Шаблон отчёта по лабораторной работе №7

Мугари Абдеррахим , НКАбд-03-22

Содержание

# 1 Цель работы :

В седьмой лабораторной работе можно будет освоить **арифметические операции** языка ассемблера.

# 2 Выполнение лабораторной работы :

## 2.1 Символьные и численные данные в NASM :

1. Здесь мы начали с создания, а затем переместились в седьмой каталог лаборатории “*~/work/arch-pc/lab07*”, после чего мы создали файл “**lab7-1.asm**”. (рис. 1)

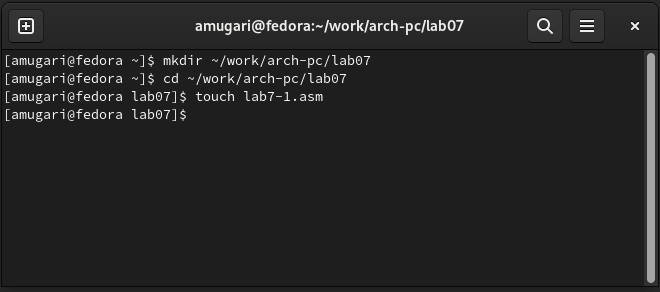


Рис. 1: Ресунок 1

1. После этого мы заполнили файл **.asm** кодом программы, отображающей значение регистра **eax**. (рис. 2)

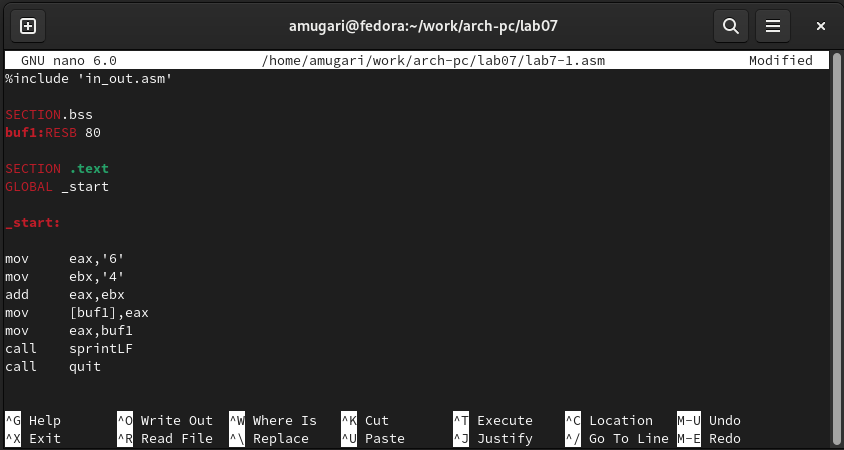


Рис. 2: Ресунок 2

* Затем мы скомпилировали файл, создали исполняемый файл и запустили программу, все это после перемещения файла **in\_out.asm** в тот же каталог, где находится **lab7-1.asm**. (рис. 2)

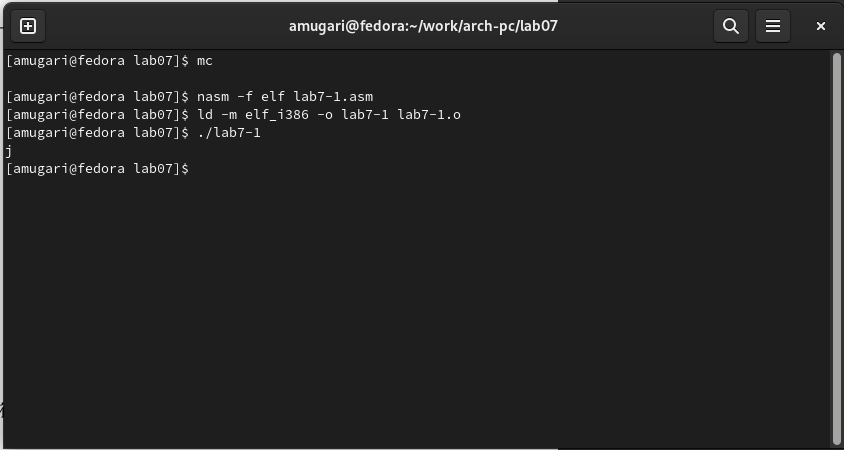


Рис. 3: Ресунок 3

1. После этого мы изменили код в листинге следующим образом :  mov eax,6  
   mov ebx,4

