Отчёт по лабораторной работе №7

Дисциплина: Архитектура компьютера

Бызова Мария Олеговна

Содержание

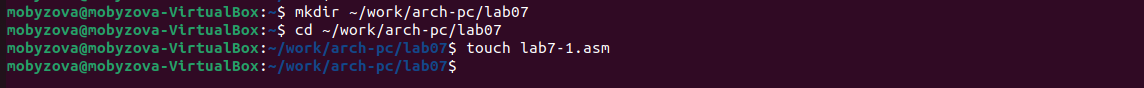
# 1 Цель работы

Целью лабораторной работы является изучение команд условного и безусловного переходов, приобретение навыков написания программ с использованием переходов, знакомство с назначением и структурой файла листинга.

# 2 Выполнение лабораторной работы

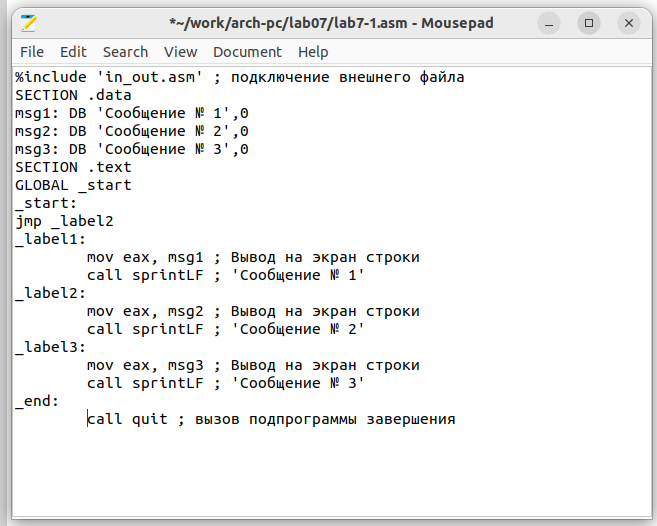
## 2.1 Реализация переходов в NASM

1. С помощью утилиты mkdir создаем директорию, в которой будем создавать файлы с программами для лабораторной работы №7. Переходим в созданный каталог с помощью утилиты cd. С помощью утилиты touch создаем файл lab7-1.asm (рис. [??]).



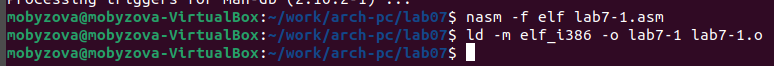
Создание необходимой директории и файла

1. При помощи Midnight Commander открываем созданный файл lab7-1.asm, вставляем в него программу с использованием инструкции jmp из листинга 7.1 (рис. [??]).

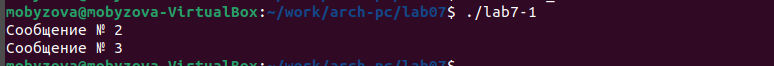


Редактирование файла

Далее создаем исполняемый файл и запускаем его (рис. [??], [??]).



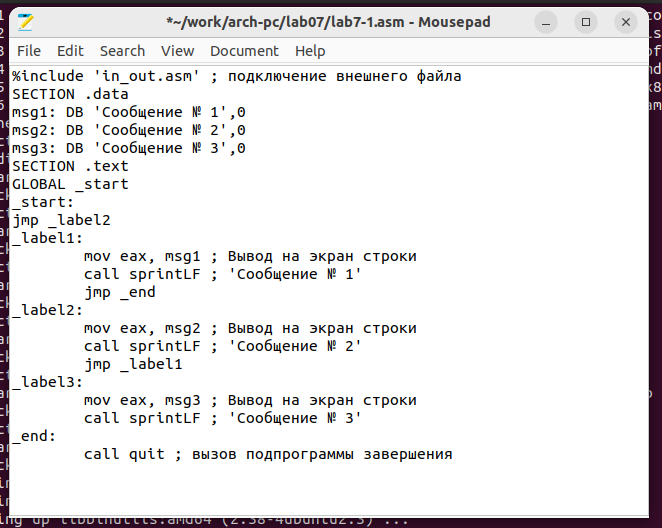
Создание исполняемого файла



Запуск исполняемого файла

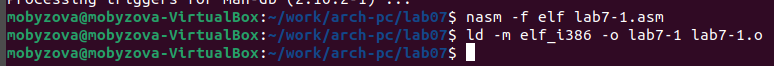
Таким образом, использование инструкции jmp \_label2 меняет порядок исполнения инструкций и позволяет выполнить инструкции начиная с метки \_label2, пропустив вывод первого сообщения.

