**РОССИЙСКИЙ** **УНИВЕРСИТЕТ** **ДРУЖБЫ**

**НАРОДОВ**

**Факультет** **физико-математических** **и** **естественных**

**наук**

**направление:** **Компьютерные** **и** **информационные**

**науки**

**Лабораторная** **работа** **№7**

**дисциплина:** **Архитектура** **компьютеров** **и** **операционные** **системы**

**студент:** **Гробман** **Александр** **Евгеньевич**

**Группа:** **НКАбд-02-23**

**Цель** **работы**

Изучение команд условного и безусловного переходов. Приобретение навыков

написания программ с использованием переходов. Знакомство с назначением и

структурой файла листинга.

**Задание**

1. Реализация переходов в NASM

2. Изучение структуры файлы листинга.

3. Задания для самостоятельной работы

**Теория**

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.

Безусловный переход выполняется инструкцией jmp. Инструкция cmp является

одной из инструкций, которая позволяет сравнить операнды и выставляет флаги

в зависимости от результата сравнения. Инструкция cmp является командой

сравнения двух операндов и имеет такой же формат, как и команда вычитания.

Листинг (в рамках понятийного аппарата NASM) — это один из выходных

файлов, создаваемых транслятором. Он имееттекстовый вид и нужен при отладке

программы, так как кроме строк самой программы он содержит дополнительную

информацию.

**Выполнение** **лабораторной** **работы**

1. **Символьные** **и** **численные** **данные** **в** **NASM**

создаю файл lab7-1.asm и ввожу туда код из листинга

запускаем исполняемый файл

Таким образом, использование инструкции jmp \_label2 меняет порядок исполнения

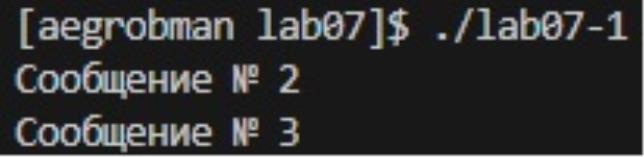
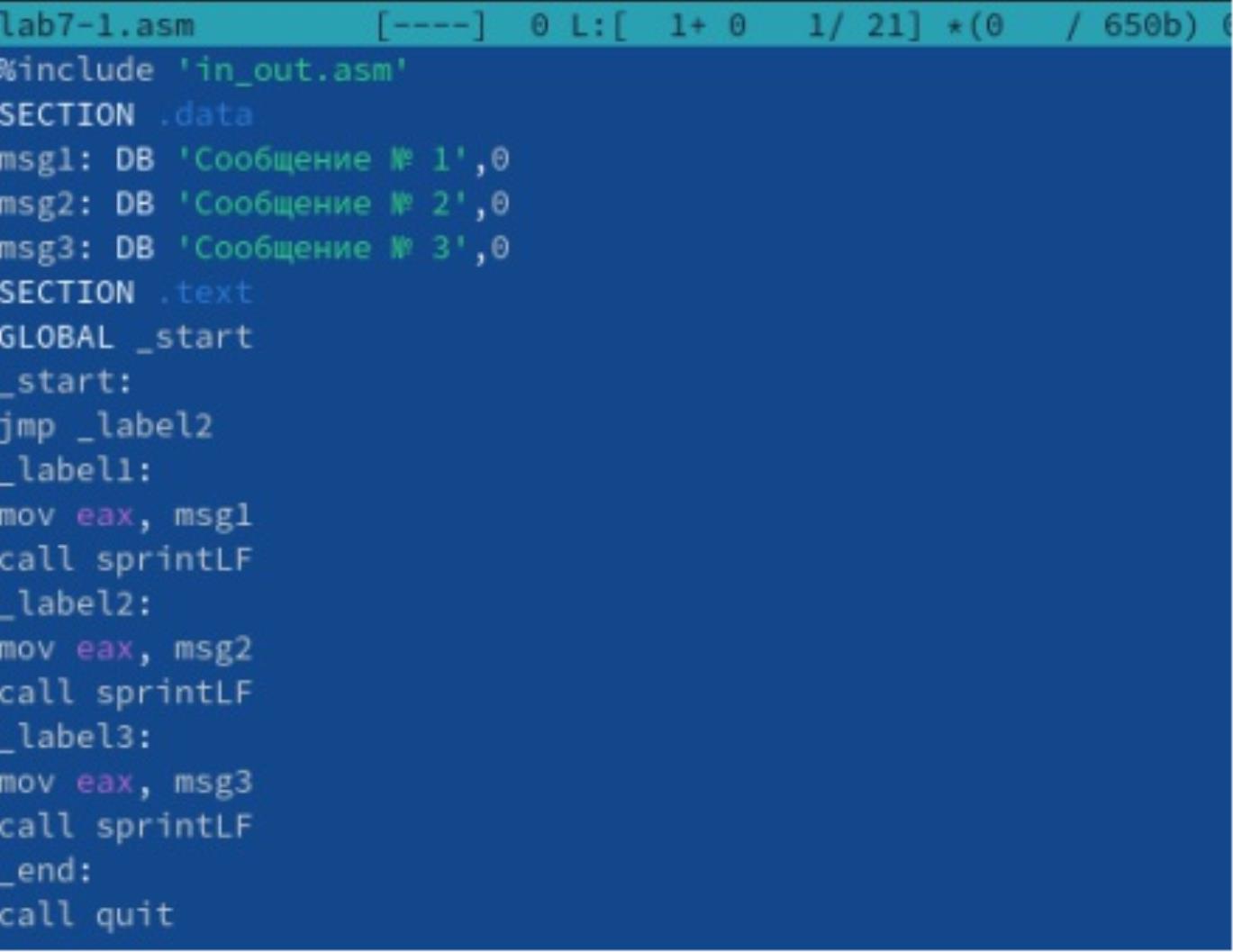
инструкций и позволяет выполнить инструкции начиная с метки \_label2,

