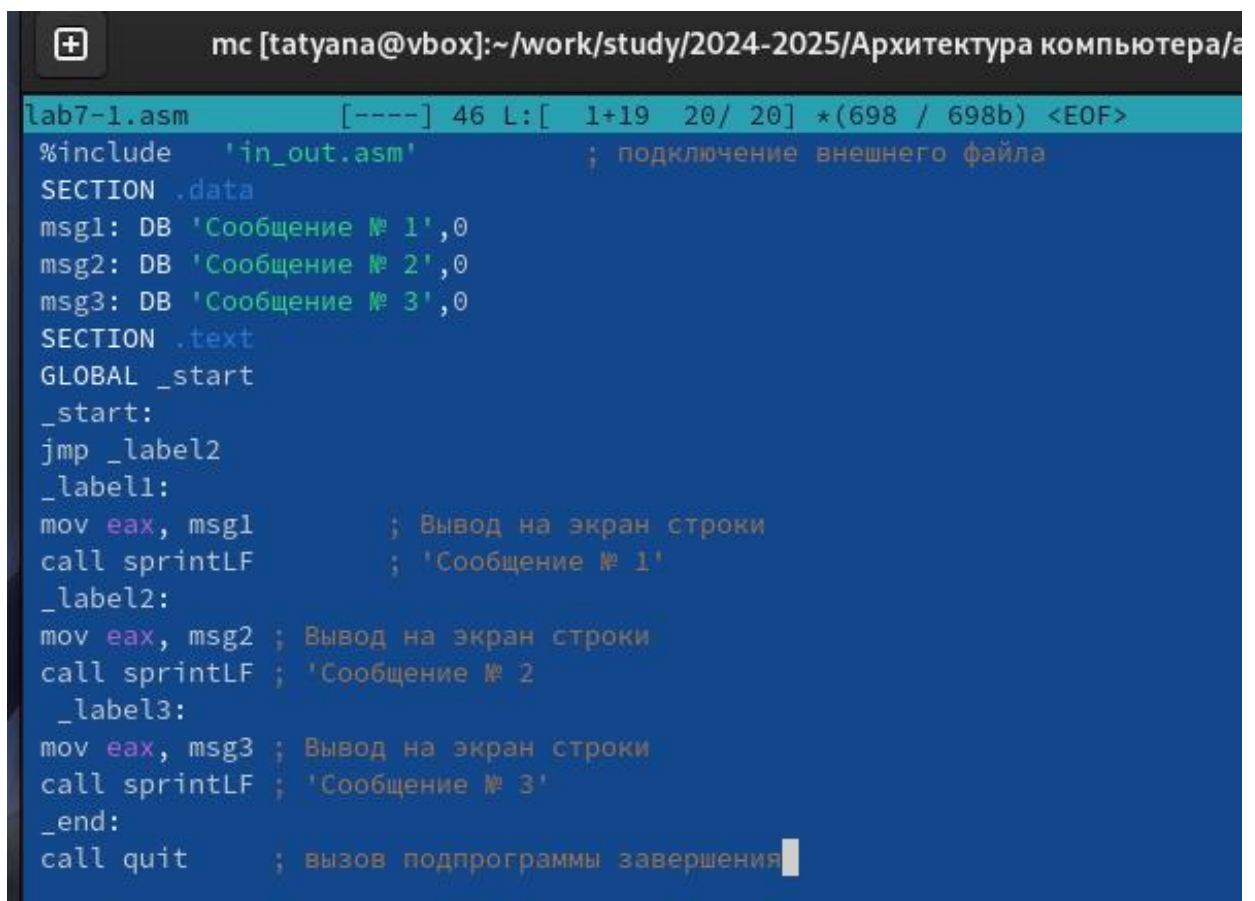
Рис 1: Открытие файла

Перехожу в него и создаю файл lab7-1.asm:

```
tatyana@vbox:~$ cd ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/labs/lab07
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ touch lab7-1.asm
```

Рис 2: Создание файла

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



```
mc [tatyana@vbox]:~/work/study/2024-2025/Архитектура компьютера/a
lab7-1.asm [----] 46 L:[ 1+19 20/ 20] *(698 / 698b) <EOF>
%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 sprintfLF ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис 3: Ввод текста программы

Создаю исполняемый файл и запускаю его:

```
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-1.asm
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
```

Рис 4: Создание и запуск файла

Изменяю текст программы в соответствии с листингом 7.2:

```
lab7-1.asm      [-M--] 47 L:[ 1+21 22/ 22] *(711 / 711b) <EOF>
#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 sprintLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
_end:
call quit      ; вызов подпрограммы завершения
```

Рис 5: Замена текста

Создаю исполняемый файл и проверяю его работу:

```
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-1.asm
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
```

Рис 6: Создание и проверка файла

Создаю файл lab7-2.asm:

```
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ touch lab7-2.asm
```

Рис 7: Создание файла

Изучаю текст программы из листинга 7.3 и ввожу в lab7-2.asm:

```
lab7-2.asm      [----]  0 L: [ 7+19 26/ 49] *(729 /1716b) 0032 0x020
section .bss
max resb 10
B resb 10
section .text
global _start
_start:
;----- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
;----- Ввод 'B'
mov ecx,B
mov edx,10
call sread
;----- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
;----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
;-----Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
;----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в 'max'
;----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jg fin ; если 'max(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [max],ecx
```

Рис 8: Ввод текста программы

Создаю исполняемый файл и проверяю его работу для разных значений B:

```
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-2.asm
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-2
Введите B: 8
Наибольшее число: 50
```

Рис 9: Создание и проверка файла

3.2 Изучение структуры файлы листинга

Создаю файл листинга для программы из файла lab7-2.asm:

```
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$
```

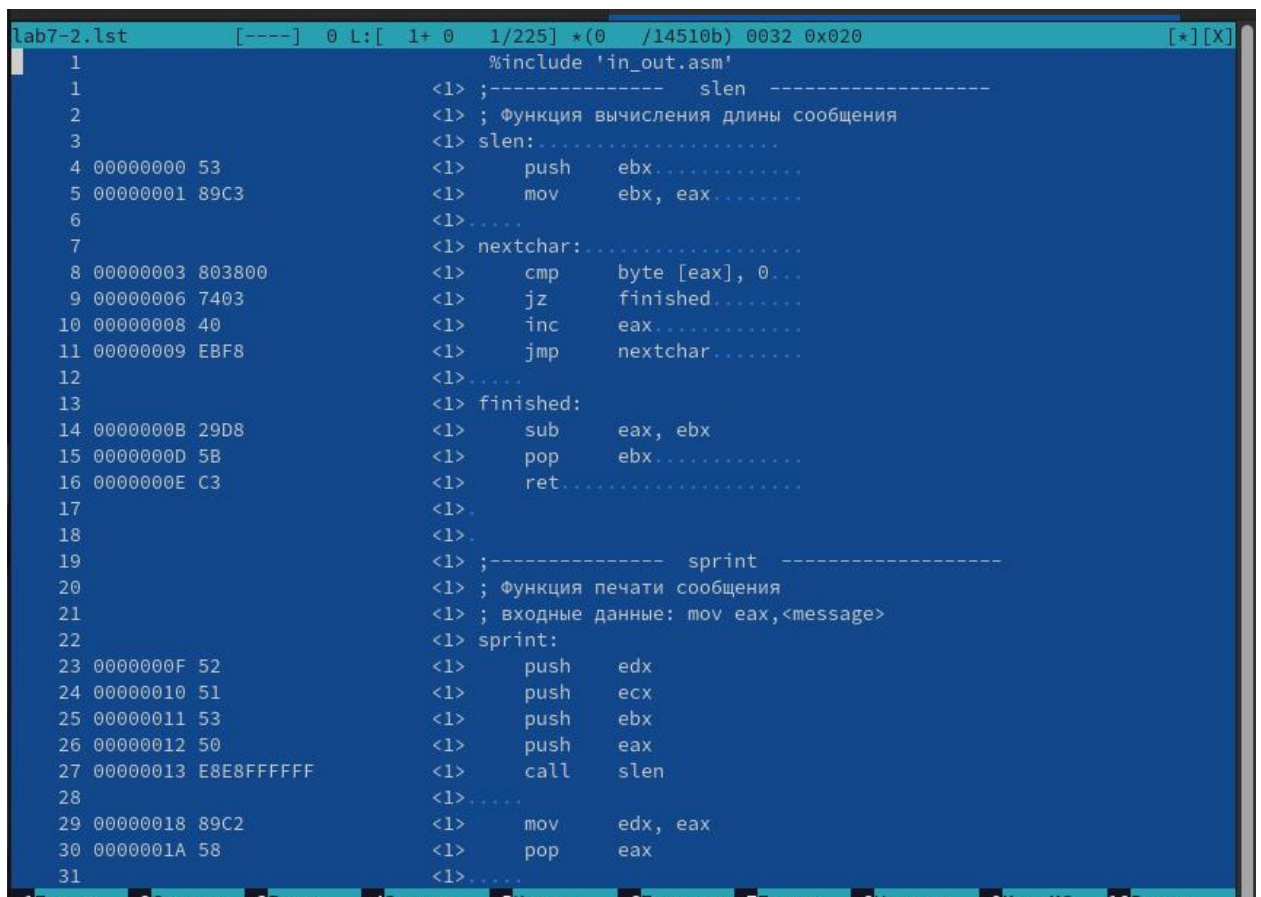
Рис 10: Создание файла

Открываю файл листинга lab7-2 с помощью текстового редактора mcedit:

```
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ mcedit lab7-2.lst
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$
```

Рис 11: Создание файла

Ознакамливаюсь с его форматом и содержимым:



```
lab7-2.lst [----] 0 L: [ 1+ 0 1/225] *(0 /14510b) 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> ;----- 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 12: Файл lab7-2

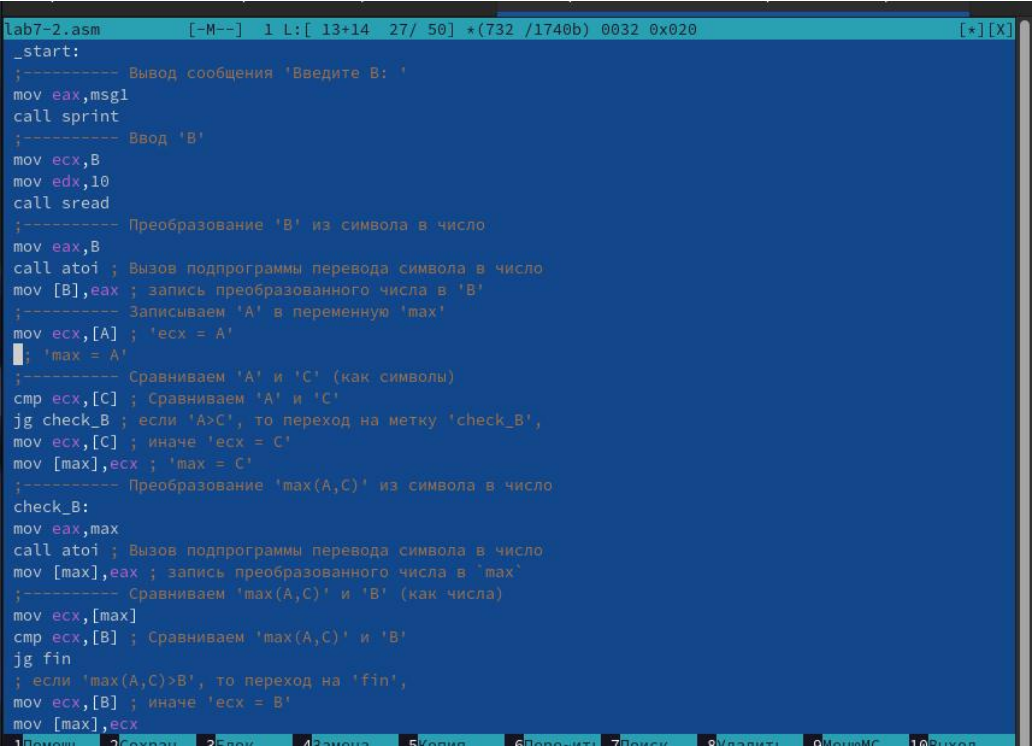
Я объясню значение таких строк как mov ebx, push ebx и pop ebx

функция mov ebx перемещает данные из указанного источника в регистр ebx.
Регистр ebx - это 32-битный регистр общего назначения в архитектуре x86.