Изменим текст программы в соответствии с листингом 7.2 таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию jmp с меткой \_label1 (т.е. переход к инструкциям вывода сообщения № 1) и после вывода сообщения № 1 добавим инструкцию jmp с меткой \_end (т.е. переход к инструкции call quit) (рис. [??]).

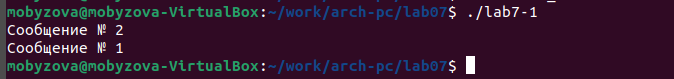


Редактирование файла

Далее создаем исполняемый файл и запускаем его (рис. [??], [??]).

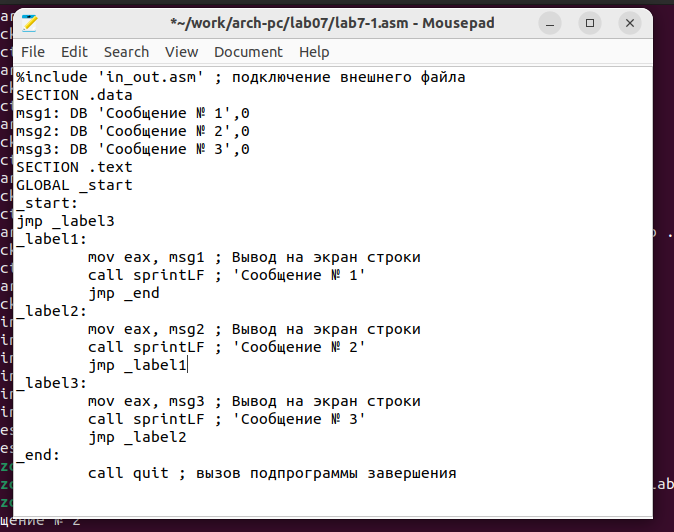


Создание исполняемого файла



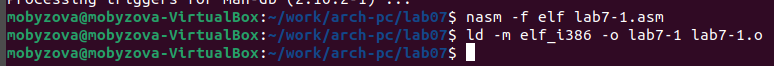
Запуск исполняемого файла

Теперь изменим текст программы, добавив или изменив инструкции jmp, таким образом, чтобы она выводила сначала ‘Сообщение № 3’, потом ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. (рис. [??]).

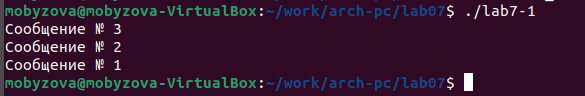


Редактирование файла

Далее создаем исполняемый файл и запускаем его (рис. [??], [??]).



Создание исполняемого файла



Запуск исполняемого файла

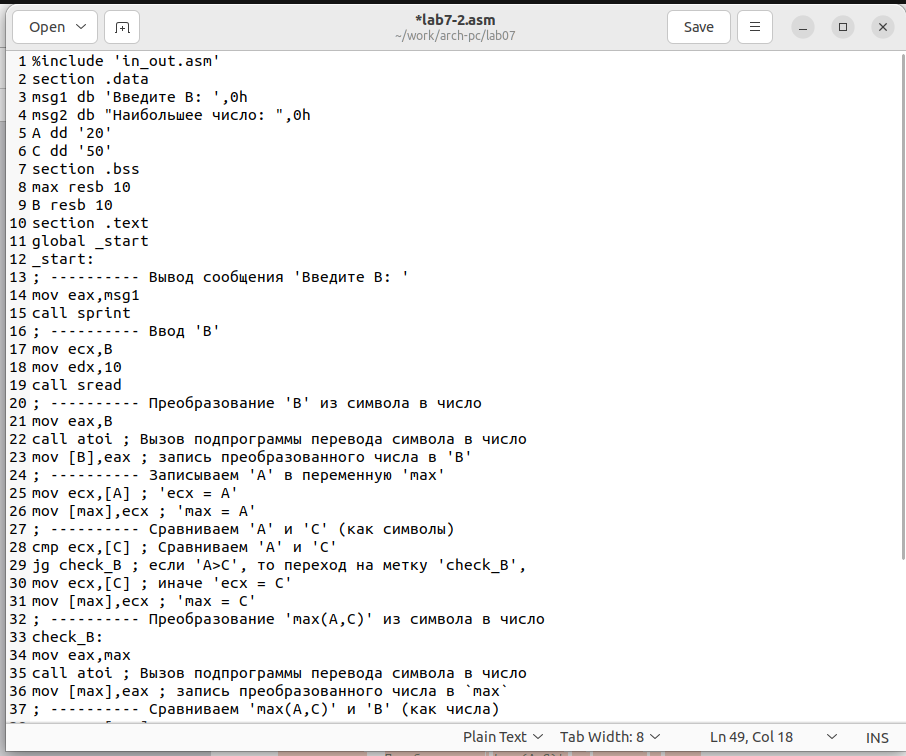
Убеждаемся, что программа отработала верно.