пропустив вывод первого сообщения.

Изменю программу таким образом, чтобы она выводила сначала ‘Сообщение

№ 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого изменяю текст

программы в соответствии с листингом 7.2.

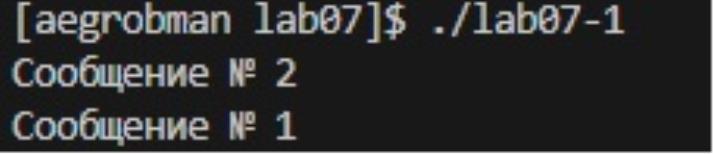
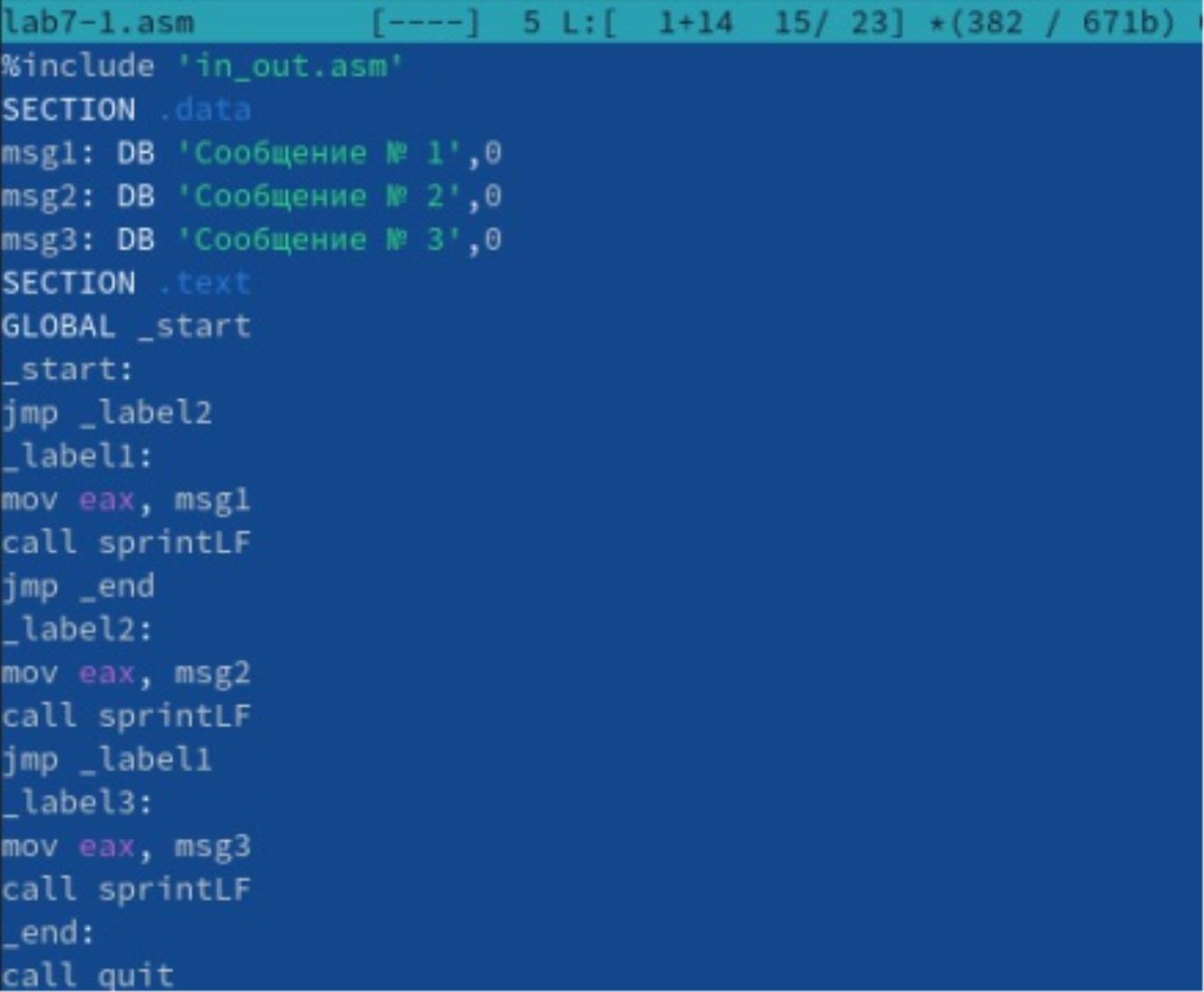


