Отчёт по лабораторной работе №6

Дисциплина: архитектура компьютеров

Симонова Виктория Игоревна

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

# 3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. Далее рассмотрены все существующие способы задания адреса хранения операндов – способы адресации. Существует три основных способа адресации: • Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: mov ax,bx. • Непосредственная адресация – значение операнда задается непосредственно в команде, Например: mov ax,2. • Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию. Арифметические операции в NASM. Схема команды целочисленного сложения add (от англ. addition - добавление) выполняет сложение двух операндов и записывает результат по адресу первого операнда. Команда add работает как с числами со знаком, так и без знака и выглядит следующим образом: add ,

# 4 Выполнение лабораторной работы

##Символьные и численные данные в NASM Создаю каталог для программам лабораторной работы № 6 (рис. [[1](#fig:001)]).

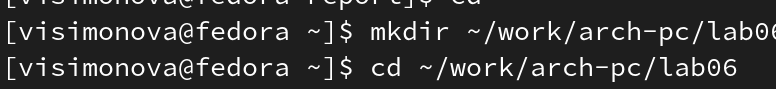


Figure 1: Создание каталога

Перейхожу в него и создаю файл lab6-1.asm (рис. [[2](#fig:002)]).

Figure 2: Создание файла

Figure 2: Создание файла

Копирую в текущий каталог файл in\_out.asm, тк он будет использоваться в последующих программах (рис. [[3](#fig:003)]).

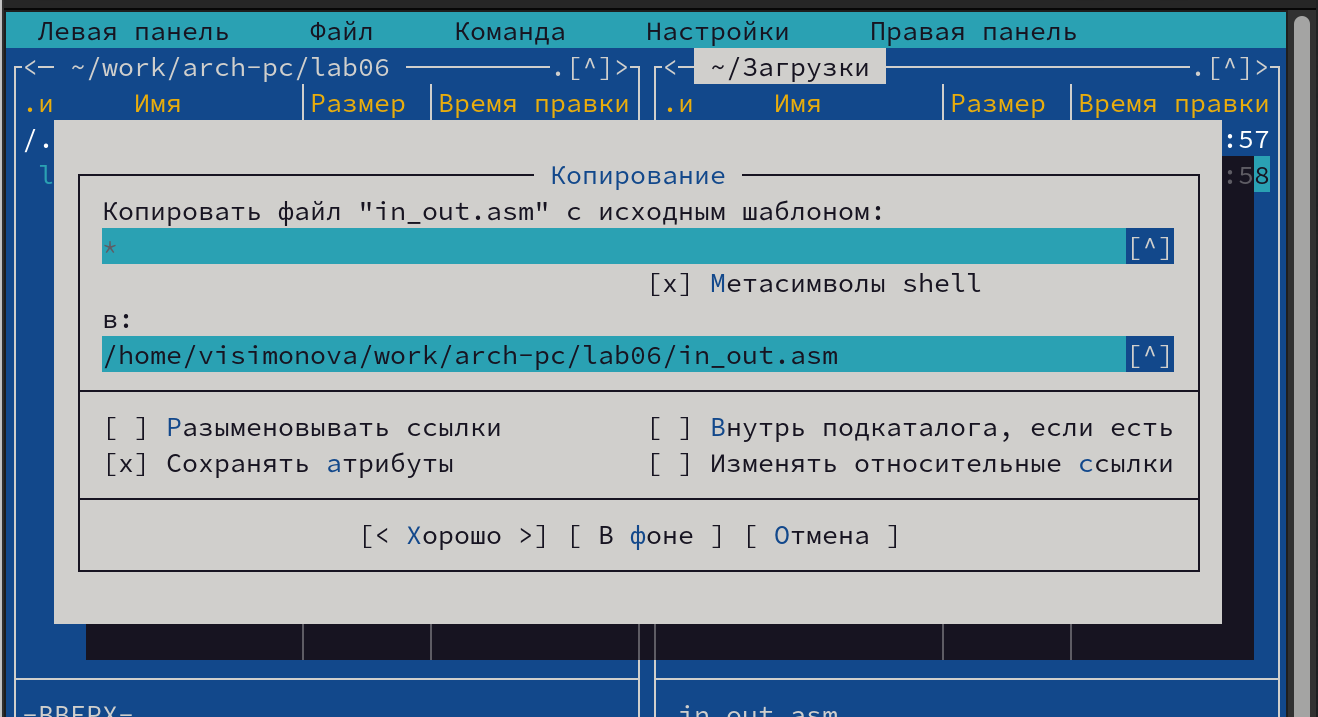


Figure 3: Копирую файл in\_out

В файл lab6-1.asm ввожу программу для вывода значений регистра eax (рис. [[4](#fig:004)]).