1. Создаем файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 (рис. [??]).

Создание файла

Создание файла

Внимательно изучив текст программы из листинга 7.3, введем его в lab7-2.asm (рис. [??]).

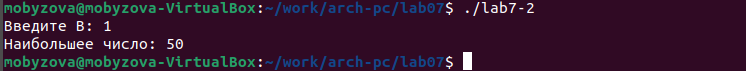


Редактирование файла

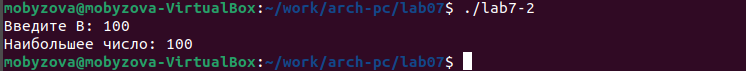
Далее создаем исполняемый файл и запускаем его. Проверим работу программы для разных значений B (рис. [??], [??], [??]).

Создание исполняемого файла

Создание исполняемого файла



Запуск исполняемого файла



Запуск исполняемого файла

Убеждаемся, что программа отработала верно.

## 2.2 Изучение структуры файлы листинга

1. Создаем файл листинга для программы из файла lab7-2.asm, пользуясь следующей утилитой (рис. [??]).

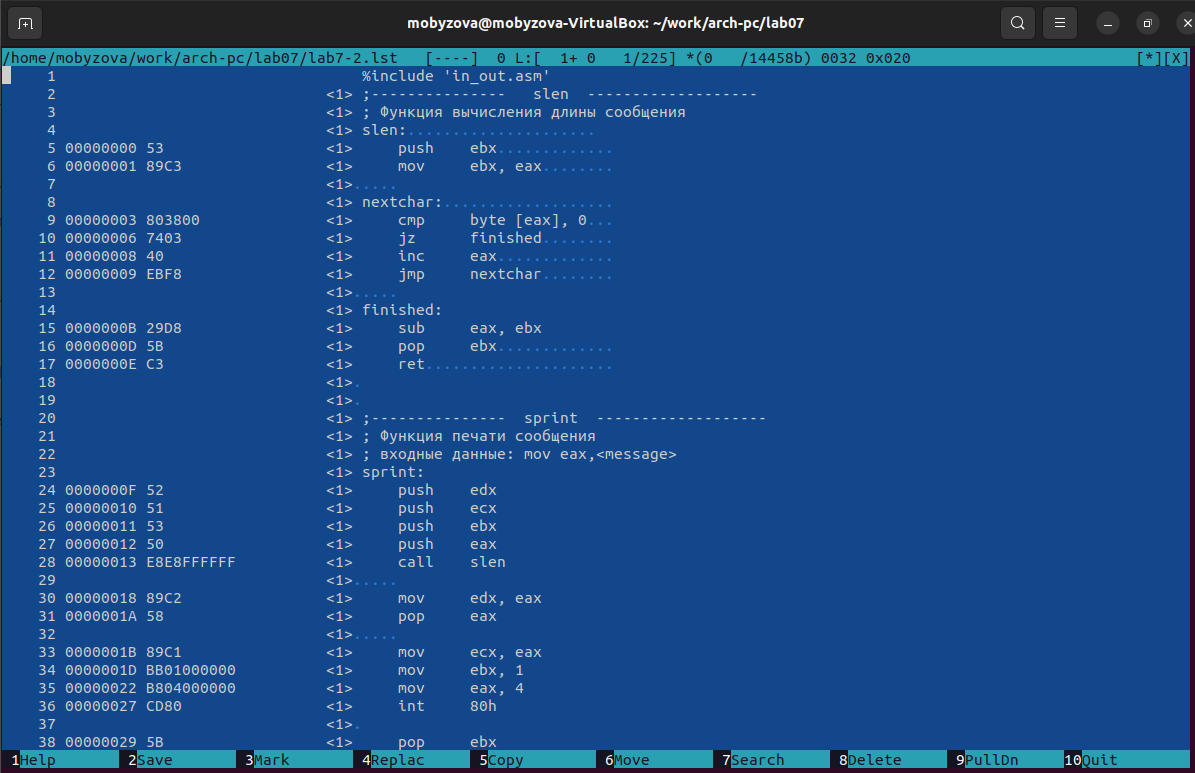
Создание файла листинга

Создание файла листинга

Откроем файл листинга lab7-2.lst с помощью любого текстового редактора, например mcedit (рис. [??], [??]).

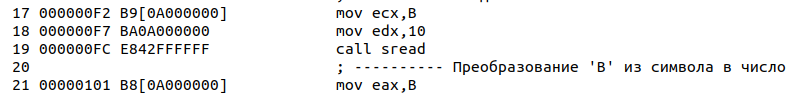
Открытие файла листинга

Открытие файла листинга



Открытый файл листинга

Подробно опишем содержимое строк 17, 19, 21 (рис. [??]).



Выбранные строки

1. 17 (номер строки) 000000F2 (адрес, начинается по смещению 000000F2 в сегменте кода) B9[0A000000] (машинный код) move ecx,B (исходный текст программы, в котором мы помещаем значение, хранящееся в B, в ячейку ecx).
2. 19 (номер строки) 000000FC (адрес, начинается по смещению 000000FC в сегменте кода) E842FFFFFF (машинный код) call sread (исходный текст программы, в котором мы вызываем подпрограмму считывания введённого значения).
