Команды безусловного и условного переходов в NASM. Программирование ветвлений

Лабораторная работа №7

Владимир Романович Козомазов

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

* Изучение команд условного и безусловного переходов в языке ассемблера NASM и приобретение навыков написания программ с использованием ветвлений.
* Знакомство с назначением и структурой файла листинга.

# 2 Выполнение лабораторной работы

## 2.1 Описание выполняемого задания

* Написать программу, реализующую безусловный переход jmp в NASM и изменить ее в соответствии с заданием.
* Написать программу, реализующую условный переход jg в NASM и вычисляющую максимум из трех чисел, два из которых заданы, а третье вводится с клавиатуры. Получить файл листинга и исследовать его стуктуру.

## 2.2 Выполнение заданий лабораторной работы

### 2.2.1 Реализация переходов в NASM

Создал каталог для программы лабораторной работы №7 и перешёл в него. (рис. 1)

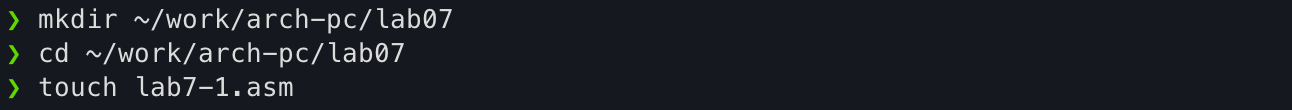


Рис. 1: создание каталога для лабораторной работы и создание файла lab7-1.asm

Ввёл код в файл lab7-1.asm из листинга 7.1 (рис. 2)



Рис. 2: Ввод кода в файл lab7-1.asm из листинга

Cкомпилировал и запустил программу lab7-1 (рис. 3)

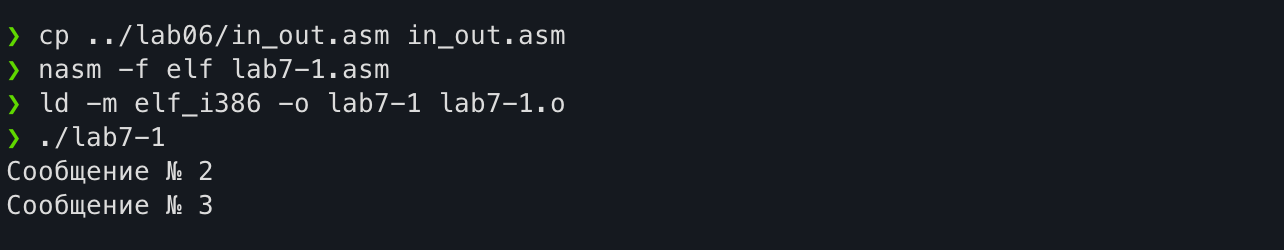


Рис. 3: Компиляция и запуск программы lab7-1

Изменил код порграммы lab7-1 из листинга 7.2 (рис. 4)



Рис. 4: Изменение кода программы lab7-1

Скомпилировал и запустил программу laab7-1 с кодом из листинга 7.2 (рис. 5)

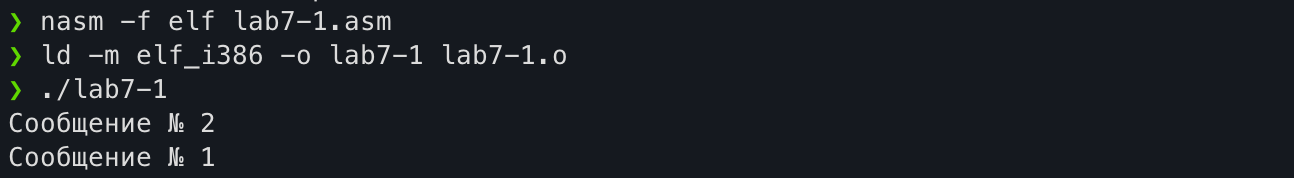


Рис. 5: Компиляция и запуск программы lab7-1

Изменил код программы lab7-1, чтобы вывод программы соответствовал заданию (рис. 6)