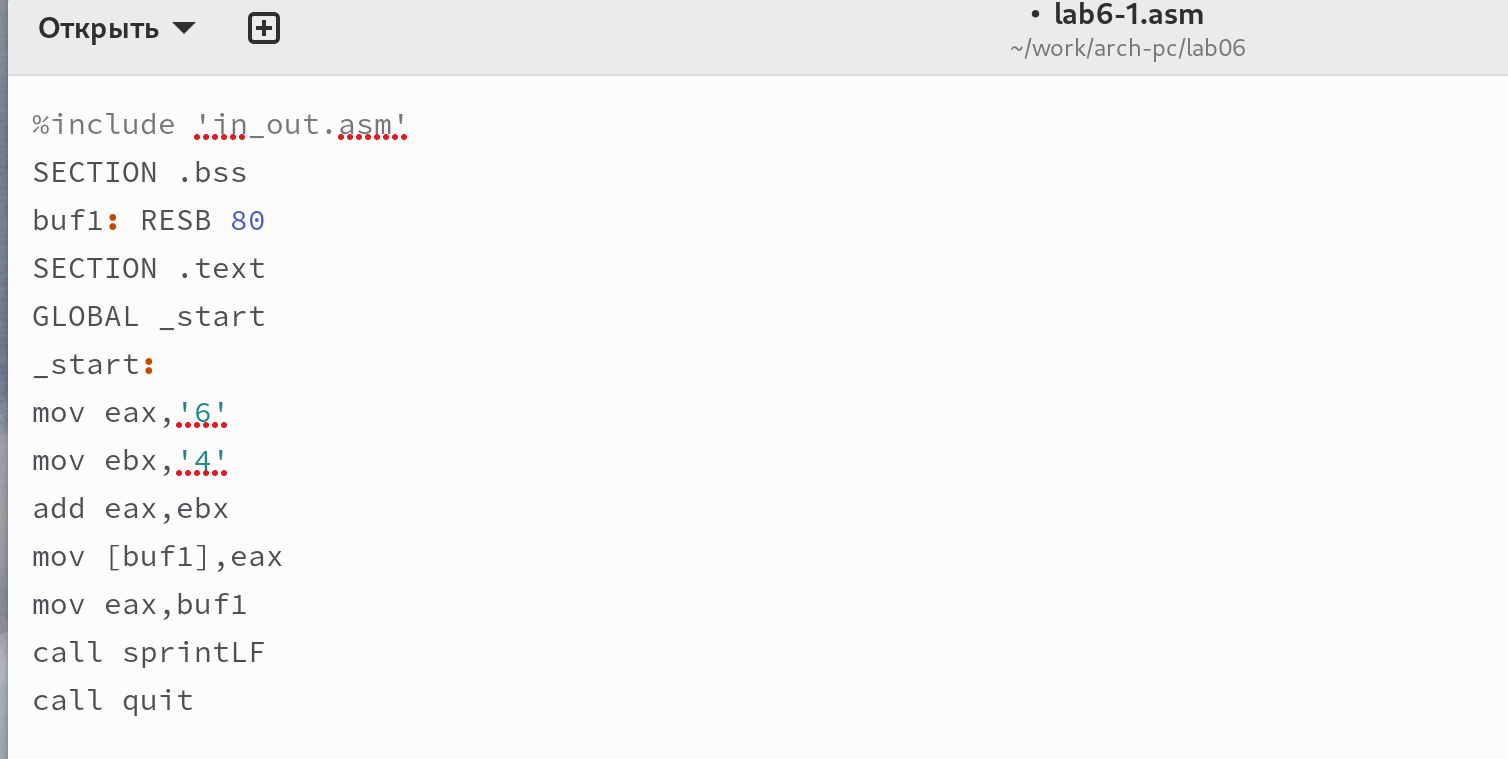


Figure 4: Ввожу текст в файл

Создаю и запускаю исполняемый файл, который выводит символ j, тк сумма двоичных кодов 4 и 6 по системе ASCII соответствует символу j (рис. [[5](#fig:005)]).



Figure 5: Создаю и запускаю исполняемый файл

Изменяю ‘6’ и ‘4’ на цифры 6 и 4 (рис. [[6](#fig:006)]).