(рис. 4)

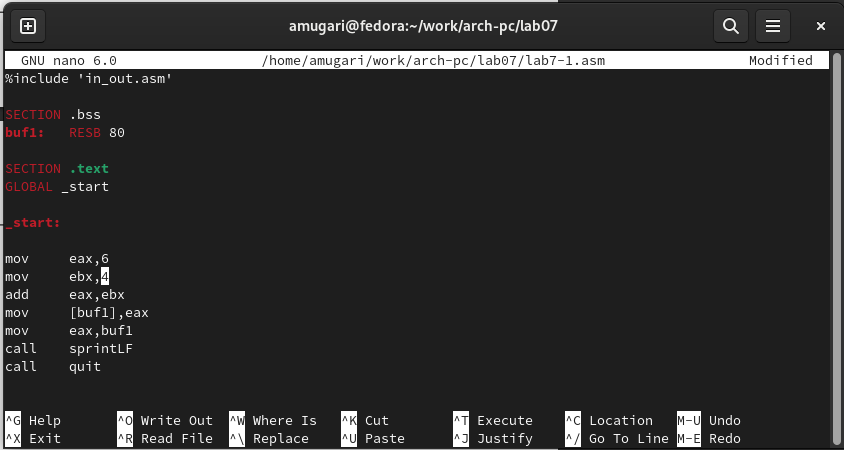


Рис. 4: Ресунок 4

* Затем мы снова скомпилировали файл и создали исполняемый файл. (рис. 5)



Рис. 5: Ресунок 5

* Проверив **ASCII tbale** символ, соответствующий коду **10** это **новая строка**, и мы можем сказать, что это было отображено, потому что при запуске программы она отобразила новую строку в качестве вывода.

1. После этого мы создали файл **lab-2.asm**, в котором мы использовали подпрограммы, расположенные в файле **in\_out.asm**. (рис. 6)

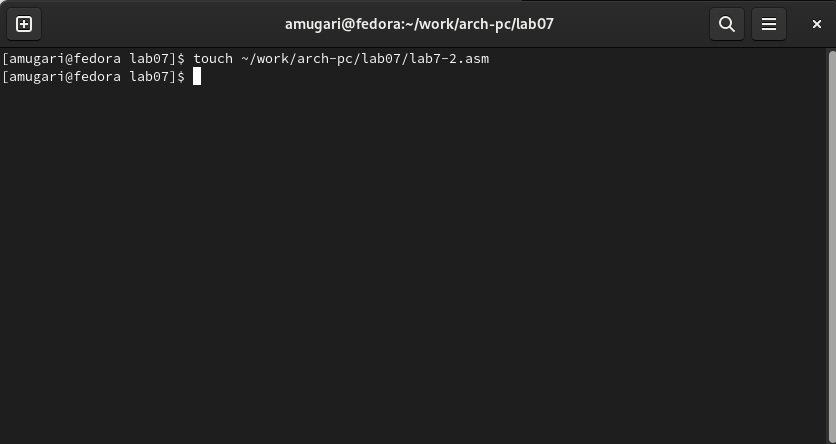


Рис. 6: Ресунок 6

* После этого мы заполнили файл необходимым кодом для вывода значения реестра с помощью подпрограммы. (рис. 7)