функция `push ebx` помещает значение регистра `ebx` в стек. Стек - это область памяти, которая используется для временного хранения данных.

функция `pop ebx` извлекает значение из вершины стека и помещает его в регистр `ebx`

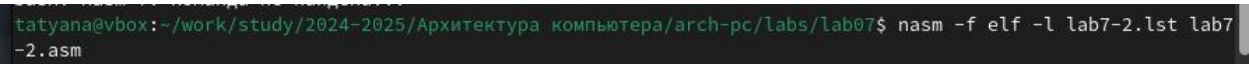
Открываю файл с программой `lab7-2.asm` и удаляю операнд `mov [max],ecx` :



```
lab7-2.asm [-M--] 1 L: [ 13+14 27/ 50] *(732 /1740b) 0032 0x020 [*] [X]
_start:
;----- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
;----- Ввод 'B'
mov ecx,B
mov edx,10
call sread
;----- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
;----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
; 'max = A'
;----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
;----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в 'max'
;----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jg fin
; если 'max(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [max],ecx
```

Рис 13: Удаление операнда

Выполняю трансляцию с получением файла листинга:



```
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
```

Рис 14: Выполнение трансляции

Появился такой файл как `lab7-2.lst`:

.и	Имя	Размер	Дата прав
./..		-ВВЕРХ-	ноя 19 19
/presentation		100	окт 8 20
/report		62	окт 8 20
in_out.asm		3942	ноя 3 23
*lab7-1		9200	ноя 19 18
lab7-1.asm		711	ноя 19 19
lab7-1.o		1456	ноя 19 18
*lab7-2		9240	ноя 20 14
lab7-2.asm		1703	ноя 20 14
lab7-2.lst		14431	ноя 20 14
lab7-2.o		1696	ноя 20 14
*lab7-3		9240	ноя 19 21
lab7-3.asm		1725	ноя 19 21
lab7-3.o		1696	ноя 19 21
*lab7-4		9300	ноя 19 22
lab7-4.asm		9300	ноя 19 22
lab7-4.o		1648	ноя 19 22