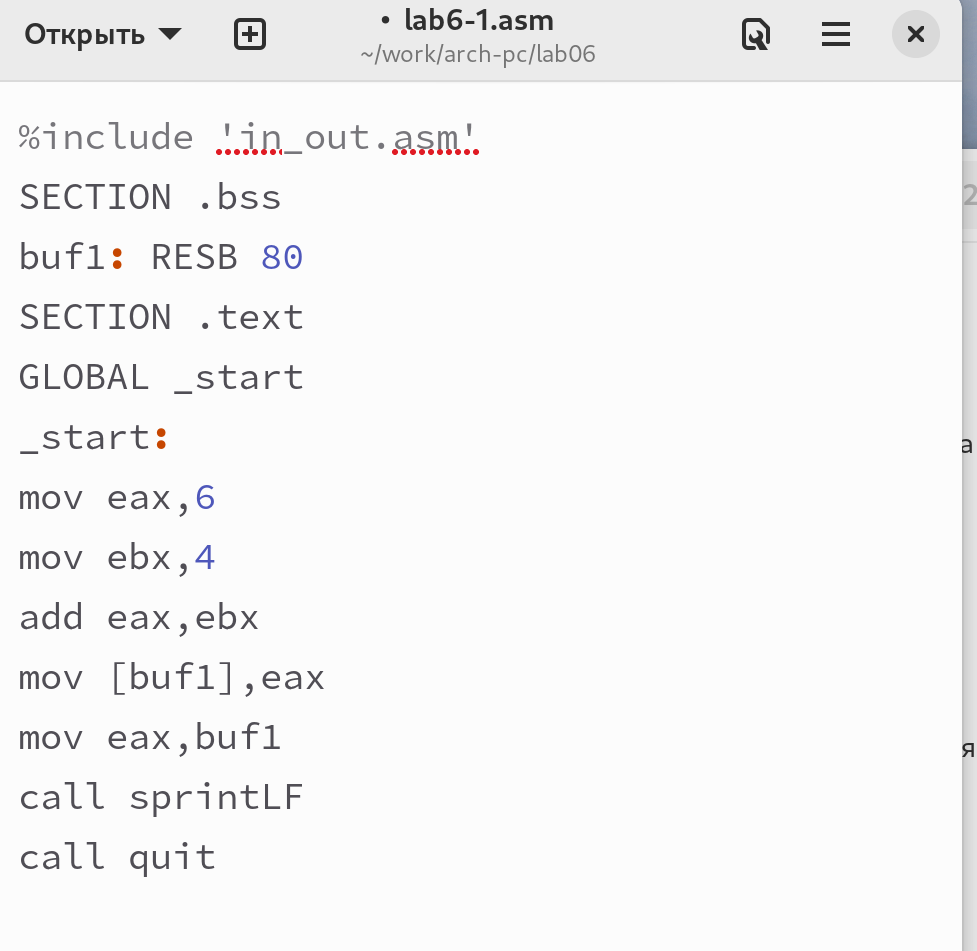


Figure 6: Изменяю файл

Создаю и запускаю исполняемый файл. Теперь выводится символ с кодом 10, это символ перевода строки и он не отображается при выводе на экран (рис. [[7](#fig:007)]).

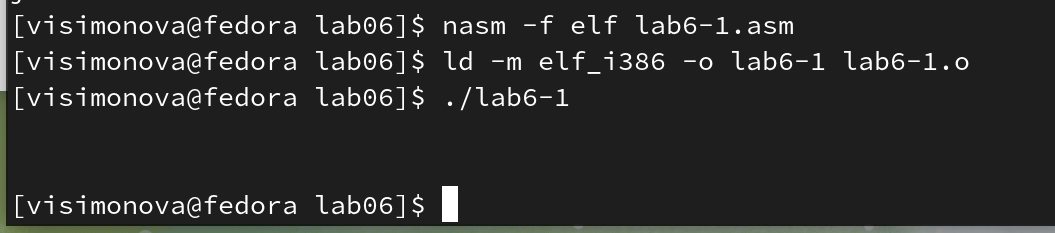


Figure 7: Создаю и запускаю исполняемый файл

Создаю файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 (рис. [[8](#fig:008)]).

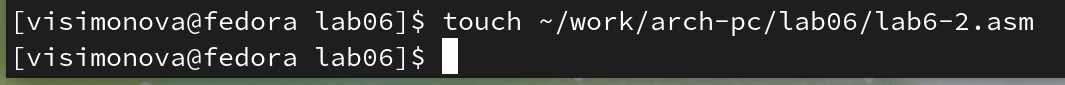


Figure 8: Создаю файл

Ввожу в него текст программы для вывода значения регистра ebx (рис. [[9](#fig:009)]).

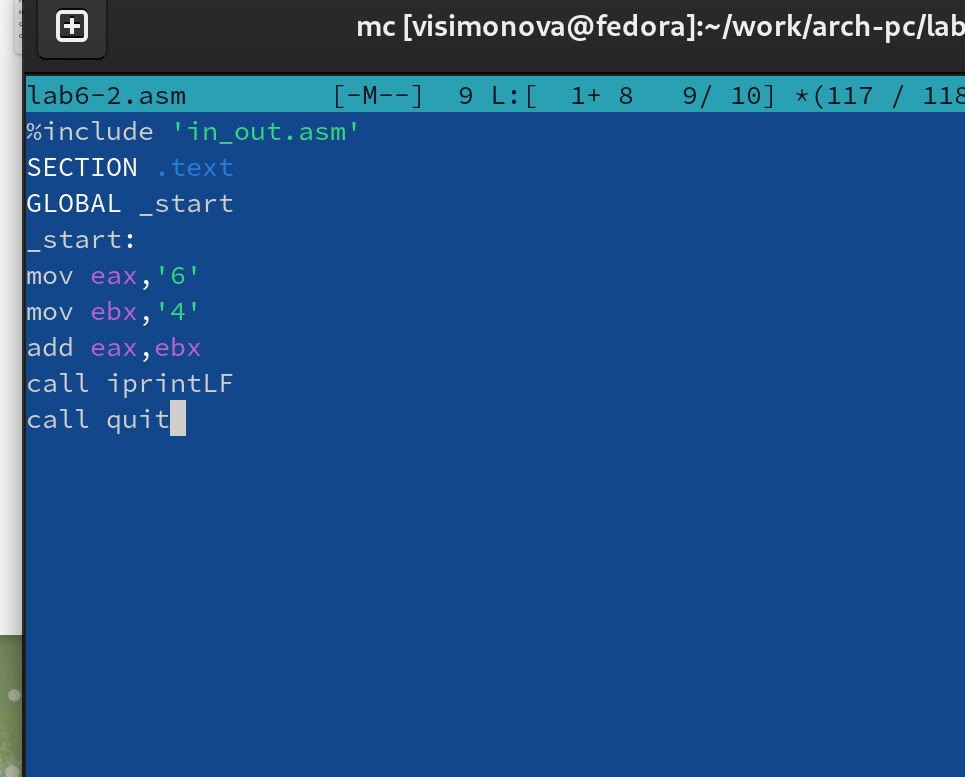


Figure 9: Ввожу код в файл

Создаю и запускаю исполняемый файл lab6-2, программа вывела число 106 , программа позволяет вывести именно число , а не символ, но мы всё ещё складываем коды (рис. [[10](#fig:010)]).

