РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7

Студент: Барханоева Раяна Магометовна

Ст.билет: 1032252468

Группа: НКАбд-01-25

МОСКВА

2025 г

Содержание

1. Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

2. Работа

Каталог (рисунок 1).

Рисунок 1.

Создала каталог и файл lab7-1.asm.

Программа (рисунок 2).

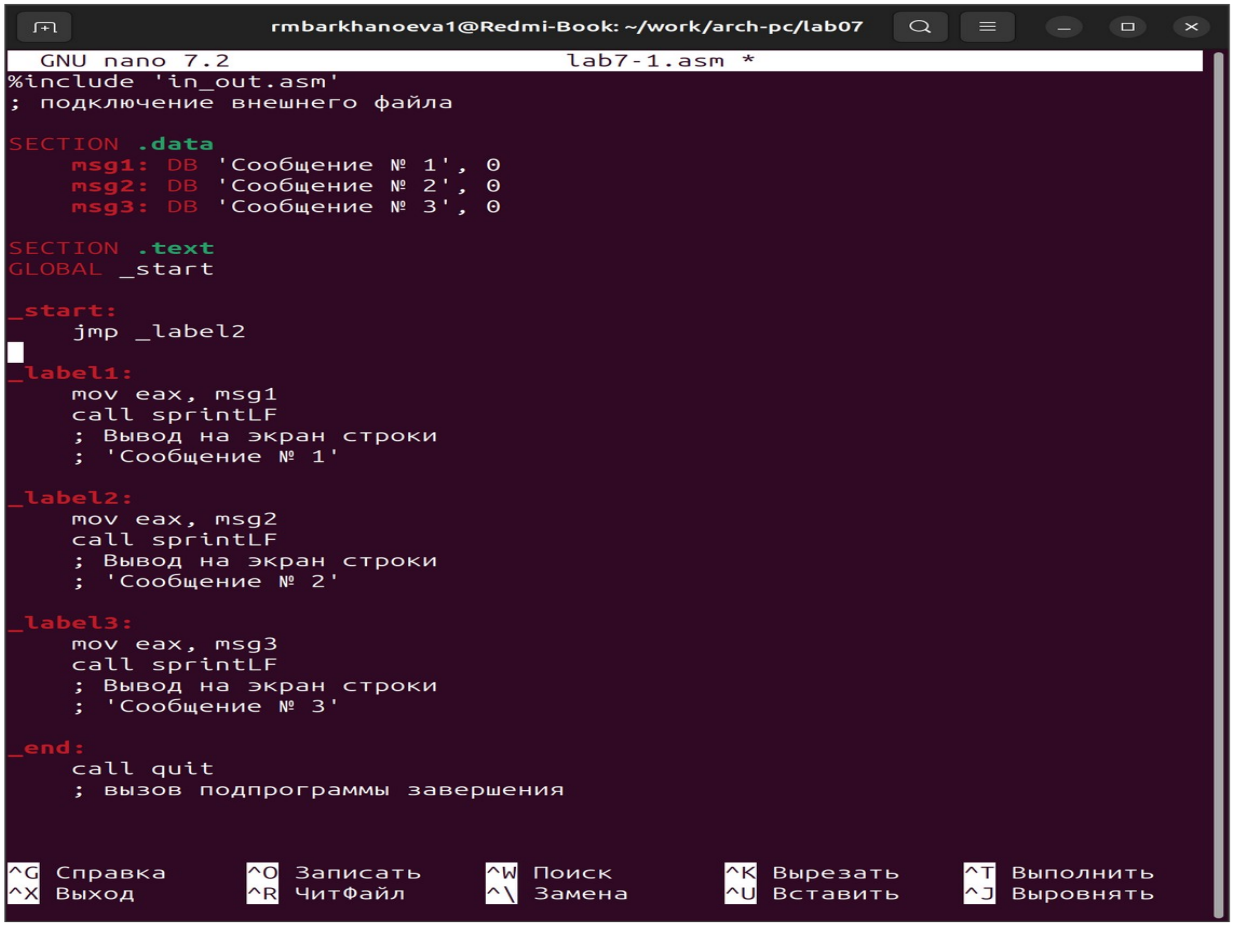


Рисунок 2.