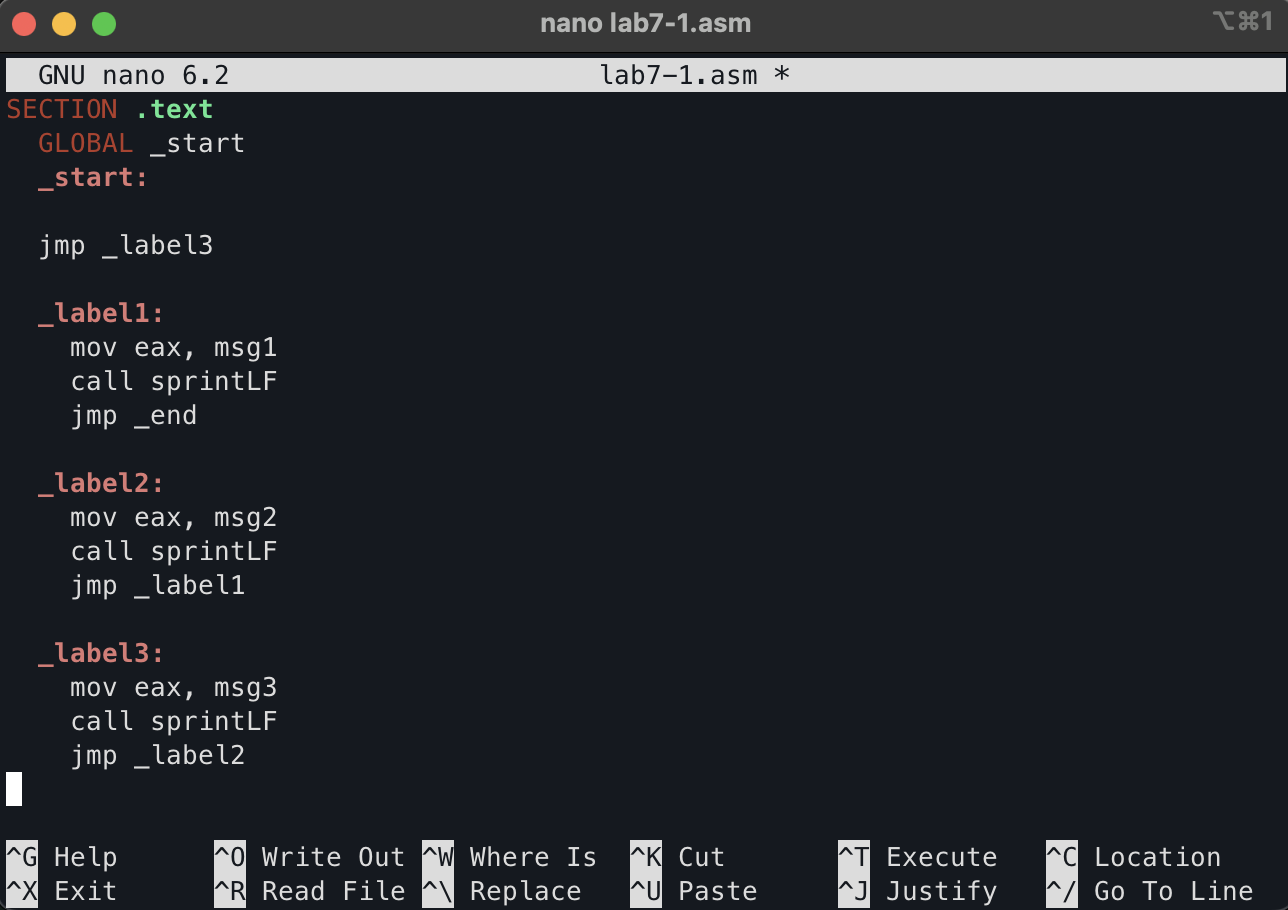


Рис. 6: Изменение кода программы lab7-1 в соответствии с заданием

Снова скомпилировал и запустил программу lab7-1 и получил результаты, соответсвующие заданию (рис. 7)

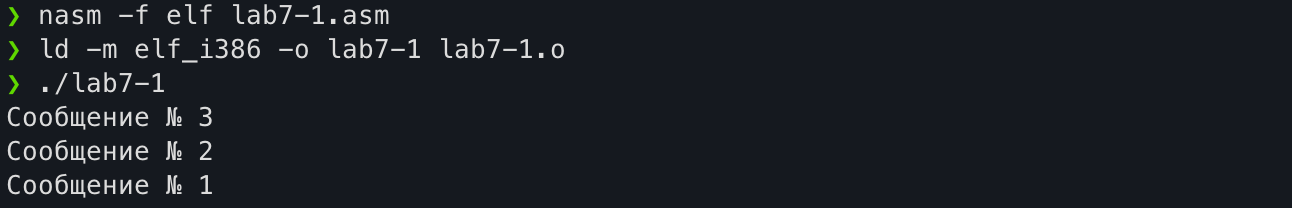


Рис. 7: Запуск программы lab7-1 и получение результатов, соответствующих заданию

Создал файл lab7-2.asm и записал код из листинга 7.3 (рис. 8)