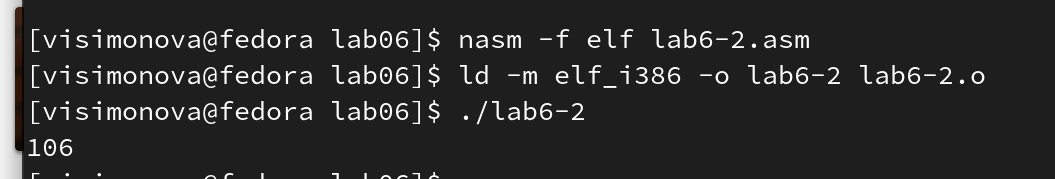


Figure 10: Создаю и запускаю исполняемый файл

Изменяю ‘6’ и ‘4’ на цифры 6 и 4 (рис. [[11](#fig:011)]).



Figure 11: Исправляю

Создаю и запускаю новый исполняемый файл lab6-2, программа вывела число 10, тк программа складывает сами числа (рис. [[12](#fig:012)]).

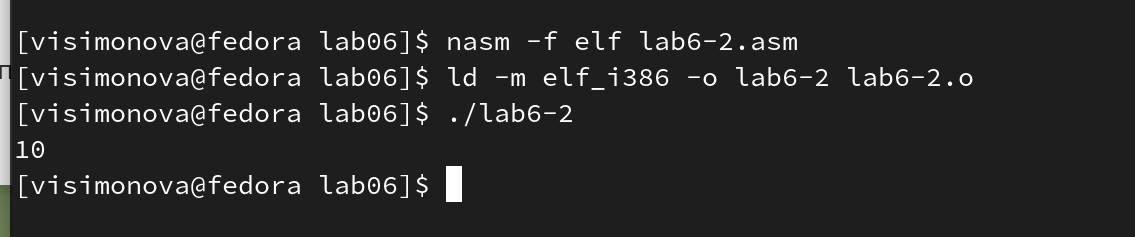


Figure 12: Запуск исполняемого файла с имправленным кодом

Заменяю функцию inprintLF на inprint (рис. [[13](#fig:013)]).

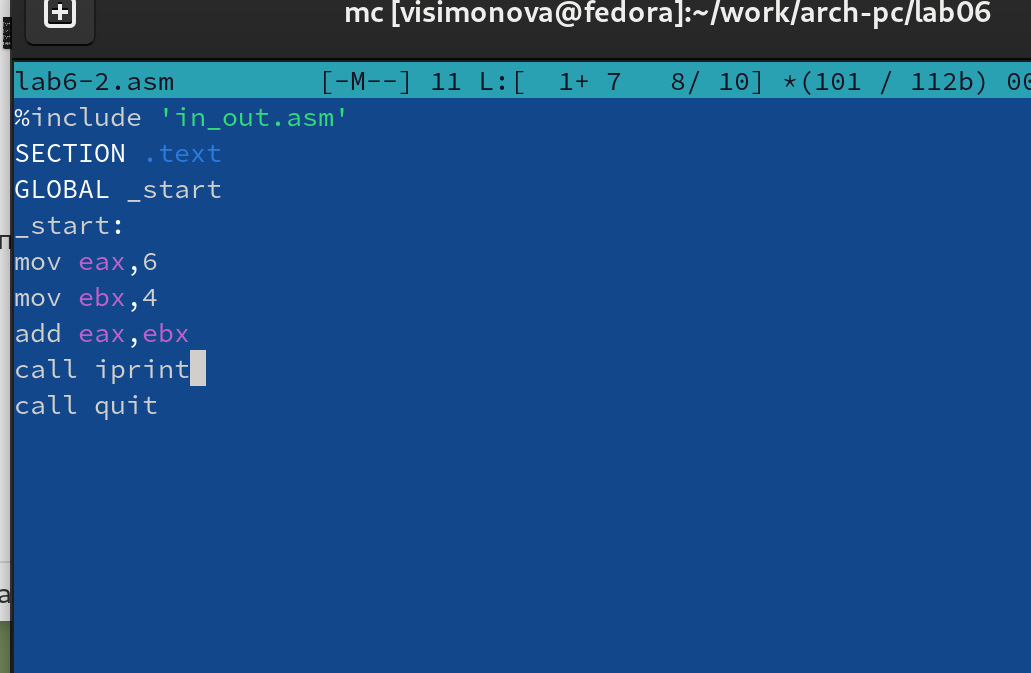


Figure 13: Меняю на inprint

Создаю и запускаю новый исполняемый файл lab6-2, программа вывела число 10, вывод никак не отличается, тк символ переноса строки не отображался, а сейчас его просто нет (рис. [[14](#fig:014)]).

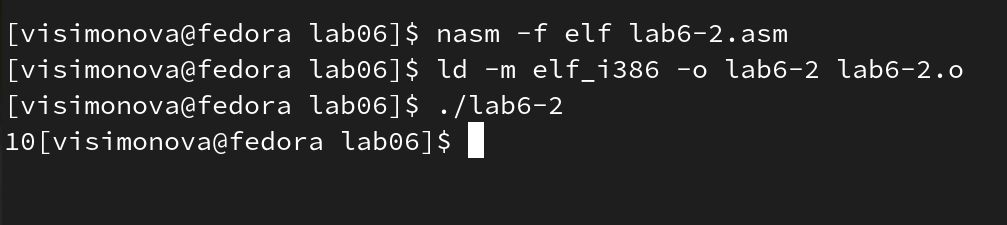


Figure 14: Компиляция и запуск файла.