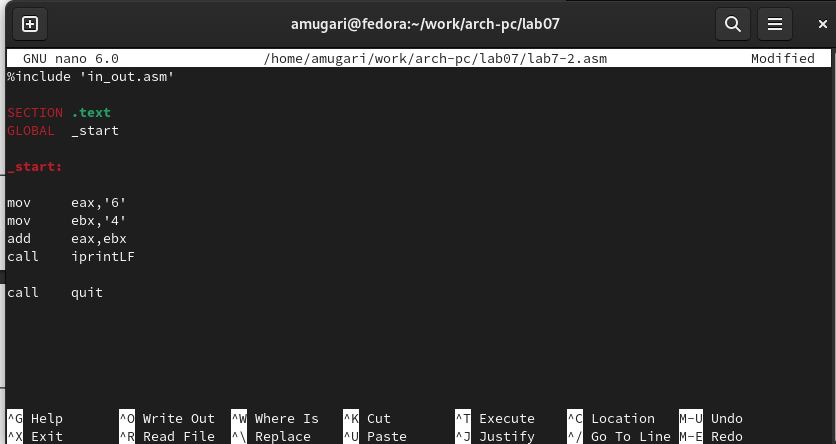


Рис. 7: Ресунок 7

* мы скомпилировали файл, создали исполняемый файл и запустили его. (рис. 8)

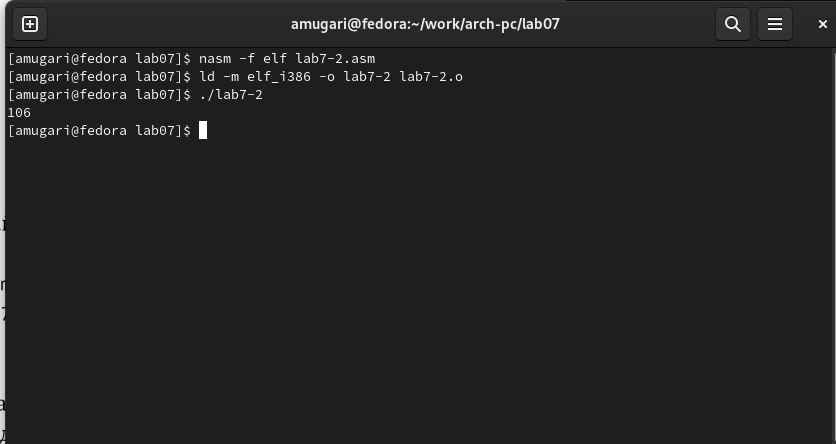


Рис. 8: Ресунок 8

1. Аналогично предыдущему примеру, мы меняем символы на цифры, заменяя строки на :  
   mov eax,6  
   mov ebx,4

. (рис. 9)



Рис. 9: Ресунок 9

* Затем мы снова скомпилировали файл и создали исполняемый файл. (рис. 10)

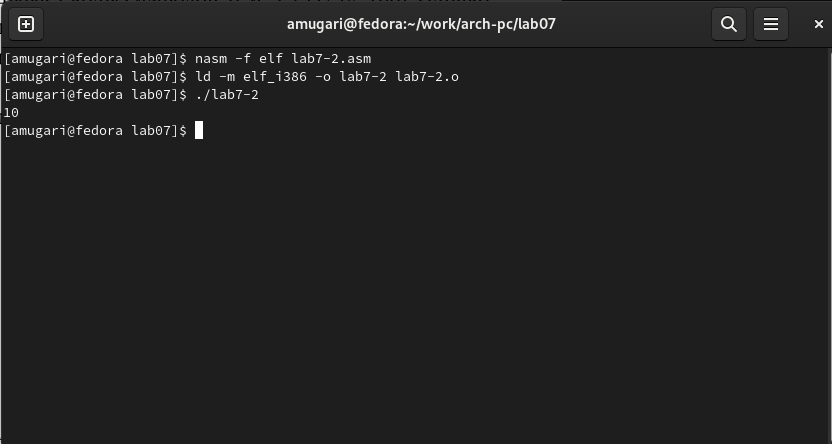


Рис. 10: Ресунок 10

* На этот раз результатом, который мы получили, действительно было добавление **6** и **4** который **10**.
* Затем мы заменили функцию **iprintLF** на **iprint**. После этого был создан исполняемый файл, и мы запустили его. (рис. 11) (рис. 12)