Вставила программу с использованием инструкции jmp

Тест программы (рисунок 3).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.
o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
```

Рисунок 3.

Создали исполняемый файл и запустили программу. Использование инструкции jmp _label2 меняет порядок исполнения инструкций и позволяет

выполнить инструкции начиная с метки _label2, пропустив вывод первого сообщения.

Программа (рисунок 4).

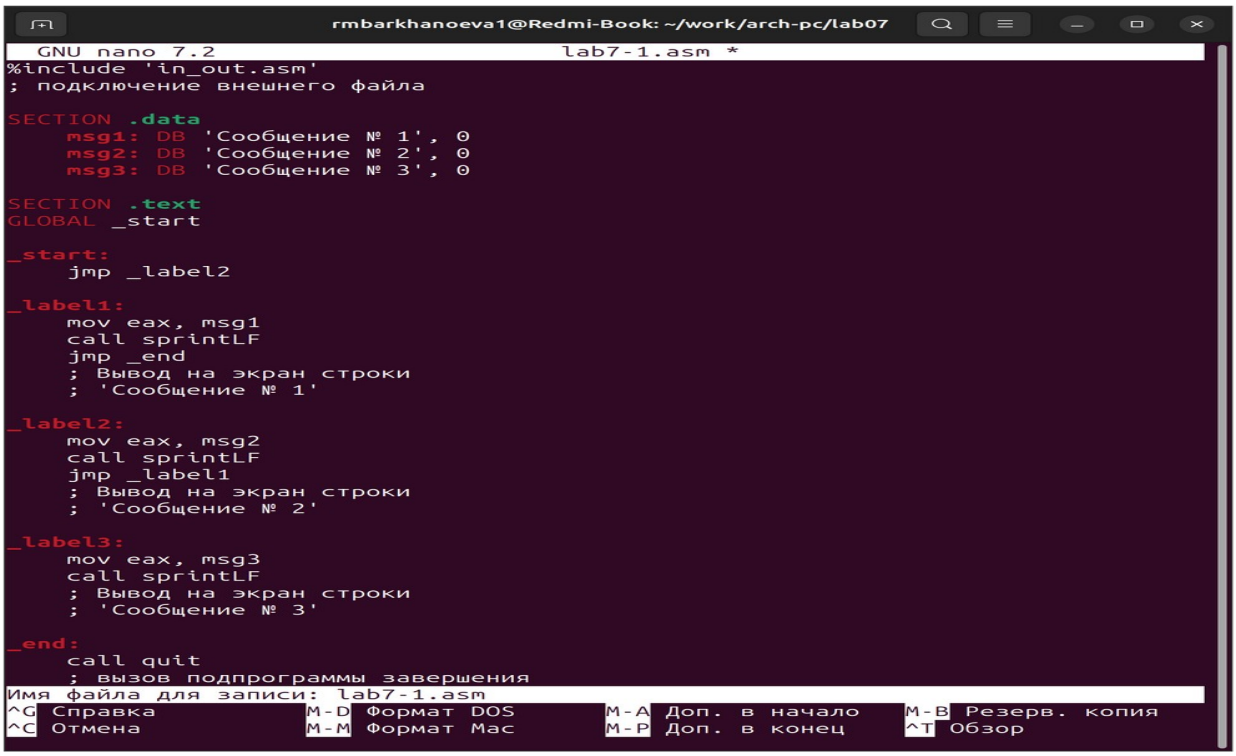


Рисунок 4.

Изменила немного программу.

Тест программы (рисунок 5.)

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.
o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
```

Рисунок 5.

Изменив программу, добавив 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), мы сделали так, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу.