##Выполнение арифметических операций в NASM

Создаю файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 (рис. [[15](#fig:015)]).

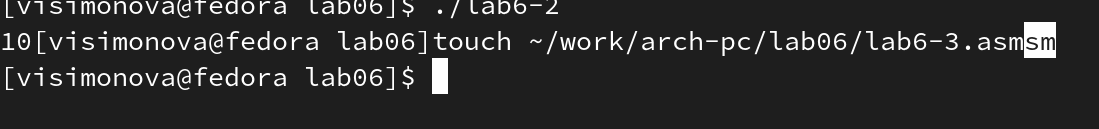


Figure 15: Создаю файл

Ввожу в файл lab6-3.asm программу вычисления выражения f(x) = (5 \* 2 + 3)/3 (рис. [[16](#fig:016)]).

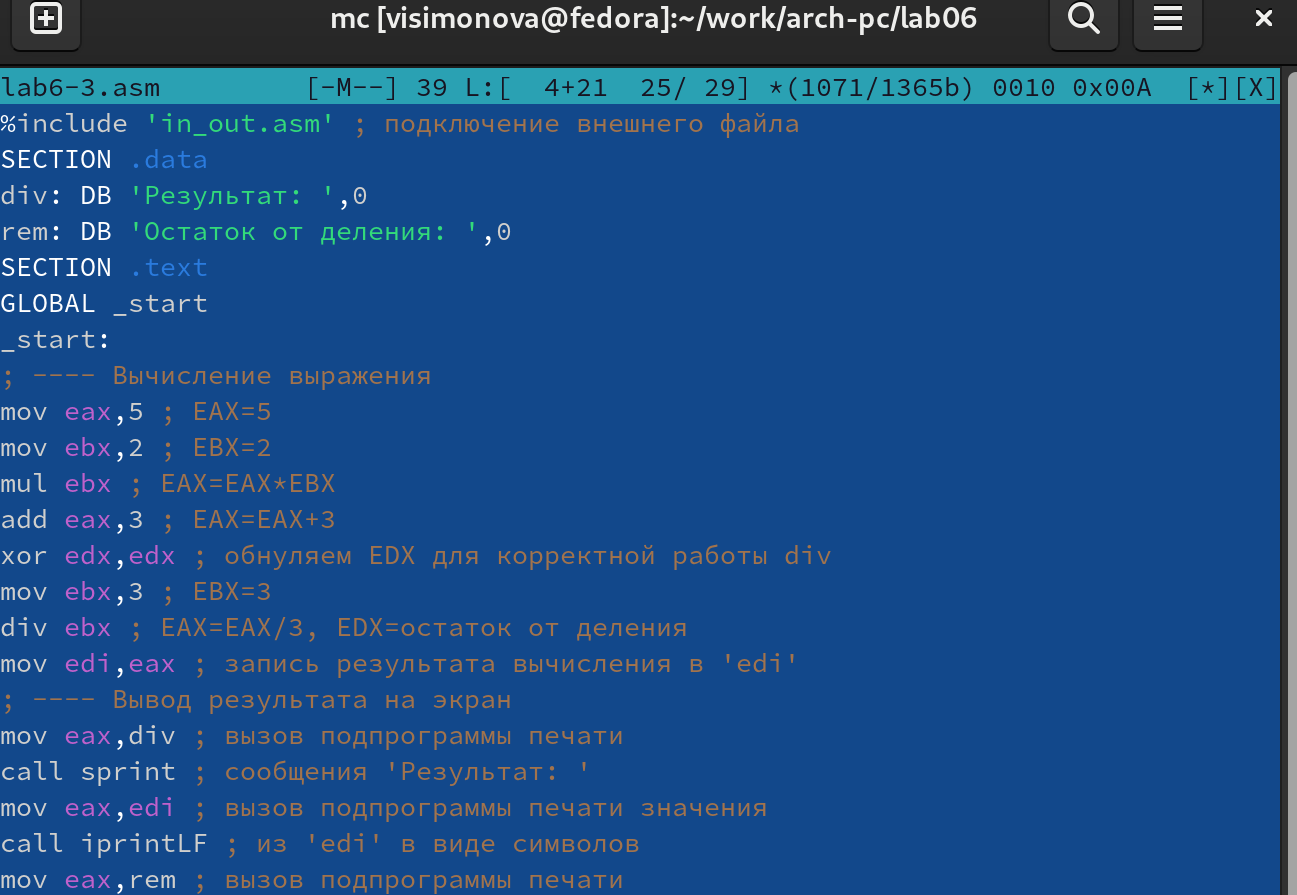


Figure 16: Ввожу в файл текст программы

Создаю и запускаю исполняемый файл lab6-3, программа вывела верный результат (рис. [[17](#fig:017)]).