3. 21 (номер строки) 00000101 (адрес, начинается по смещению 00000101 в сегменте кода) B8[0A000000] (машинный код) mov eax,B (исходный текст программы, в котором мы помещаем значение, хранящееся в B, в ячейку eax).

Затем зайдём в файл программы lab7-2.asm для удаления одного операнда. Удалим операнд в строчке: mov ecx, [A] (рис. [??]).

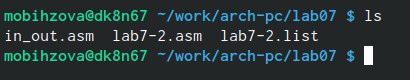
Изменение файла

Изменение файла

После сохранения изменений в программе проассемблируем её с получением файла листинга. В ходе проассемблирования система выдаёт ошибку и создаёт только файл lab7-2.list (рис. [??], [??]).

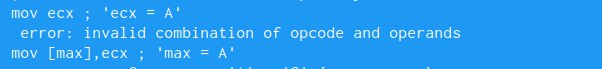
Создание файла листинга

Создание файла листинга



Созданные файлы

Зайдем в листинг для изучения того, что добавилось в него, после возникновения ошибки (рис. [??]). Мы видим, что в него дополнительно добавляется строка “\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* error: invalid combination of opcode and operands”, предупрждающая об ошибке.



Открытый файл листинга

# 3 Выполнение заданий для самостоятельной работы

1. Задание: Напишите программу нахождения наименьшей из 3 целочисленных переменных a, b, 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.

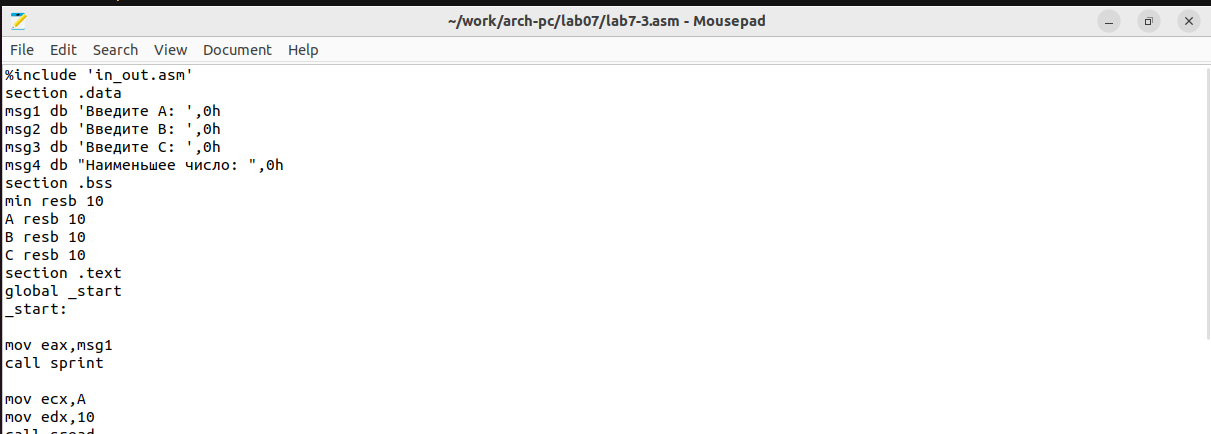
Для написания программы возьмём за основу программу “Листинг 3”, в которую внесём изменения для правильного выполнения задания. В ходе написания программы будем использовать команду: JL (Переход если a < b).