Рис. 11: Ресунок 11

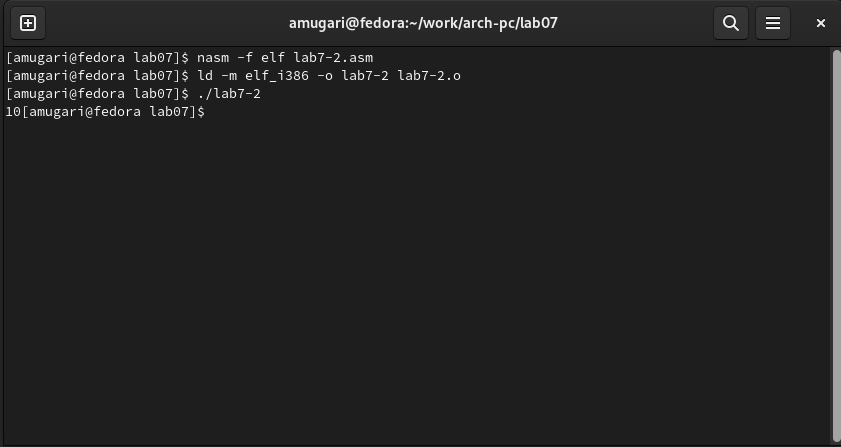


Рис. 12: Ресунок 12

* Когда мы использовали подпрограмму **iprint**, мы заметили, что вывод отличается от предыдущего,потому что при использовании **iprint** не создается новая строка после вывода.

## 2.2 Выполнение арифметических операций в NASM :

1. В качестве примера выполнения арифметических операций в NASM при- ведем программу вычисления арифметического выражения

* Мы создали файл lab7-3.asm в каталоге **~/work/arch-pc/lab07**. (рис. 13)

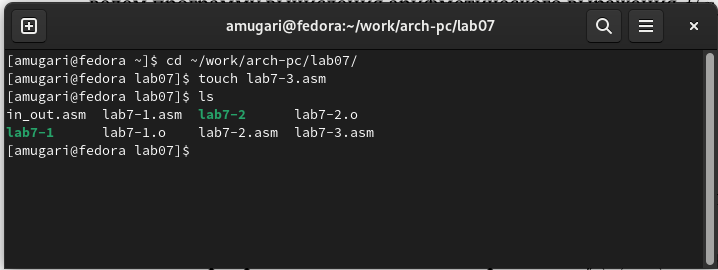


Рис. 13: Ресунок 13

1. Затем мы заполнили файл необходимым кодом. (рис. 14)

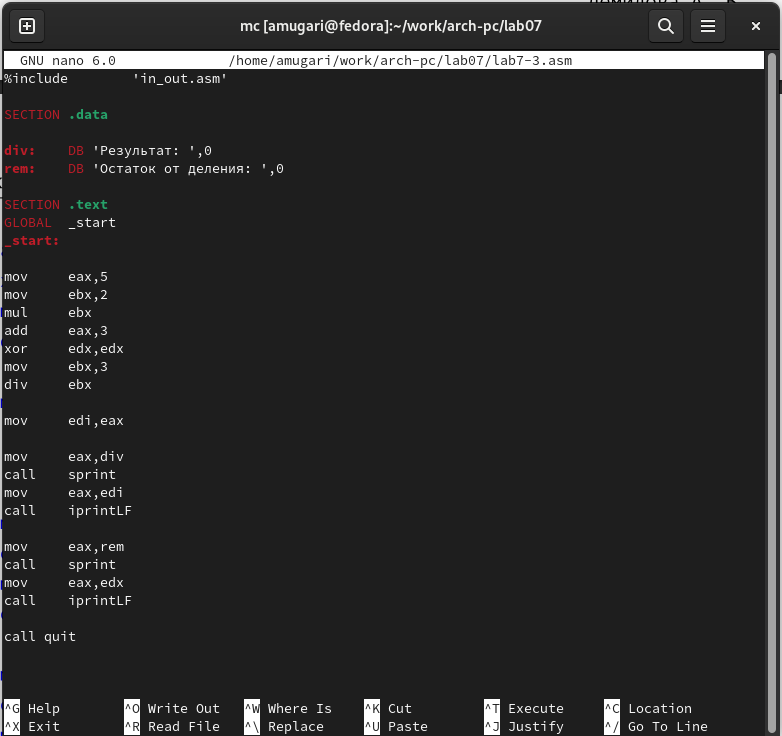


Рис. 14: Ресунок 14

* Создали исполняемый файл и запустили его. (рис. 15)