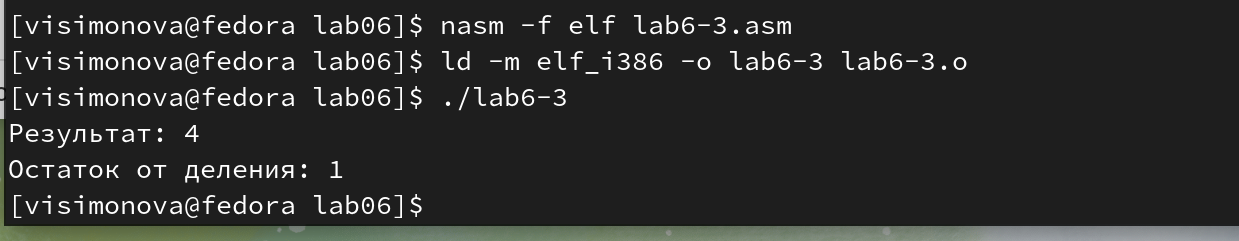


Figure 17: Запуск исполняемого файла lab6-3

Изменяю программу так, чтобы она высичляла значение для выражения f(x) = (4 \* 6 + 2)/5 (рис. [[18](#fig:018)]).

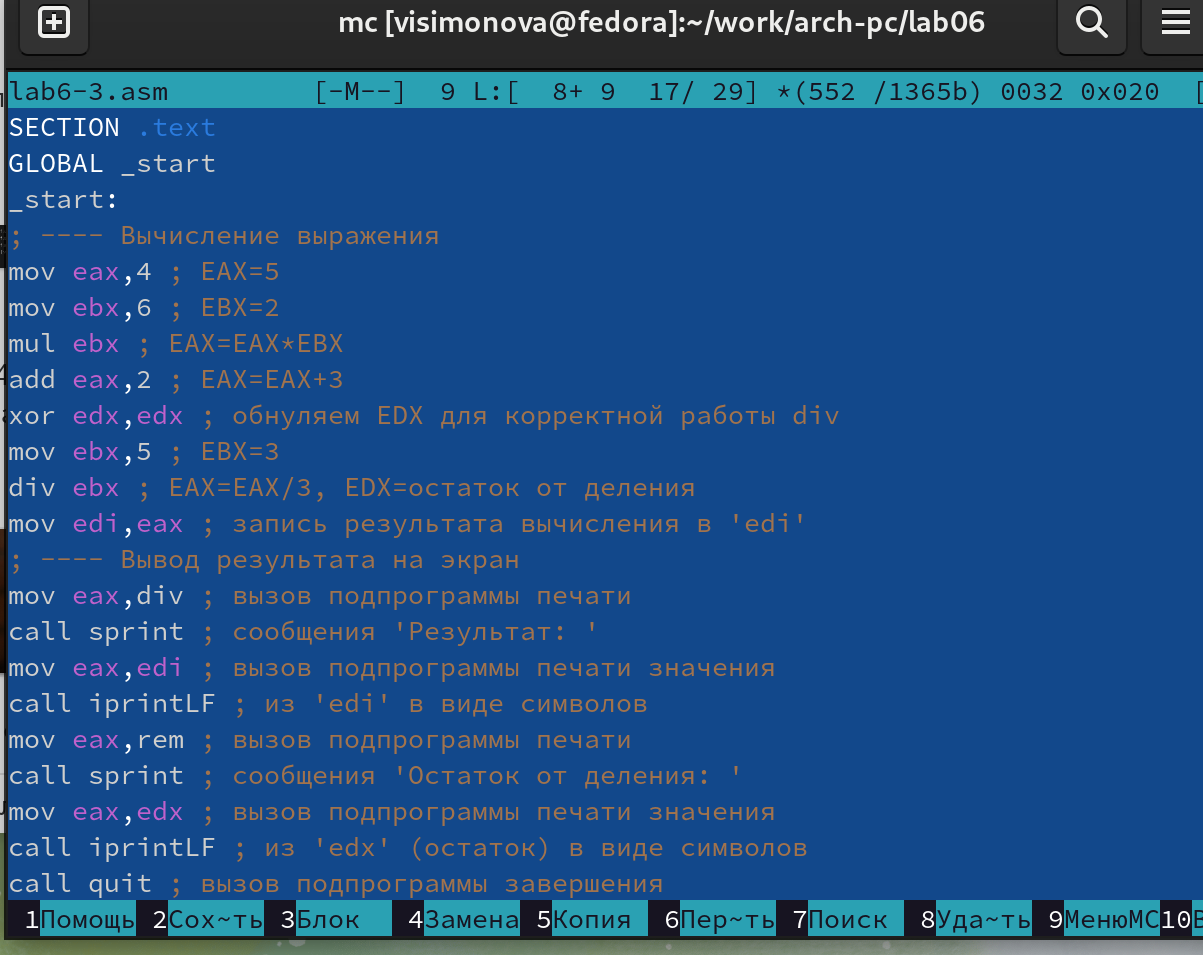


Figure 18: Изменяю файл lab6-3

Создаю и запускаю новый исполняемый файл lab6-3, программа вывела верный результат (рис. [[19](#fig:019)]).

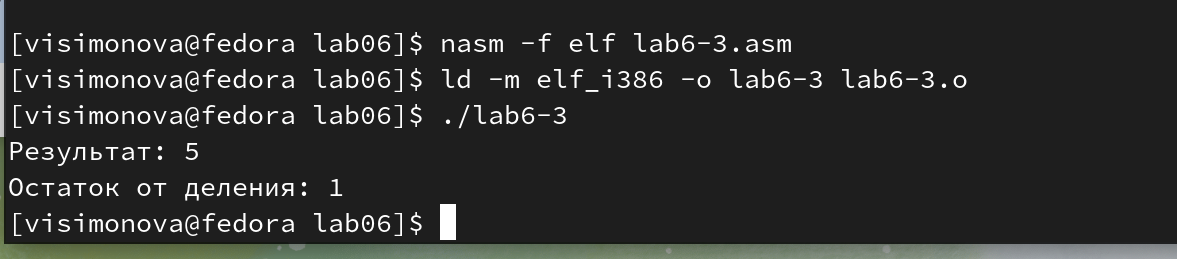


Figure 19: Запуск изменённого lab6-3

Создаю файл variant.asm в каталоге ~/work/arch-pc/lab06 и ввожу ы него текст для вычисления варианта задания по номеру студенческого билета (рис. [[20](#fig:020)]).