Рис 15: Файл `lab7-2.lst`

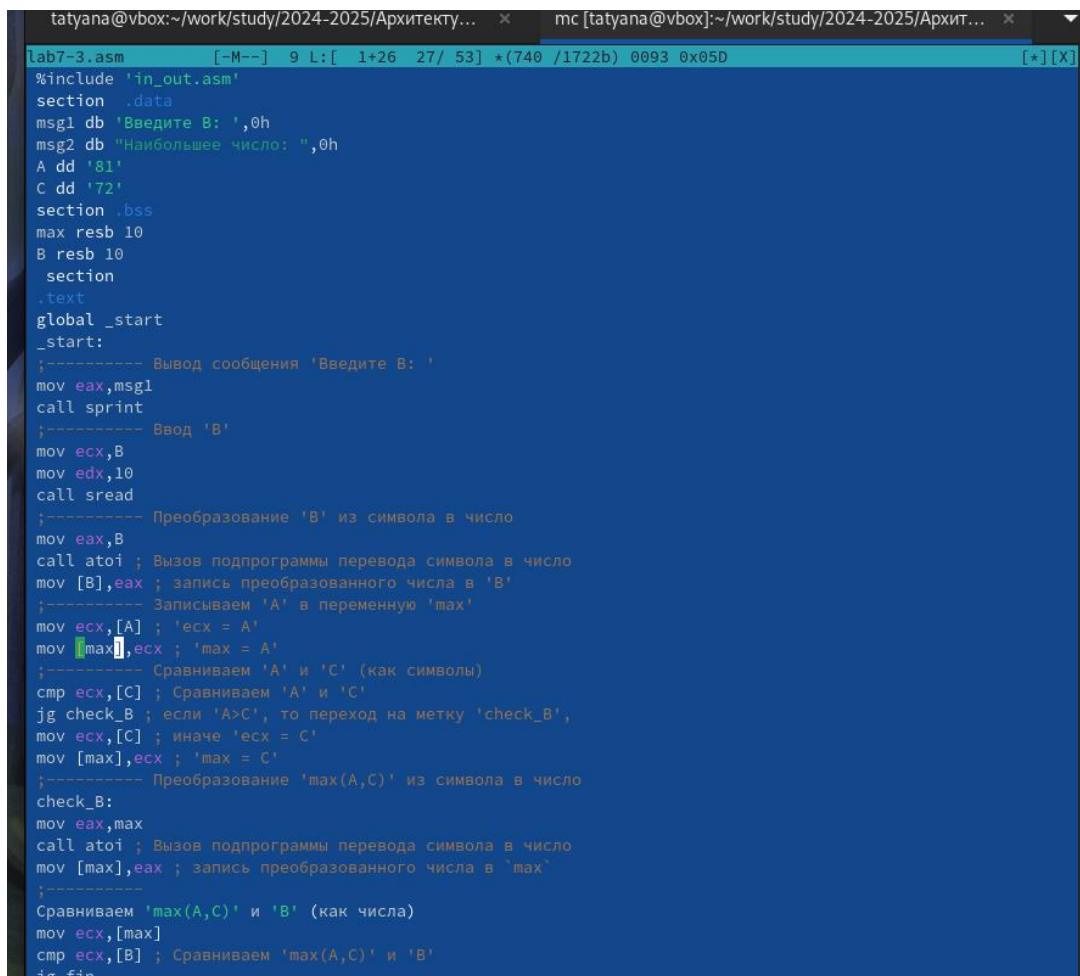
3.3. Задание для самостоятельной работы

Создаю файл lab7-3.asm:

```
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ touch lab7-3.asm
```

Рис 16: Создание файла

Пишу программу нахождения наименьшей из 3 целочисленных переменных a, b, c . Значения переменных беру из табл.7.5 в соответствии с 14 вариантом, полученным при выполнении лабораторной работы №7:



```
lab7-3.asm      [-M--]  9 L:[ 1+26 27/ 53] *(740 /1722b) 0093 0x05D      [+][X]
%include 'in_out.asm'
section .data
msg1 db 'Введите B: ',0h
msg2 db "Наибольшее число: ",0h
A dd '81'
C dd '72'
section .bss
max resb 10
B resb 10
section
.text
global _start
_start:
;----- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
;----- Ввод 'B'
mov ecx,B
mov edx,10
call sread
;----- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
;----- Записываем 'A' в переменную 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
;----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
;----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в 'max'
;-----
Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jg fin
```

Рис 17: Ввод текста программы

Создаю исполняемый файл и проверяю его работу:

```
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-3.asm
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-3
Введите B: 22
Наибольшее число: 72
```

Рис 18: Создание и проверка файла

Создаю файл lab7-4.asm:

```
tatyana@vbox: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ touch lab7-4.asm
```

Рис 19: Создание файла

Пишу в него программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений:

```
lab7-4.asm [----] 0 L: [ 7+39 46/ 47] *(563 /
x: RESB 80
a: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax, msg_x
call sprint
mov ecx, x
mov edx, 80
call sread
mov eax, x
call atoi
mov edi, eax

mov eax, msg_a
call sprint
mov ecx, a
mov edx, 80
call sread
mov eax, a
call atoi
mov esi, eax
```

Рис 20: Ввод текста программы

Создаю исполняемый файл и проверяю его работу для значений $x_1=2$, $a_1=3$, $x_2=4$, $a_2=2$:

```
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-4.asm
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-4
Введите x: 2
Введите a: 3
Результат: 5
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ mc
tatyana@vbox:~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-4
Введите x: 4
Введите a: 2
Результат: 2
```

Рис 21: создание и проверка файла

4. ВЫВОДЫ

В ходе выполнения лабораторной работы я изучила команды условного и безусловного переходов, приобрела навыки написания программ с использованием переходов и познакомилась с назначением и структурой файла листинга.

5. Список литературы

Курс: Архитектура компьютеров и операционные системы. Раздел
"Архитектура компьютеров" (02.03.00, УГСН) (rudn.ru)