Программа (рисунок).6

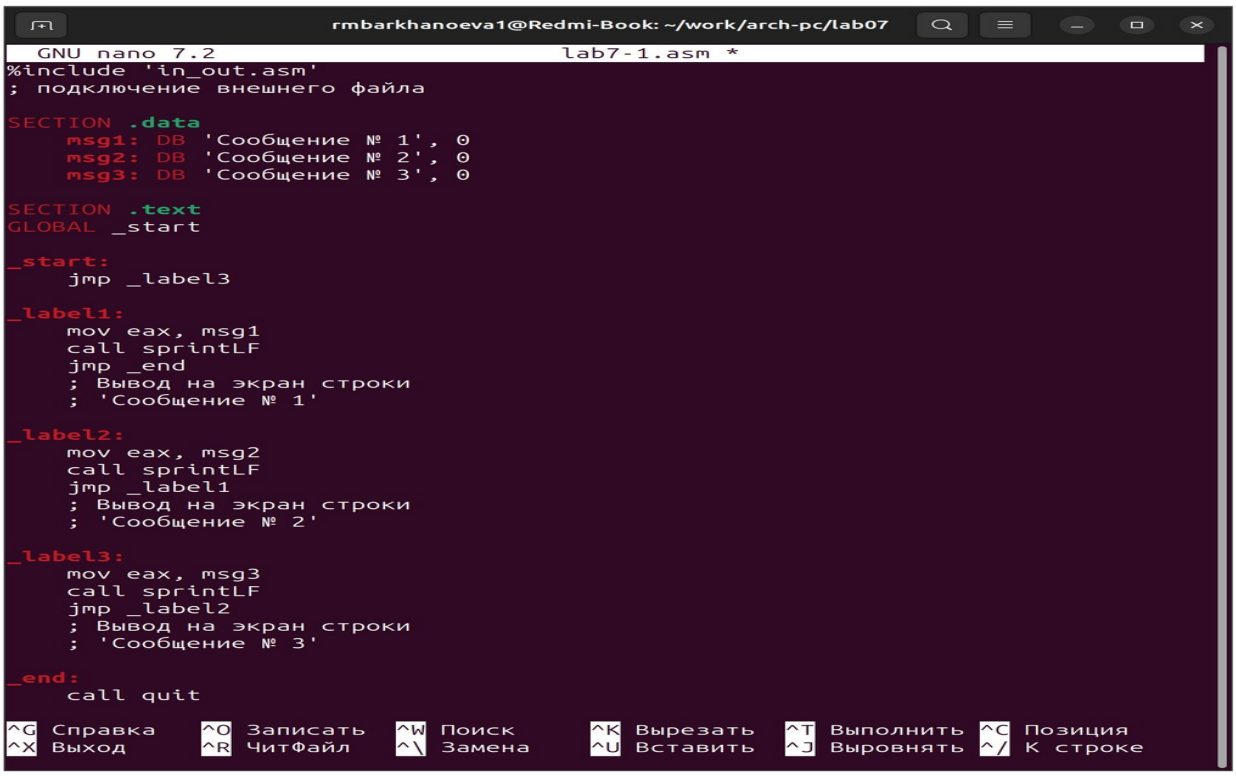


Рисунок 6.

Изменила программу, чтобы вывод был 3 2 1

Тест программы (рисунок 7).