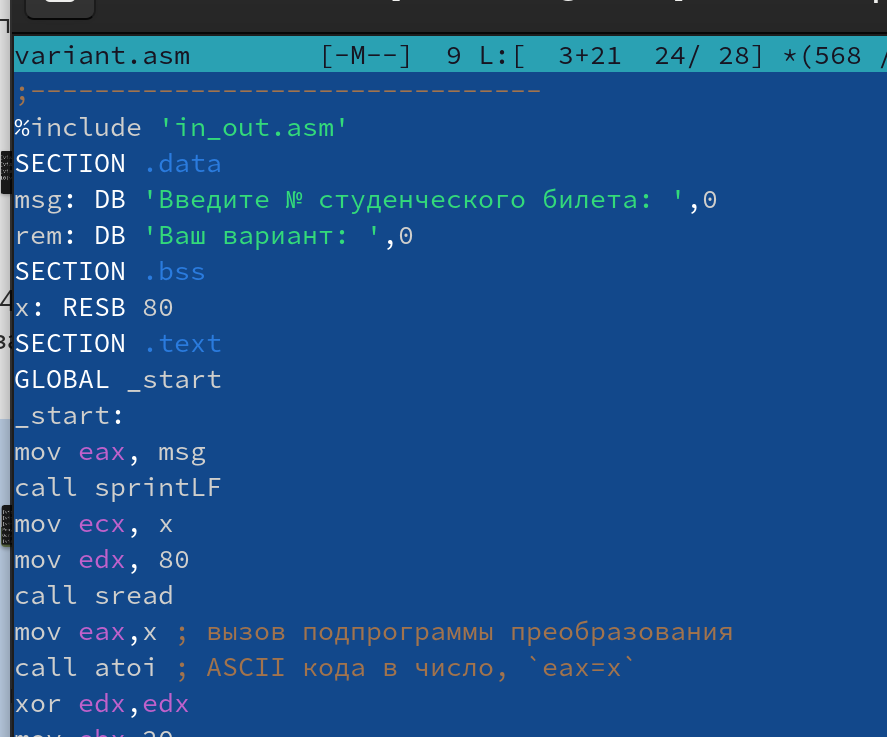


Figure 20: Ввожу в файл программу для вычисления номера задания

Создаю и запускаю исполняемый файл, ввожу номер свогего студенческого билета, результат программы-14 вариант (рис. [[21](#fig:021)]).

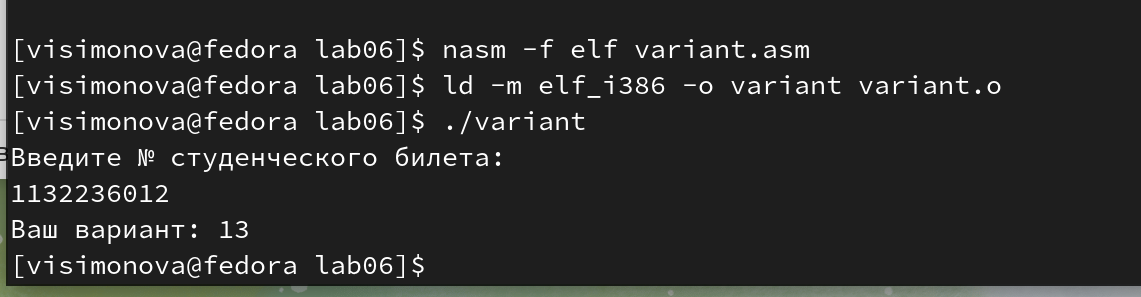


Figure 21: Запуск программы для вычисления варианта

###Ответы на вопросы

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

mov eax,rem  
call sprint

1. Для чего используется следующие инструкции?

mov ecx, x ;кладем адрес вводимой строки в регистр ecx  
mov edx, 80; запись длинны вводимой строки в регистр edx  
call sread ; ввод программы из внешнего файла , вводим сообщение с клавиатуры

1. Для чего используется инструкция “call atoi”?

Вызов программы из внешнего файла , которая отвечает за преобразование ASCII кода.

1. Какие строки листинга 6.4 отвечают за вычисления варианта?

xor edx,edx; обнуление регистра edx  
mov ebx,20; edx=20  
div ebx; edx=edx/20  
inc edx;edx=edx+1

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

В регистр edx

1. Для чего используется инструкция “inc edx”?

Это инкремент,команда inc ebx увеличивает значение регистра ebx на 1

1. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений?

mov eax,edx  
call iprintLF

## 4.1 Задание для самостоятельной работы

Создаю файл zadanie.asm и ввожу в него текст программы для вычисления выражения 13) f(x) = (8*x + 6)* 10 (рис. [[22](#fig:022)]).