Рис. 8: Создание файлаlab7-2.asm и запись в него кода

Скомпилировал и запустил программу lab7-2 с кодом из листинга 7.3 (рис. 9)

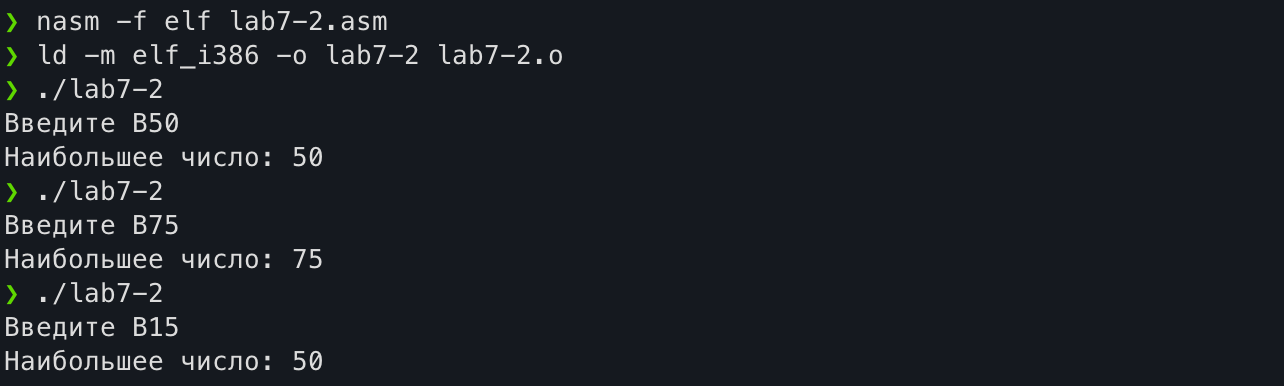


Рис. 9: Компиляция и запуск программы lab7-2, проверка работы программы для разных значчений В

### 2.2.2 Изучение структуры файла листинга

Создал файл листинга lab7-2.lst и открыл его при помощи текстового редактора mcedit (рис. 10)

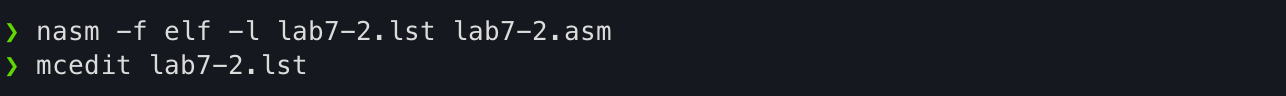


Рис. 10: Создание файла листинга lab7-2.lst и его открытие

### Объяснение трех строк файла листинга

Cтрока листинга 1:

%include 'in\_out.asm'

Эта строка является макрокомандой включения файла, она включает весь код из файла in\_out.asm в файл lab7-2.asm, что мы и видим в файле листинга (рис. 11)

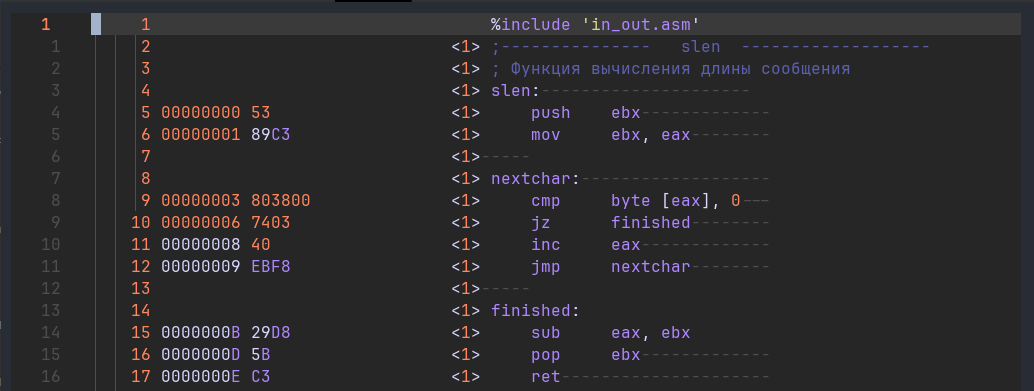


Рис. 11: Объяснение строки 1 в файле листинга lab7-2.lst

Строка листинга 194:

mov ecx, B

Данная строка соответствует строке 20 исходного файла (номер строки нааходится в столбце 1). Далее в столбце 2 находится адрес смещения (offset). В столбце 3 находится машинный код выполняемой инструкции и данные, а в столбце 4 находится собственно сам ассемблерный код (рис. 12)