Запускаем исполняемый файл

Затем изменяю текст программы, добавив в начале программы jmp \_label3,

jmp \_label2 в конце метки jmp \_label3, jmp \_label1 добавляю в конце метки jmp

\_label2, и добавляю jmp \_end в конце метки jmp \_label1



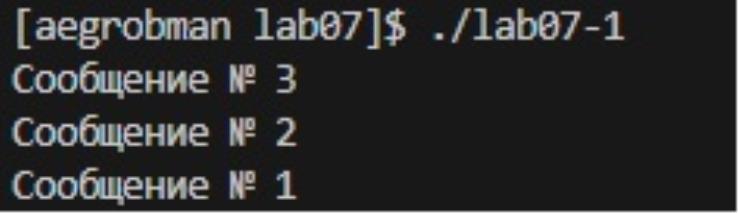
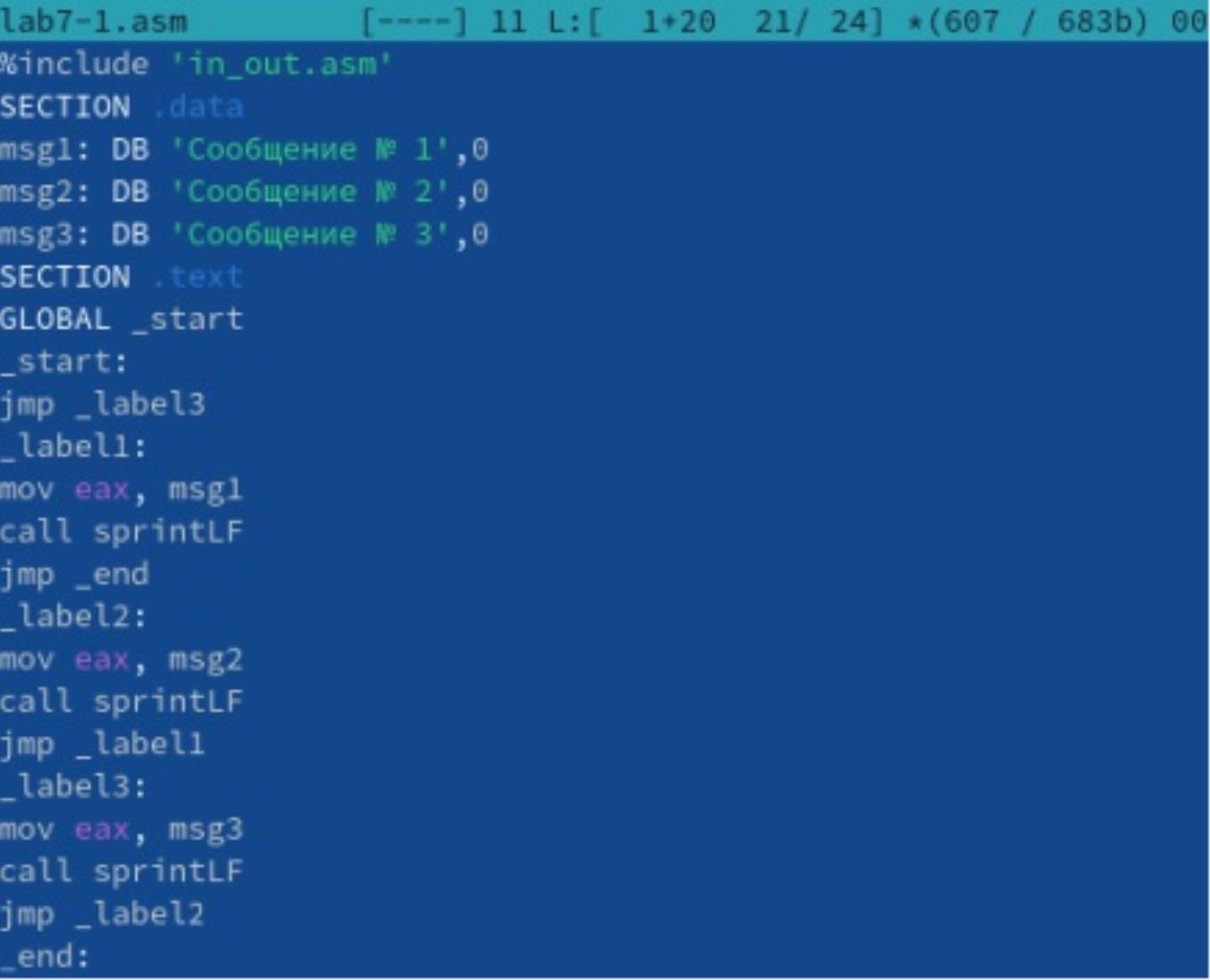
Запускаем исполняемый файл