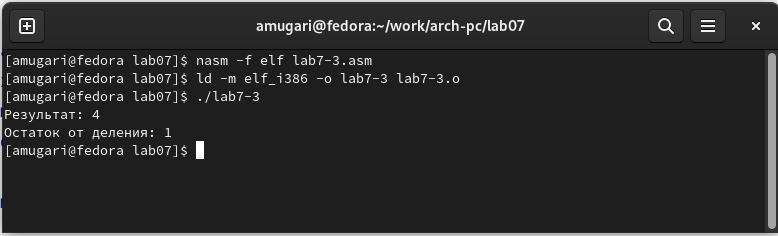


Рис. 15: Ресунок 15

* Затем мы изменили текст программы, чтобы вычислить выражение:  
    
  (рис. 16)



Рис. 16: Ресунок 16

* мы создали исполняемый файл и проверили его работу. (рис. 17)

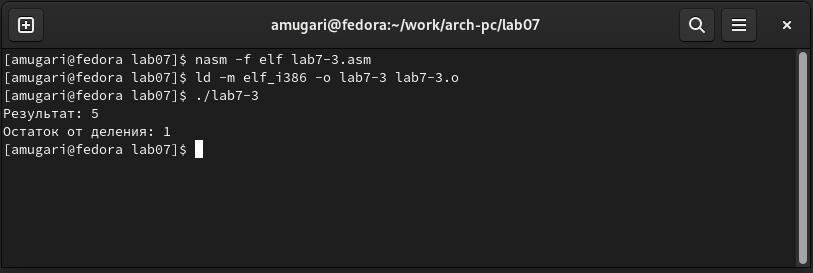


Рис. 17: Ресунок 17

1. На этом шаге мы написали программу, которая может вычислить дисперсию, которую мы получаем из номера студенческого билета.

* Мы начали с создания файла **variant.asm**. (рис. 18)

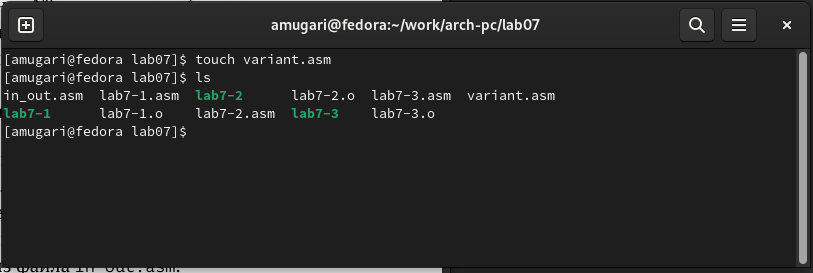


Рис. 18: Ресунок 18

* После этого мы написали код программы. (рис. 19)