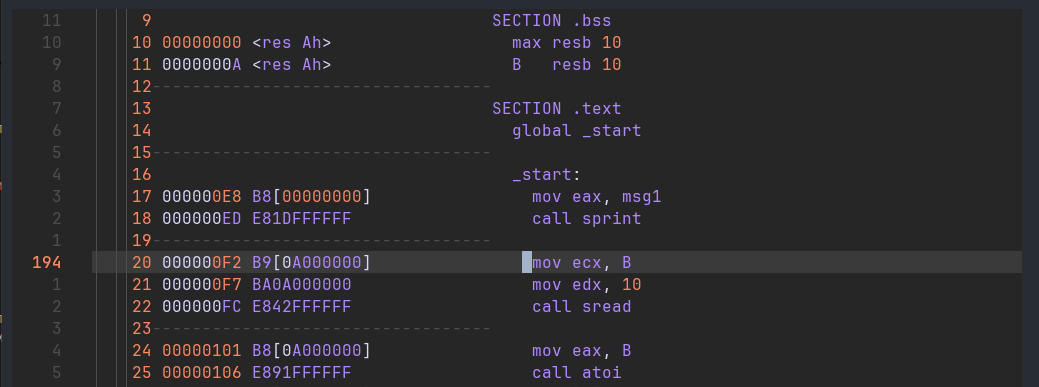


Рис. 12: Объяснение строки 194 в файле листинга lab7-2.lst

Строка листинга 206:

jg check\_B

В первом столбце этой строки находится номер строки исходного файла, во втором столбце находится адрес смещения, в третьем столбце находится машинный код выполняемой инструкции, а в четвертом столбце собственно сам ассемблерный код (рис. 13)

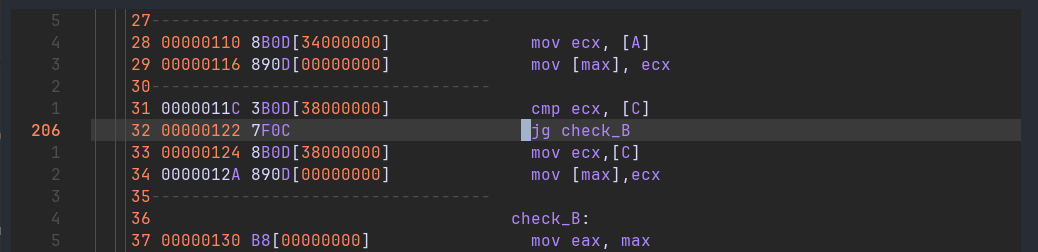


Рис. 13: Объяснение строки 206 в файле листинга lab7-2.lst

### 2.2.3 Трансляия кода программы и изучение листинга с ошибкой

Изменил инструкцию cmp, удалив один операнд в файле lab7-2.asm (рис. 14)