Рассмотрим программу, которая определяет и выводит на экран наибольшую

из 3 целочисленных переменных: A,B и C. Значения для A и C задаются в программе,

значение B вводиться с клавиатуры.

Создаю файл lab7-2.asm в каталоге ~/work/arch-pc/lab07.



Создаю исполняемый файл и проверьте его работу.

2. **Изучение** **структуры** **файлы** **листинга**

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

Открываю файл листинга lab7-2.lst с помощью текстового редактора и внимательно

изучаю его формат и содержимое.

В представленных трех строчках содержаться следующие данные:

“2” - номер строки кода, “; Функция вычисления длинны сообщения” - комментарий к

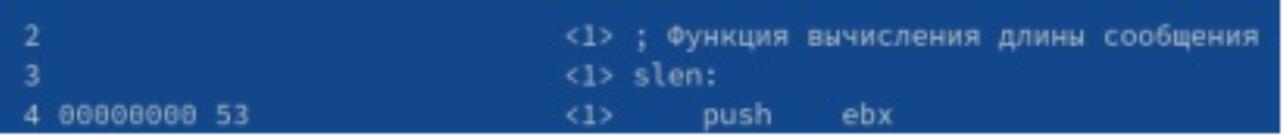
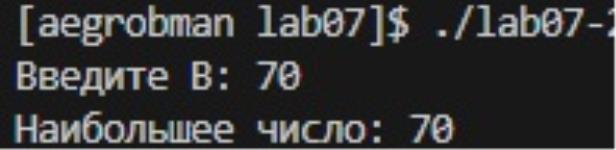
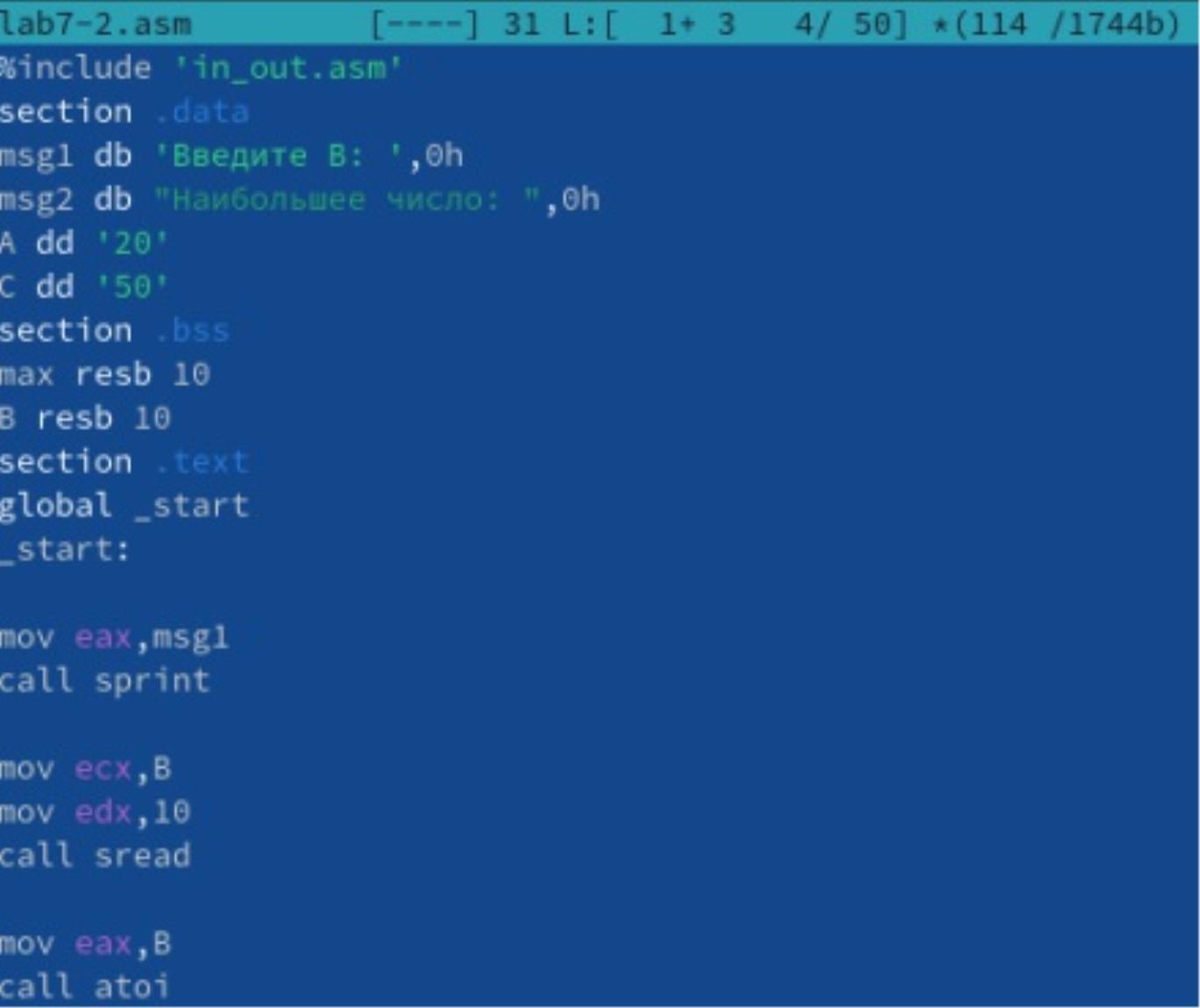
коду, не имеет адреса и машинного кода.

“3” - номер строки кода, “slen” - название функции, не имеет адреса и машинного кода.

“4” - номер строки кода, “00000000” - адрес строки, “53” - машинный код, “push

ebx” - исходный текст программы, инструкция “push” помещает операнд “ebx” в

стек.



**Возвращаемся** **в** **lab7-2.asm**

Открываю файл с программой lab7-2.asm и в выбранной мной инструкции с

двумя операндами удаляю выделенный операнд.

На выходе я не получаю ни одного файла из-за ошибки:инструкция mov (единственная в