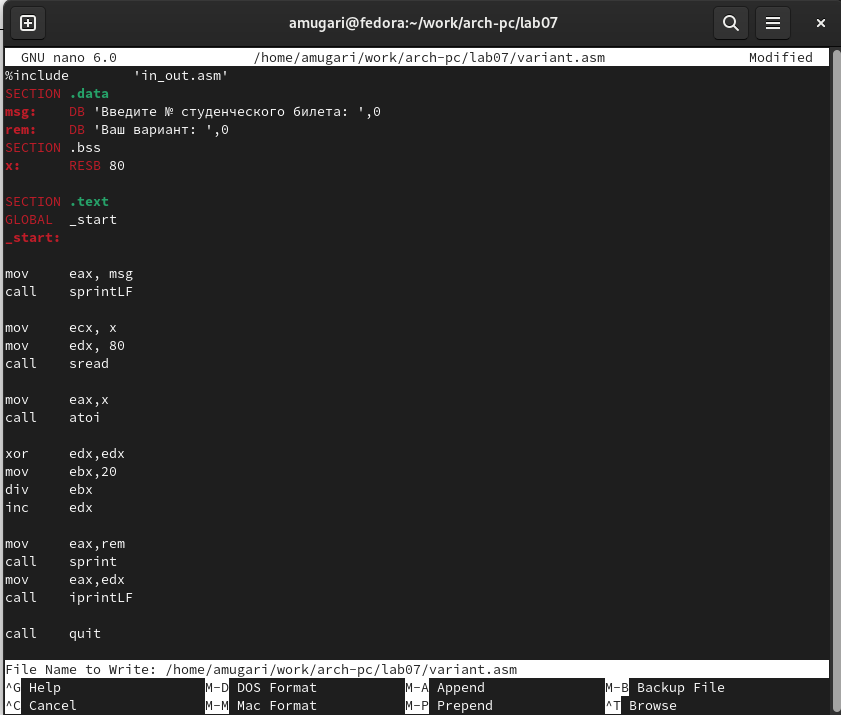


Рис. 19: Ресунок 19

* мы создали исполняемый файл и проверили его работу, и действительно, в зависимости от номера студента он генерирует номер варианта. (рис. 20)

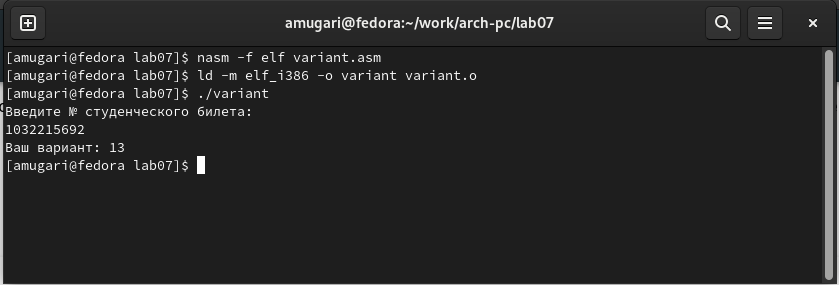


Рис. 20: Ресунок 20

### 2.2.1 Вопросы :

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

**O:**  
rem: DB ‘Ваш вариант:’,0  
mov eax,rem  
call sprint

1. Для чего используется следующие инструкции?  
   mov ecx, x  
   mov edx, 80  
   call sread

**O:** Эти инструкции были использованы для того, чтобы позволить пользователю вводить данные

1. Для чего используется инструкция “call atoi”?

**O:** Эта инструкция используется для преобразования значения x из **ASCII-кода** в целое число.

1. Какие строки листинга 7.4 отвечают за вычисления варианта?

**O:**  
xor edx,edx  
mov ebx,20  
div ebx  
inc edx

1. В какой регистр записывается остаток от деления при выполнении ин- струкции “div ebx”?

**O:**Остаток был записан в регистре **edx**

1. Для чего используется инструкция “inc edx”?

**O:** Эта инструкция была использована для увеличения значения в регистре **edx**

1. Какие строки листинга 7.4 отвечают за вывод на экран результата вычис- лений?

**O:**  
mov eax,edx  
call iprintLF

## 2.3 Выводы по результатам выполнения заданий :

* В ходе лабораторной работы мы освоили выполнение арифметических операций на языке ассемблера и углубились в использование подпрограммы.

# 3 Задание для самостоятельной работы :

1. В этой работе нам пришлось написать программу, которая просит пользователя ввести значение переменной и решить математическое выражение.