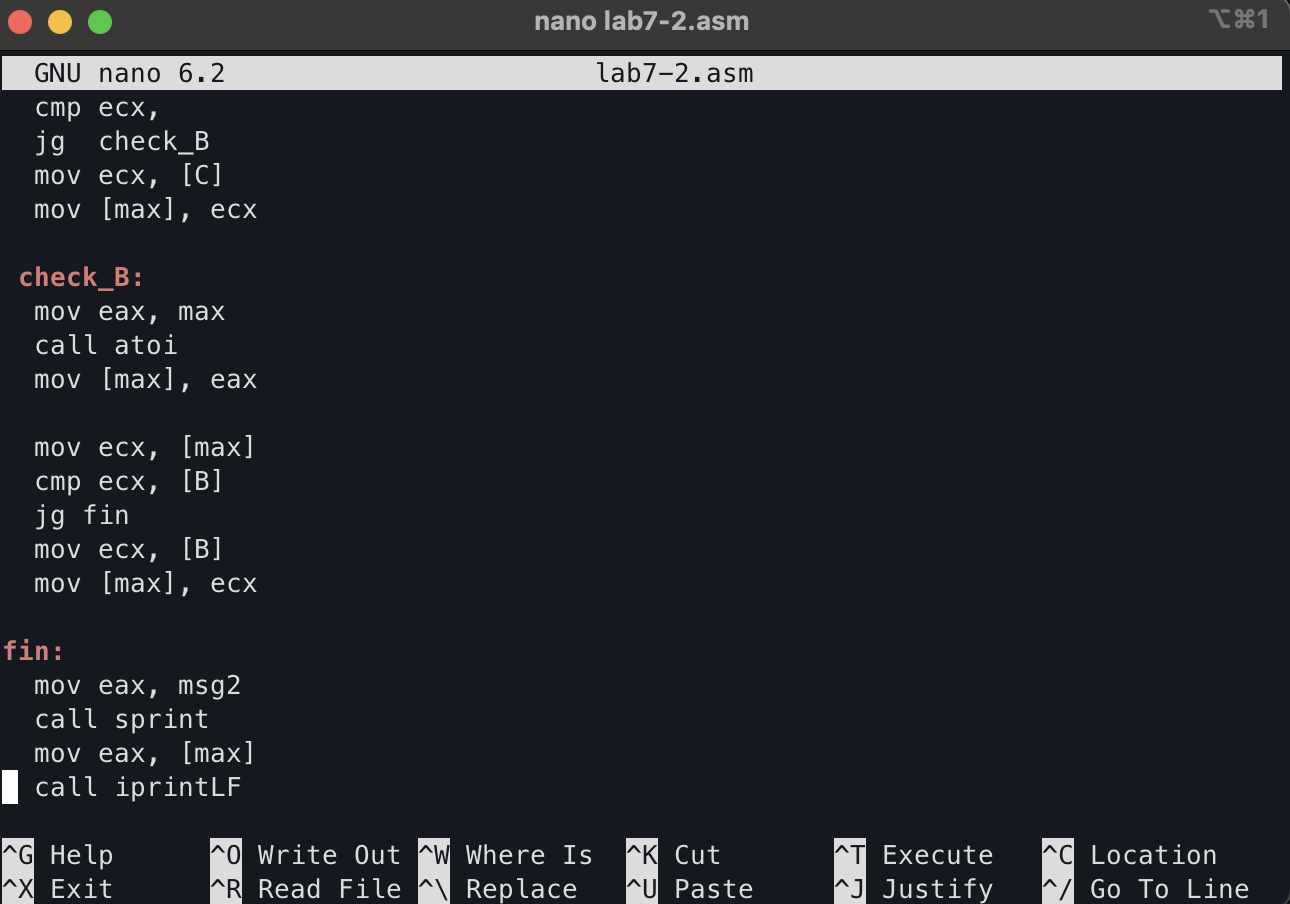


Рис. 14: Удаление операнда в инструкции cmp в файле lab7-2.asm

Выполнил трансляцию с получением файла листига (рис. 15)

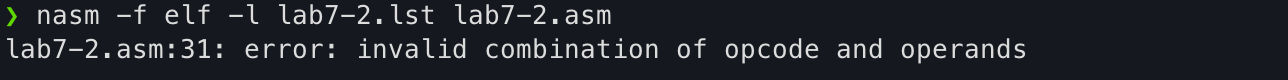


Рис. 15: Создание листинга для файла lab7-2.asm с ошибкой

Проверил листинг файла lab7-2.asm, созданный с ошибкой и заметил изменения данных в листинге (рис. 16). В листинге добавилась строка с данными об ошибке.