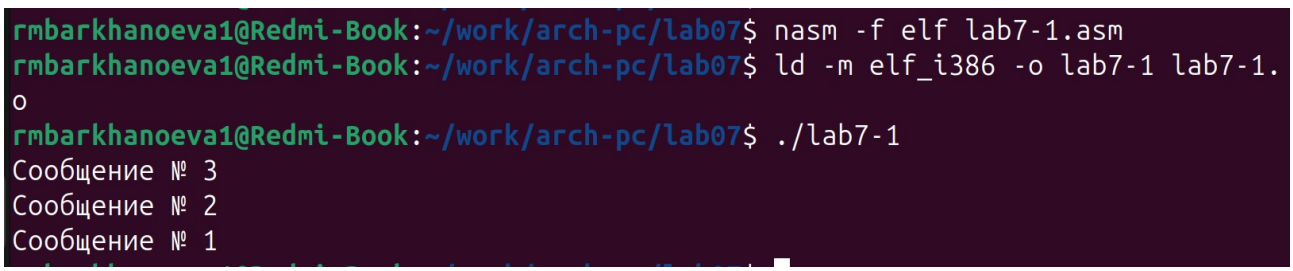
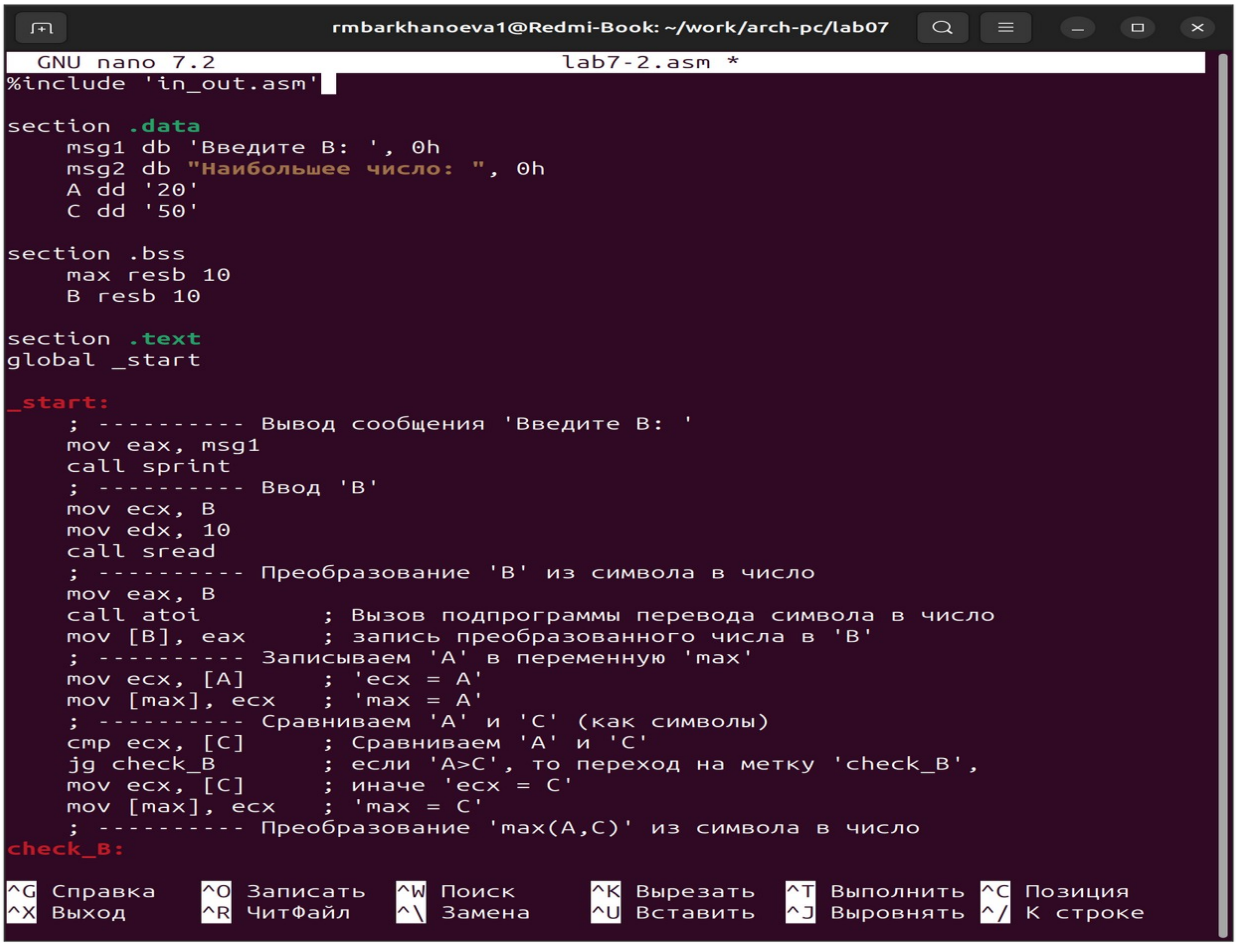


Рисунок 7.