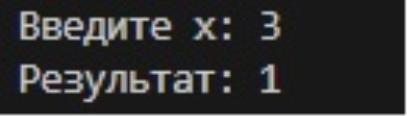
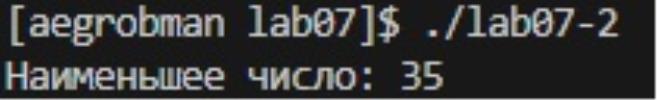
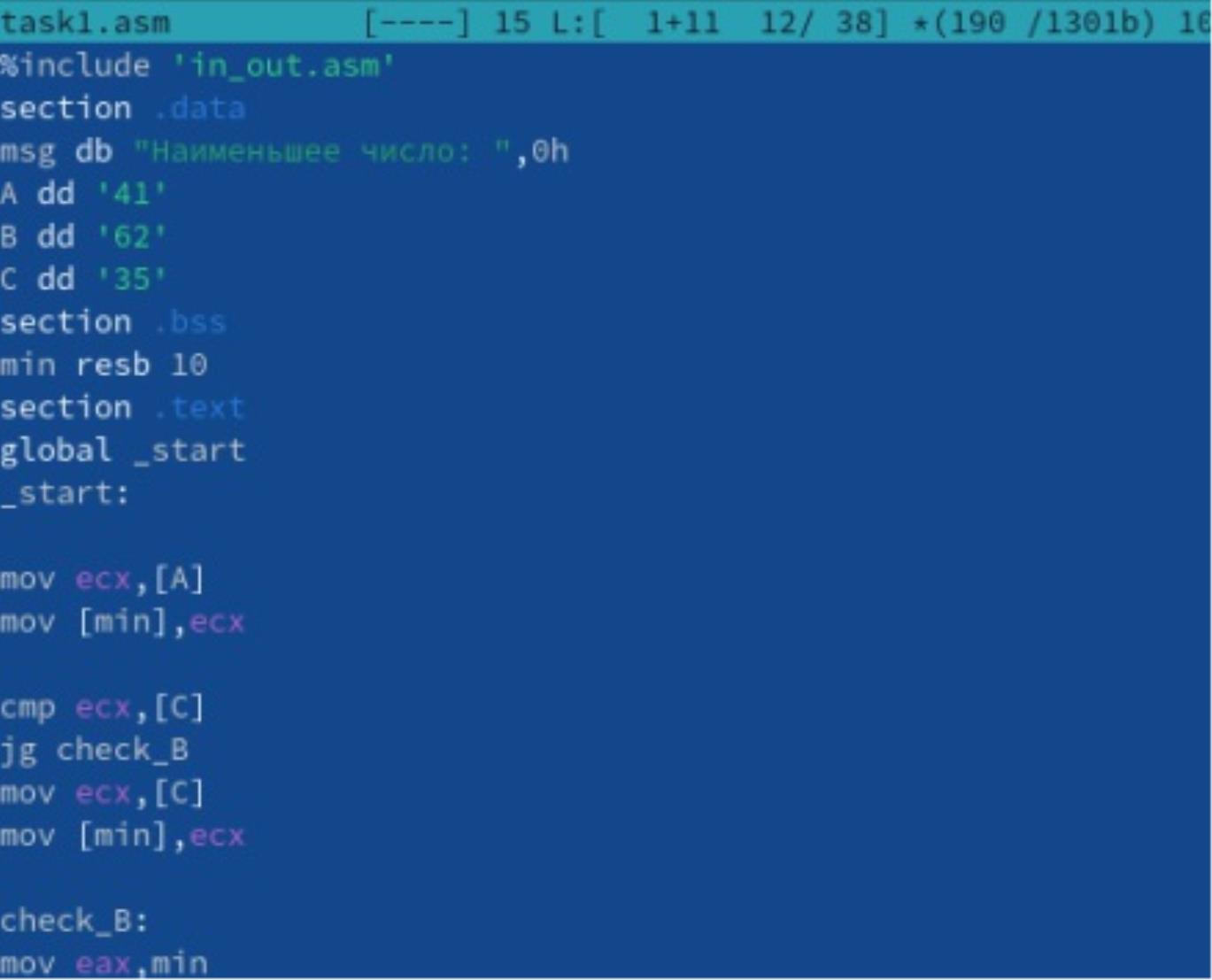
коде содержит два операнда) не может работать, имея только один

операнд, из-за чего нарушается работа кода.

3. **Задания** **для** **самостоятельной** **работы**

первый номер

и второй номер



5. **Вывод**

С помощью данной лабораторной работы я освоила арифметические инструкции языка

ассемблер NASM, что пригодится мне при выполнении последующих

лабораторных работ.

**Отправляем** **файлы** **на** **гитхаб.**

Ссылка на отчёт https://github.com/DaOneme/AEGrobman\_study\_2023-

2024\_arhpc/tree/main/Labs/Lab06 (https://github.com/DaOneme/AEGrobman\_study\_2023-

2024\_arhpc/tree/main/Labs/Lab06)