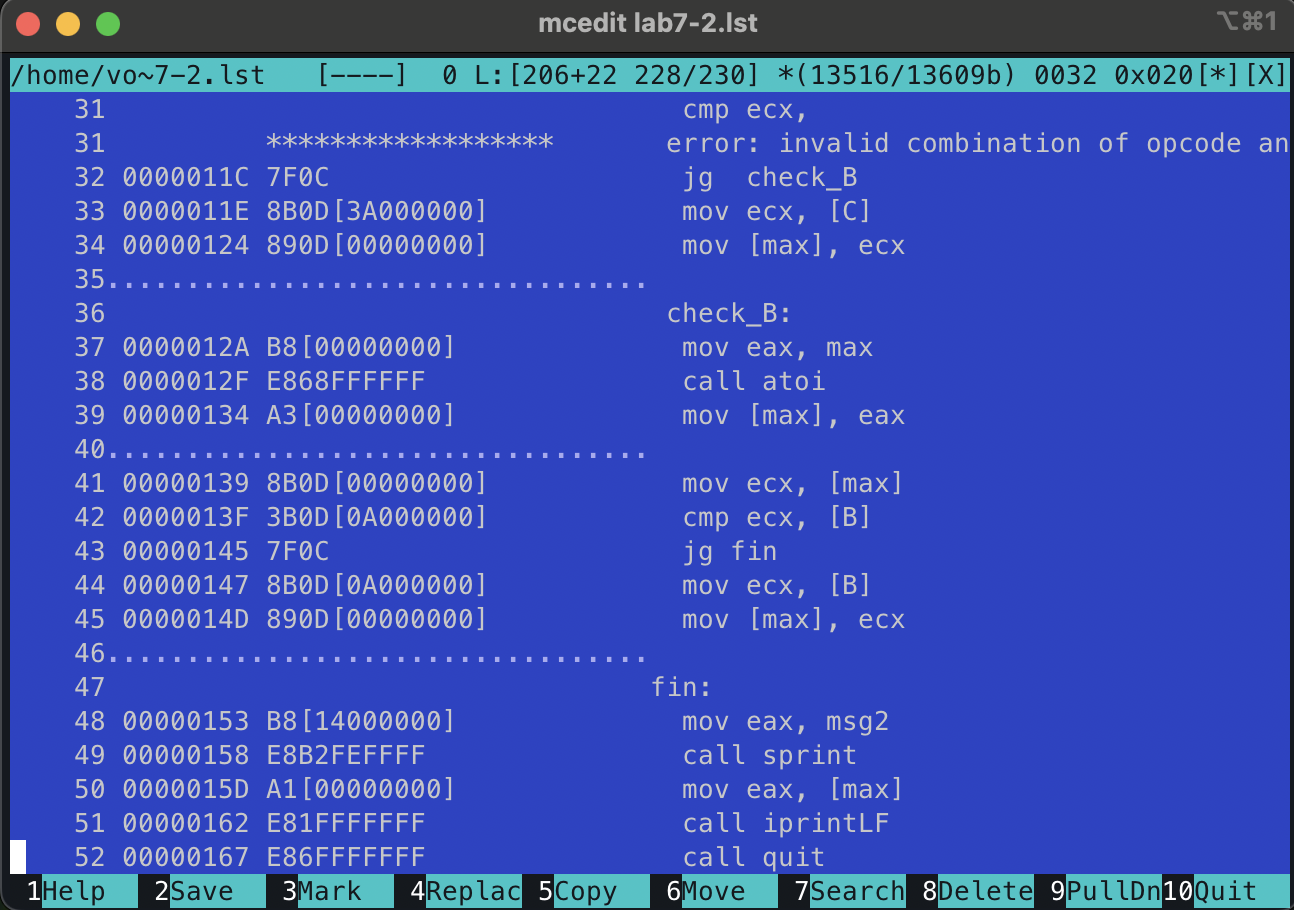


Рис. 16: Открытие листинга файла lab7-2.asm, созданный с ошибкой

Вывод: когда в ассемблерном коде допущена ошибка, то при трансляции объектный файл не создается, а выводится сообщение об ошибке. Файл листинга создается и в него выводится строка с данными об ошибке.

## 2.3 Выводы по результатам выполнения заданий лабораторной работы

При выполнении заданий лабораторной работы были написаны программы, с помощью которых были изучены инструкции безусловного и условного переходов. Также были получен файл листиинга и изучена его структура.

# 3 Выполнение самостоятельного задания

## 3.1 Описание выполняемого самостоятельного задания

Задание для самостоятельной работы состоит из двух частей: \* Написать программу нахождения наименьшего из трех целых чисел. \* Написать программу вычисления функции

в зависимости от вводимого с клавиатуры значения и параметра .

## 3.2 Выполнение заданий для самостоятельной работы

### 3.2.1 Выполнение задания 1

Для программы нахождения наименьшего из трех чисел создал файл lab7\_min\_of\_three.asm и ввел в него код (рис. 17)

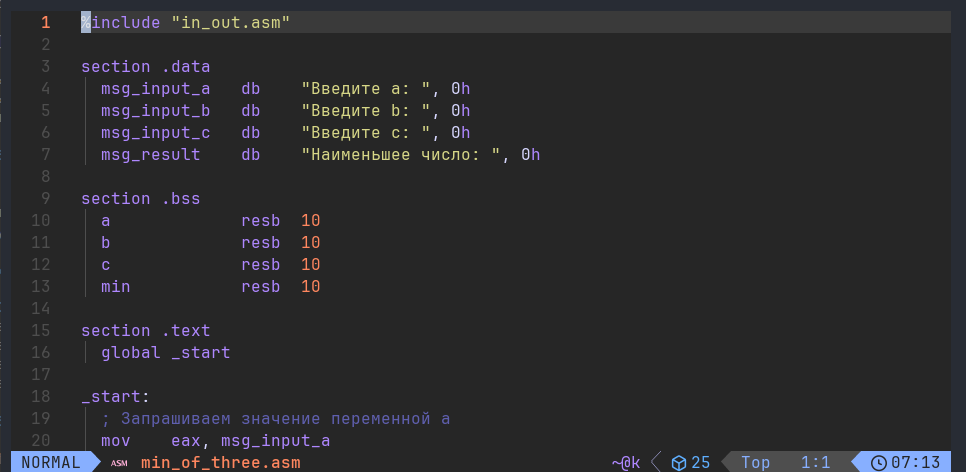


Рис. 17: Написание программы lab7\_min\_of\_three.asm

Проверил работу программы lab7\_min\_of\_three на данных из задания (рис. 18)