Запустили и протестировали программу.

Программа (рисунок 8).

Рисунок 8.



Программа которая выводит наибольшее число из 3.

Тест программы (рисунок 9).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.
o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 10
Наибольшее число: 50
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 5
Наибольшее число: 50
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 60
Наибольшее число: 60
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$
```

Рисунок 9.

Выводит наибольшее число

Создание файла (рисунок 10).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2
.asm
```

Рисунок 10.

Создали файл литстинга.

2.1. Ответы на вопросы:

1. jg fin ; если 'max(A,C)>B', то переход на 'fin', :

Инструкция `jg` (jump if greater — переход если больше) выполняет условный переход, если результат предыдущего сравнения показал, что первое значение больше второго (при знаковом сравнении).

2. mov ecx, [B] ; иначе 'ecx = B' :

Эта инструкция выполняется только если переход `jg fin` не сработал, то есть когда `max(A, C) ≤ B`.
Значение переменной `B` загружается из памяти в регистр `ecx`.

3. mov [max], ecx:

Здесь значение из регистра `ecx` записывается в переменную `max`. Поскольку в `ecx` находится `B`, это означает, что максимальным значением считается `B`.

Выходные файлы (рисунок 11).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ls
in_out.asm  lab7-1.asm  lab7-2      lab7-2.lst
lab7-1      lab7-1.o    lab7-2.asm  lab7-2.o
```

Рисунок 11.

3. Самостоятельная работа

Программа (рисунок 12).

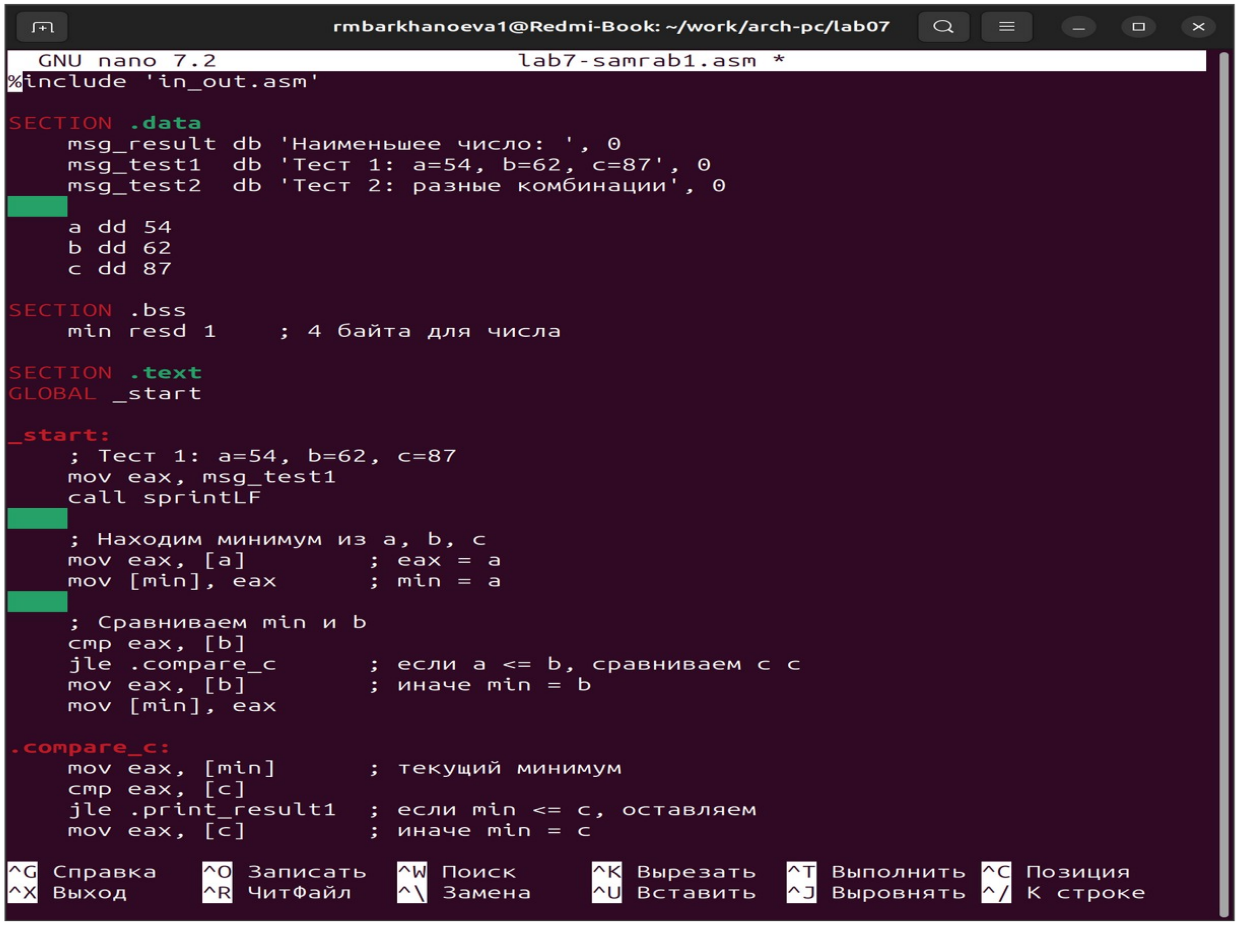


Рисунок 12.

Это программа программа нахождения наименьшей из 3 целочисленных переменных a,b и C

Тест программы (рисунок 13).

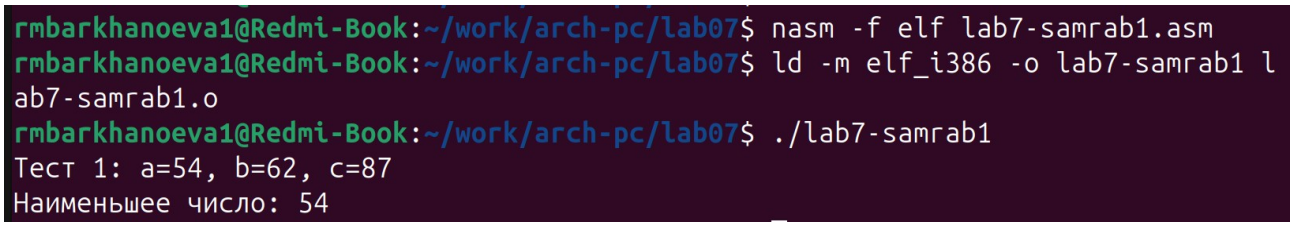


Рисунок 13.

Запуск программы. Вывод наименьшего числа.

Программа (рисунок 14).

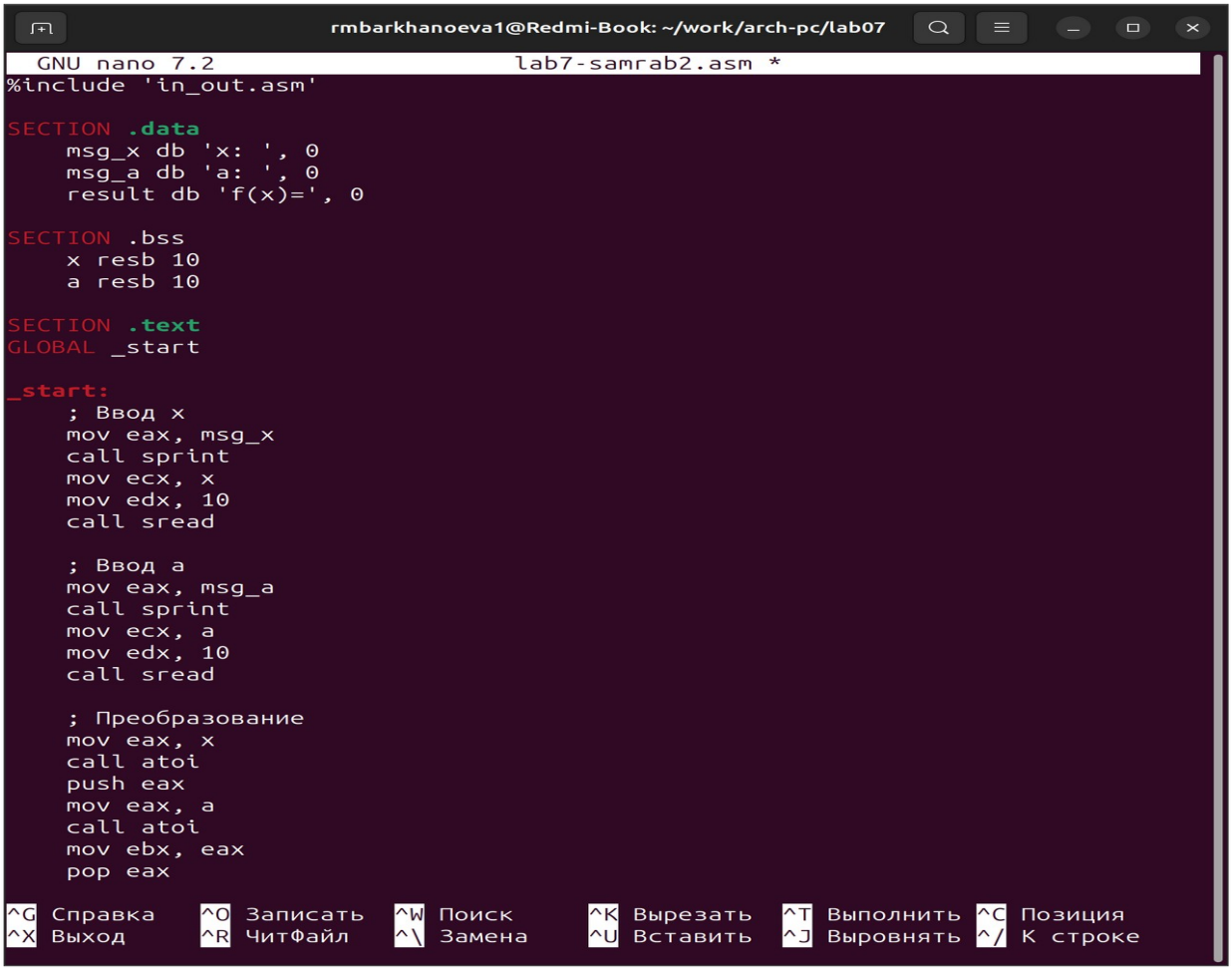


Рисунок 14.

Программа, которая для введенных с клавиатуры значений x и a вычисляет

значение заданной функции $f(x)$ и выводит результат вычислений.

Тест программы (рисунок 14).

```
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ nasm -f elf lab7-samrab2.asm
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-samrab2 l
ab7-samrab2.o
rmbarkhanoeva1@Redmi-Book:~/work/arch-pc/lab07$ ./lab7-samrab2
x: 10
a: 5
f(x)=10
```

Рисунок 14.

Создали исполняемый файл и проверили на корректность.

4. Вывод

В ходе лабораторной работы были изучены команды условного и безусловного переходов в ассемблере NASM. Приобретены навыки написания программ с использованием инструкций jmp для безусловных переходов и различных вариантов jcc (например, je, jne, jq) для реализации ветвлений на основе состояния флагов процессора. Изучена работа регистра флагов и команды сравнения cmp, которая устанавливает эти флаги для последующего

анализа. Также освоена структура файла листинга, создаваемого транслятором, и его роль в отладке программ, что позволяет анализировать соответствие исходного кода сгенерированному машинному коду и смещениям.