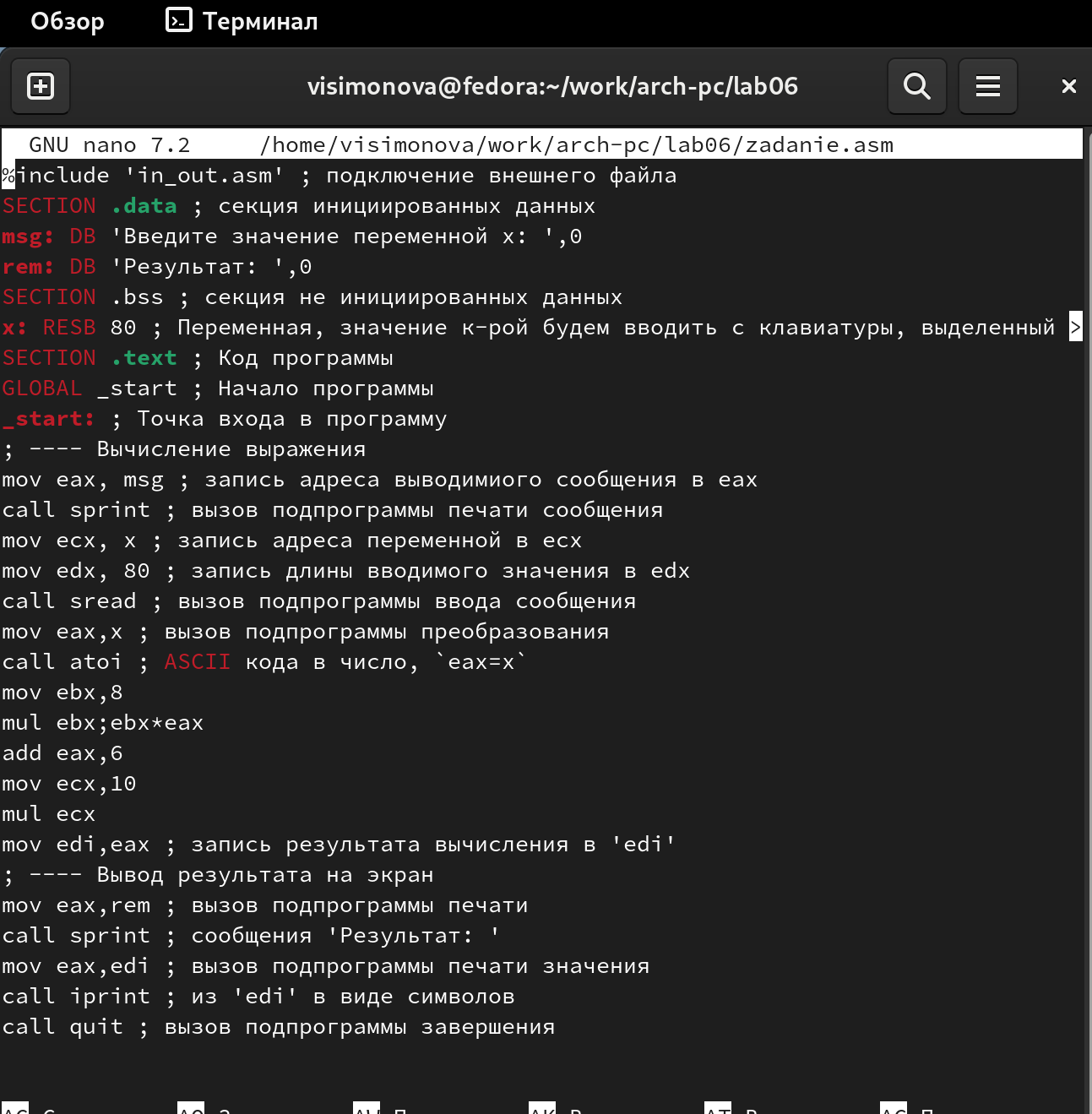


Figure 22: Мой код

Создаю и запускаю исполняемый файл, ввожу первое и второе значение переменной (рис. [[23](#fig:023)]).

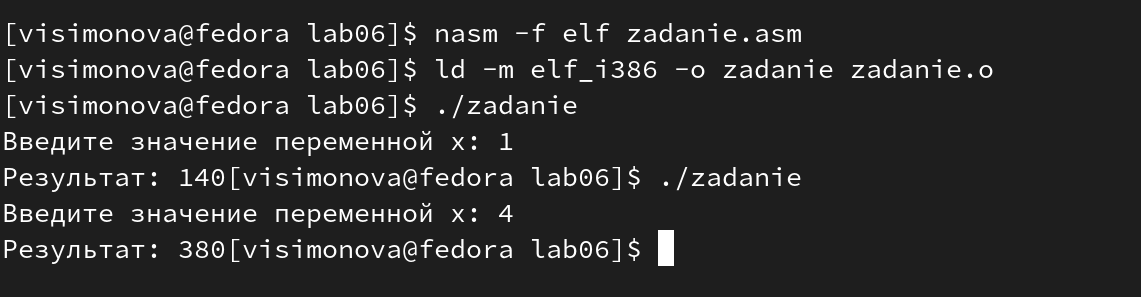


Figure 23: Запуск

# 5 Выводы

Я освоила арифметические инструкции языка ассемблера NASM.

# Список литературы

1. [Лабораторная работа №7](https://esystem.rudn.ru/pluginfile.php/1584637/mod_resource/content/1/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%967.pdf)
2. [Таблица ASCII](https://www.rapidtables.com/code/text/ascii-table.html)