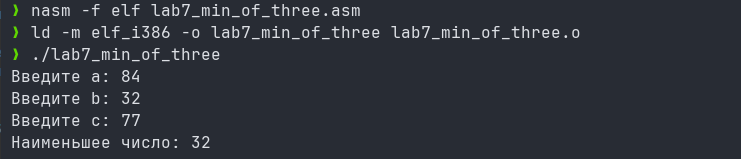


Рис. 18: Проверка работы программы lab7\_min\_of\_three.asm

### 3.2.2 Выполнение задания 2

Для программы вычисления функции создал файл lab7\_calc\_function.asm и ввел в него соответствующий код (рис. 19)

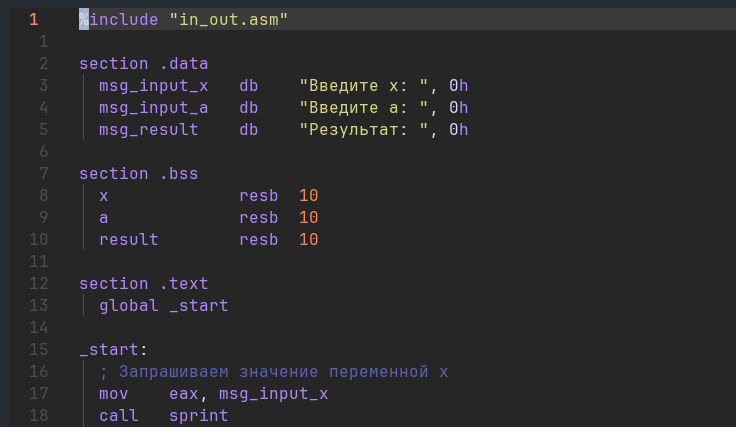


Рис. 19: Написание программы lab7\_calc\_function.asm

Проверка работы программы lab7\_calc\_function на данных из задания (рис. 20)

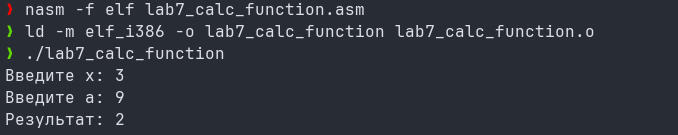


Рис. 20: Проверка работы программы lab7\_calc\_function.asm

## 3.3 Выводы по результатам выполнения самостоятельного задания

В ходе выполнения самостоятельного задания были написаны программы, которые вычисляют наименьшее из трех чисел и вычисляют функцию в зависимости от вводимого с клавиатуры значения и параметра . Были использованы методы сравнения и ветвлений, освоенные в ходе выполнения лабораторной работы.

# 4 Выводы

* В ходе выполнения лаборатороной работы были изучены команды безусловного и условного переходов в языке ассемблера NASM, приобретены навыки написания программ с использованием ветвлений, а также написаны несколько программ, использующих ветвления.
* При трансляции программы был получен файл листинга, изучено содержание файла листинга и его структура.

# 5 Листинги написанных программ

## 5.1 Программа lab7-1.asm

%include 'in\_out.asm'  
  
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  
 call sprintLF  
 jmp \_end  
  
 \_label2:  
 mov eax, msg2  
 call sprintLF  
 jmp \_label1  
  
 \_label3:  
 mov eax, msg3  
 call sprintLF  
 jmp \_label2  
  
 \_end:  
 call quit

## 5.2 Программа lab7-2.asm

%include 'in\_out.asm'  
  
SECTION .data  
 msg1 DB "Введите В: ",0h  
 msg2 DB "Наибольшее число: ",0h  
 A dd "20"  
 C dd "50"  
  
SECTION .bss  
 max resb 10  
 B resb 10  
  
SECTION .text  
 global \_start  
  
 \_start:  
 mov eax, msg1  
 call sprint  
  
 mov ecx, B  
 mov edx, 10  
 call sread  
  
 mov eax, B  
 call atoi  
 mov [B], eax  
  
 mov ecx, [A]  
 mov [max], ecx  
  
 cmp ecx,  
 jg check\_B  
 mov ecx,[C]  
 mov [max],ecx  
  
 check\_B:  
 mov eax, max  
 call atoi  
 mov [max], eax  
  
 mov ecx, [max]  
 cmp ecx, [B]  
 jg fin  
 mov ecx, [B]  
 mov [max], ecx  
  
 fin:  
 mov eax, msg2  
 call sprint  
 mov eax, [max]  
 call iprintLF  
 call quit