Создаем файл lab7-3.asm с помощью утилиты touch (рис. [??]).

Создание файла

Создание файла

Открываем созданный файл для редактирования, вводим в него текст программы (рис. [??]).

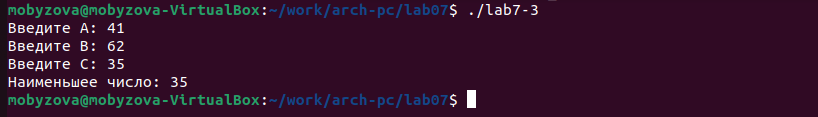


Написание программы

Далее создаем исполняемый файл и запускаем его (рис. [??], [??]).

Создание исполняемого файла

Создание исполняемого файла



Запуск исполняемого файла

Введя числа из моего варианта №10, мы можем проверить правильность работы программы.

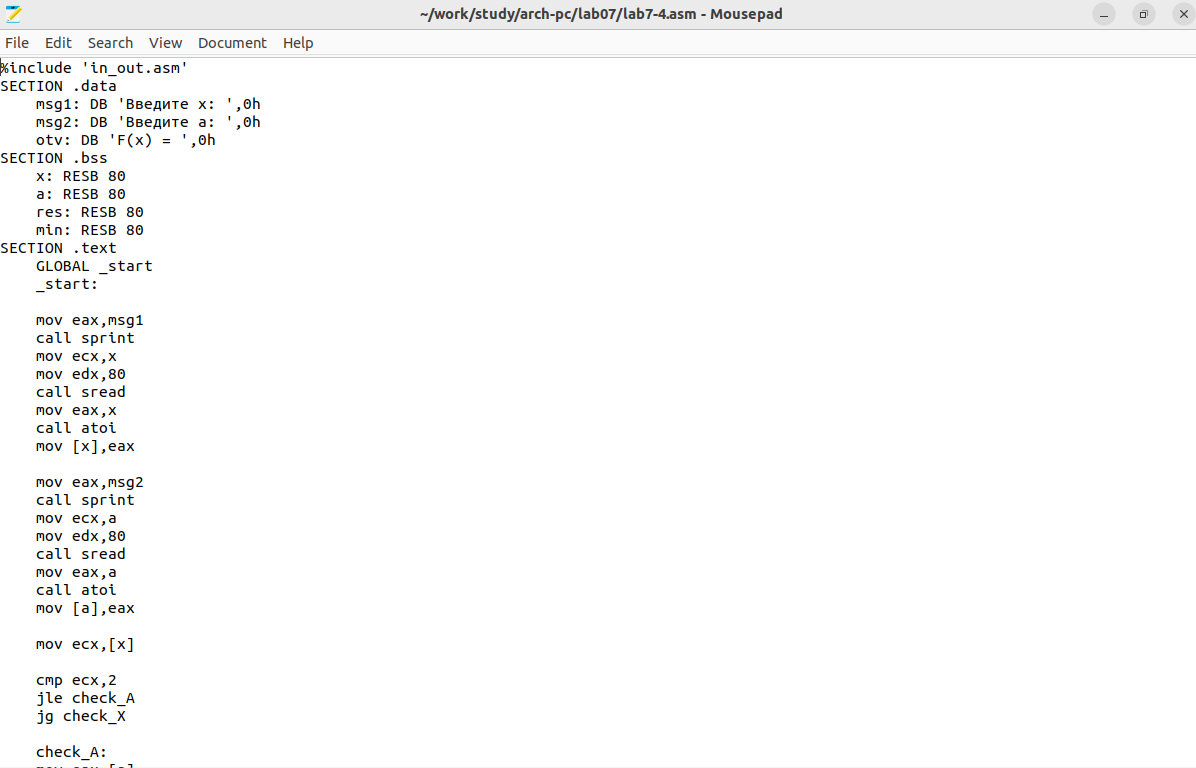
1. Задание: Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции f(a) и выводит результат вычислений. Вид функции f(x) выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений x и a из 7.6.