* Мой вариант : **13**
* математическое выражение
* Итак, мы начали с создания asm-файла, в котором будет находиться наш код. (рис. 21)

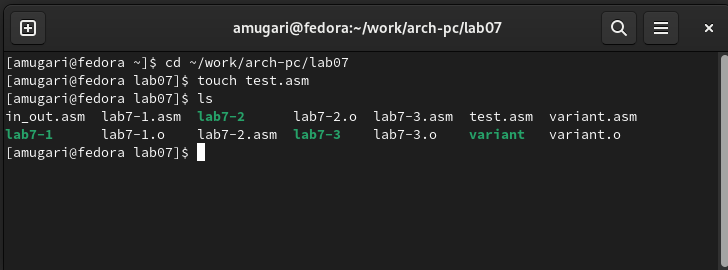


Рис. 21: Ресунок 21

* После этого мы написали код нашей программы. (рис. 22)

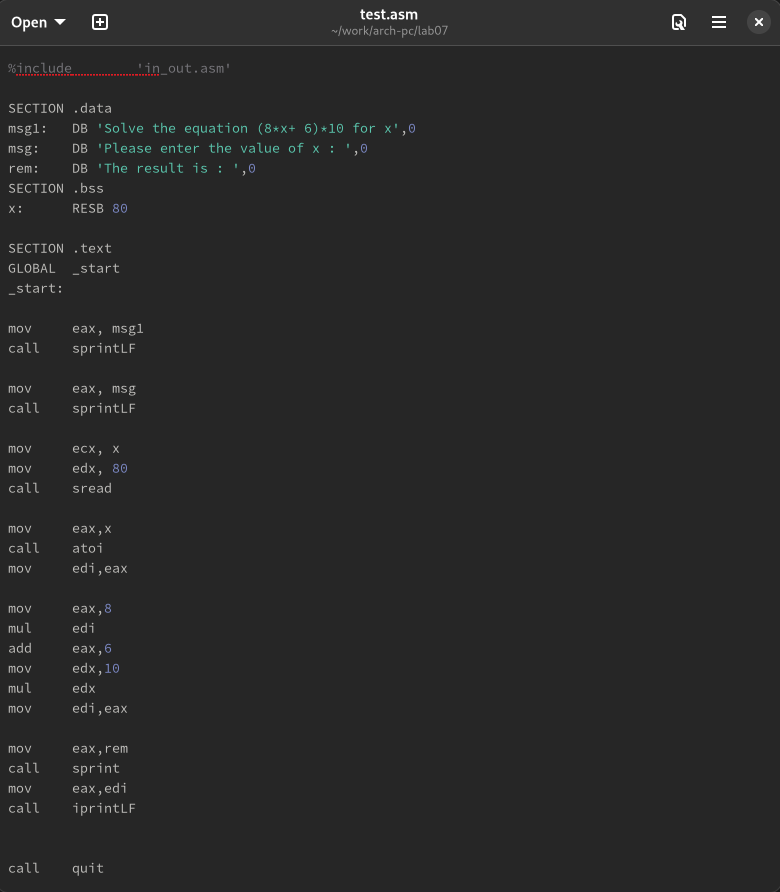


Рис. 22: Ресунок 22

* и, наконец, мы проверяем корректность кода, который мы написали, используя два разных значения

Как указано на следующем рисунке (рис. 23)

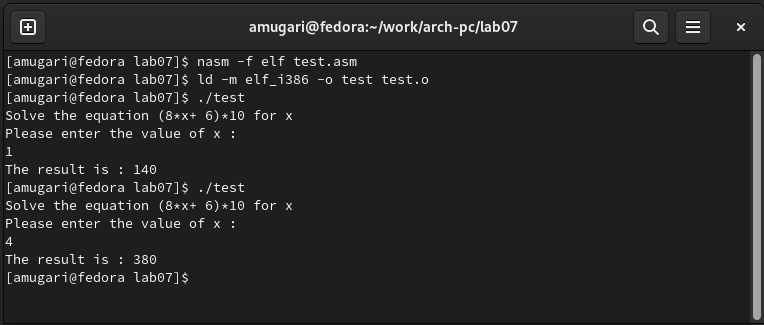


Рис. 23: Ресунок 23

## 3.1 Выводы по результатам выполнения заданий :

В этой части мы смогли узнать, как преобразовать некоторые математические идеи в реальный код на ассемблере, что помогло нам получить более глубокое представление о том, как работать с регистрами.

# 4 Выводы, согласованные с целью работы :

* В седьмой лаборатории мы в основном научились писать программы, выполняющие арифметические операции, и научились вычислять математические выражения средней сложности.