## 5.3 Программа lab7\_min\_of\_three.asm

%include "in\_out.asm"  
  
section .data  
 msg\_input\_a db "Введите a: ", 0h  
 msg\_input\_b db "Введите b: ", 0h  
 msg\_input\_c db "Введите c: ", 0h  
 msg\_result db "Наименьшее число: ", 0h  
  
section .bss  
 a resb 10  
 b resb 10  
 c resb 10  
 min resb 10  
  
section .text  
 global \_start  
  
\_start:  
 ; Запрашиваем значение переменной a  
 mov eax, msg\_input\_a  
 call sprint  
 ; Организуме ввод с клавиатуры значения переменной a  
 mov ecx, a  
 mov edx, 10  
 call sread  
 ; Преобразуем строку, полученную с клавиатуры, в число  
 mov eax, a  
 call atoi  
 mov [a], eax  
  
 ; Запрашиваем значение переменной b  
 mov eax, msg\_input\_b  
 call sprint  
 ; Организуме ввод с клавиатуры значения переменной b  
 mov ecx, b  
 mov edx, 10  
 call sread  
 ; Преобразуем строку, полученную с клавиатуры, в число  
 mov eax, b  
 call atoi  
 mov [b], eax  
  
 ; Запрашиваем значение переменной c  
 mov eax, msg\_input\_c  
 call sprint  
 ; Организуме ввод с клавиатуры значения переменной a  
 mov ecx, c  
 mov edx, 10  
 call sread  
 ; Преобразуем строку, полученную с клавиатуры, в число  
 mov eax, c  
 call atoi  
 mov [c], eax  
  
 ; min = a  
 mov ecx, [a]  
 mov [min], ecx  
  
 ; Сравниваем a и b  
 cmp ecx, [b]  
 ; Если a > b, то min = a и переходим  
 ; к сравнению min и c  
 jl check\_c  
 ; Иначе min = b  
 mov ecx, [b]  
 mov [min], ecx  
  
 ; Сравниваем min(a,b) и c  
check\_c:  
 mov ecx, [min]  
 cmp ecx, [c]  
 ; Если min < c, то уже нашли min  
 jl fin  
 ; Иначе min = c  
 mov ecx, [c]  
 mov [min], ecx  
  
fin:  
 ; Выводим сообщение пользователю  
 mov eax, msg\_result  
 call sprint  
 ; Выводим результат  
 mov eax, [min]  
 call iprintLF  
  
 call quit

## 5.4 Программа lab7\_calc\_function.asm

%include "in\_out.asm"  
  
section .data  
 msg\_input\_x db "Введите x: ", 0h  
 msg\_input\_a db "Введите a: ", 0h  
 msg\_result db "Результат: ", 0h  
  
section .bss  
 x resb 10  
 a resb 10  
 result resb 10  
  
section .text  
 global \_start  
  
\_start:  
 ; Запрашиваем значение переменной x  
 mov eax, msg\_input\_x  
 call sprint  
 ; Организуме ввод с клавиатуры значения переменной a  
 mov ecx, x  
 mov edx, 10  
 call sread  
 ; Преобразуем строку, полученную с клавиатуры, в число  
 mov eax, x  
 call atoi  
 mov [x], eax  
  
 ; Запрашиваем значение переменной a  
 mov eax, msg\_input\_a  
 call sprint  
 ; Организуме ввод с клавиатуры значения переменной b  
 mov ecx, a  
 mov edx, 10  
 call sread  
 ; Преобразуем строку, полученную с клавиатуры, в число  
 mov eax, a  
 call atoi  
 mov [a], eax  
  
 ; Сравниваем а и 7  
 mov ecx, [a]  
 cmp ecx, 7  
 jl a\_lower\_7  
 ; Если a >= 7, то result = a - 7  
 mov eax, [a]  
 sub eax, 7  
 mov [result], eax  
 jmp fin  
  
a\_lower\_7:  
 ; Если a < 7, то result = a \* x  
 mov eax, [a]  
 mov ecx, [x]  
 mul ecx  
 mov [result], eax  
  
fin:  
 ; Выводим сообщение пользователю  
 mov eax, msg\_result  
 call sprint  
 ; Печатаем результат  
 mov eax, [result]  
 call iprintLF  
  
 call quit