Создаем файл lab7-3.asm с помощью утилиты touch (рис. [??]).

Создание файла

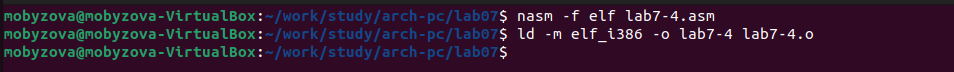
Создание файла

Открываем созданный файл для редактирования, вводим в него текст программы вычисления функции: x-2 если x>2 и 3\*a если x<=2 (выражение указано в соответствии с вариантом №10, полученным при выполнении лабораторной работы № 6).(рис. [??]).

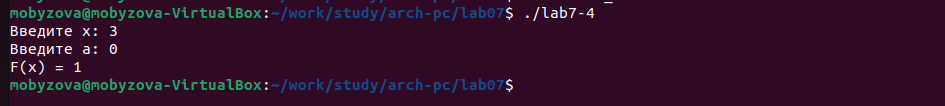


Написание программы

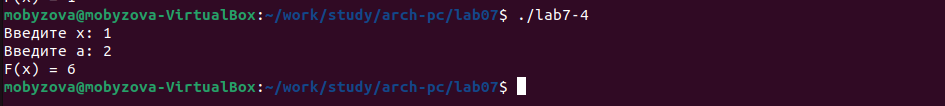
Далее создаем исполняемый файл и запускаем его (рис. [??], [??], [??]).



Создание исполняемого файла



Запуск исполняемого файла



Запуск исполняемого файла

Убеждаемся, что программа отработала верно.

**Листинг №1. Изменённая программа lab7-1.asm**

%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 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'  
 jmp \_label2  
\_end:  
 call quit ; вызов подпрограммы завершения

**Листинг №2. Задание для самостоятельной работы №1**

section .data  
msg1 db 'Введите A: ',0h  
msg2 db 'Введите B: ',0h  
msg3 db 'Введите C: ',0h  
msg4 db "Наименьшее число: ",0h  
section .bss  
min resb 10  
A resb 10  
B resb 10  
C resb 10  
section .text  
global \_start  
\_start:  
  
mov eax,msg1  
call sprint  
  
mov ecx,A  
mov edx,10  
call sread  
  
mov eax,A  
call atoi   
mov [A],eax  
  
mov eax,msg2  
call sprint  
  
mov ecx,B  
mov edx,10  
call sread  
  
mov eax,B  
call atoi  
mov [B],eax  
  
mov eax,msg3  
call sprint  
  
mov ecx,C  
mov edx,10  
call sread  
  
mov eax,C  
call atoi  
mov [C],eax  
  
  
mov ecx,[A]  
mov [min],ecx  
  
cmp ecx,[C]  
jl check\_B  
mov ecx,[C]   
mov [min],ecx  
  
check\_B:  
 mov ecx,[min]  
 cmp ecx,[B]  
 jl fin  
 mov ecx,[B]  
 mov [min],ecx  
  
fin:  
 mov eax, msg4  
 call sprint  
 mov eax,[min]  
 call iprintLF  
 call quit

**Листинг №3. Задание для самостоятельной работы №2**

%include 'in\_out.asm'  
SECTION .data  
 msg1: DB 'Введите x: ',0h  
 msg2: DB 'Введите a: ',0h  
 otv: DB 'F(x) = ',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  
 call sread  
 mov eax,a  
 call atoi  
 mov [a],eax  
  
 mov ecx,[x]  
  
 cmp ecx,2  
 jle check\_A  
 jg check\_X  
  
 check\_A:  
 mov eax,[a]  
 mov ebx, 3  
 mul ebx  
 mov [res],eax  
 jmp fin  
  
 check\_X:  
 add ecx, -2  
 mov [res],ecx  
 jmp fin  
  
 fin:  
 mov eax,otv  
 call sprint  
 mov eax,[res]  
 call iprintLF  
 call quit

# 4 Выводы

В ходе выполнения лабораторной работы мы познакомились с назначением и структурой файла листинга, изучили команды условного и безусловного переходов, а также приобрели навыки написания программ с использованием переходов.