Отчёт по лабораторной работе №6

Лобанова Екатерина Евгеньевна

Содержание

Список иллюстраций

Список таблиц

1 1. Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 2. Порядок выполнения лабораторной работы

2.1 2.1 Символьные и численные данныев NASM

Создаем каталог для программ лабораторной работы № 6 lab06, переходим в него и создаем файл lab6-1.asm (рис. 2.1).

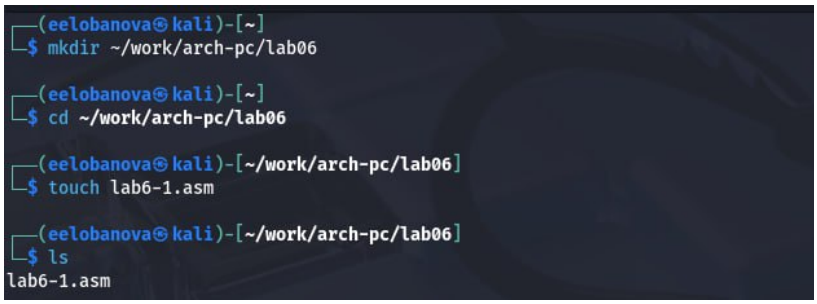

Рисунок 2.1: Создание каталога и файла для лабораторной работы 6

Рассмотрим примеры программ вывода символьных ичисленных значений.Программы будут выводить значения записанные в регистр eax.

Вводим в файл lab6-1.asm текст программы из листинга 6.1. (рис. 2.2).

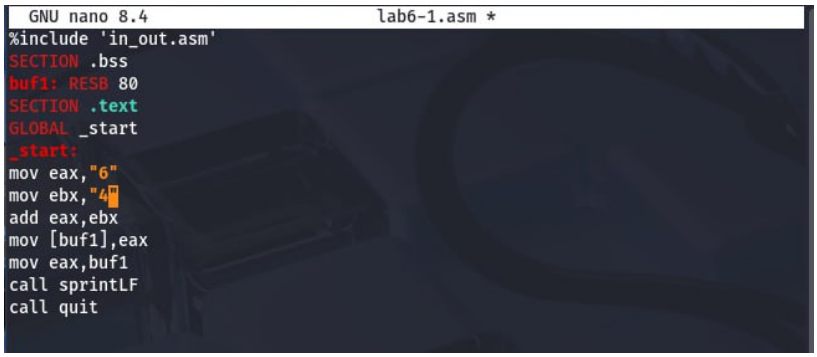

Рисунок 2.2: Используя команду nano вводим текст листинга в файл.

Копируем файл in_out.asm в новый каталог lab06. Создаем исполняемый файл и запускаем его, видим что результатом компеляции является символ j. (рис. 2.3).

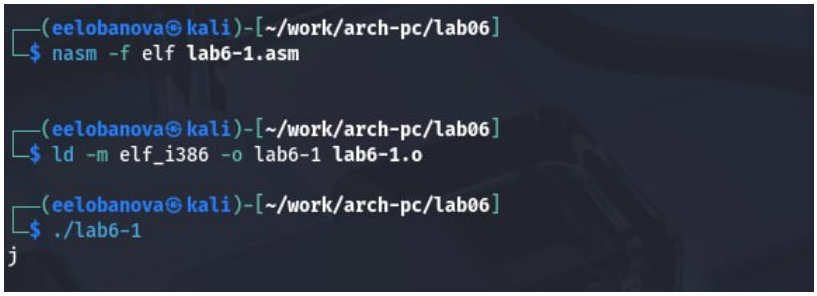

Рисунок 2.3: Компелируем файл и проверяем корректронсть его работы.

Меняем текст программы, убрав ковычки mov eax,„6“ mov ebx,„4“ в данных строках.(рис. 2.4).

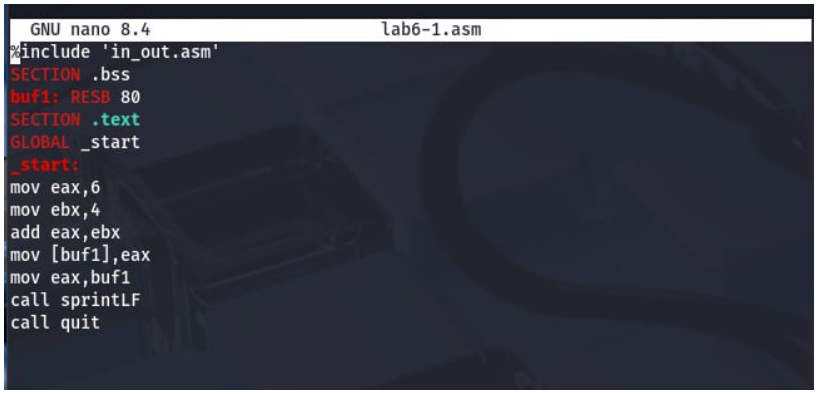

Рисунок 2.4: Меняем программу

Снова создаем исполняемый файл и запустите его, сморим на изменения. Выводтся пустое поле. (рис. 2.5).

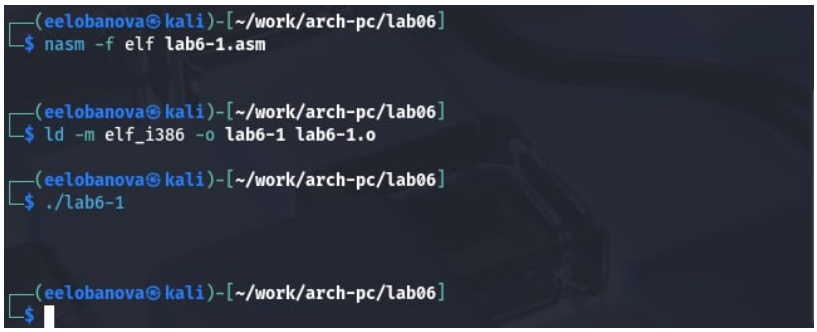

Рисунок 2.5: Скомпелировали файл и сморим на результат изменения программы

Пользуясь таблицей ASCII определили, что код 10 соотвествует символу перевода строки, поэтому он не отображается на экране.

Создаем файл lab6-2.asm в каталоге ~/work/arch-pc/lab06. (рис. 2.6).

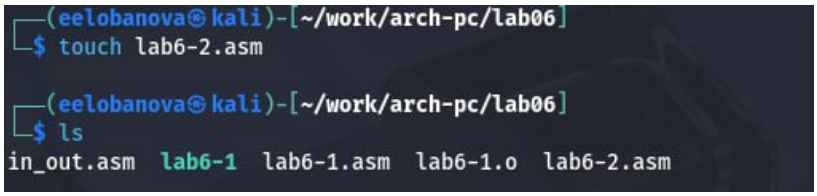

Рисунок 2.6: Создаем файл использую touch и проверяем что он есть

Вводим в файл lab6-2.asm текст листинга 2.(рис. 2.7).

Рисунок 2.7: Заполняем файл

Создаем исполняемый файл и запускаем его.(рис. 2.8).

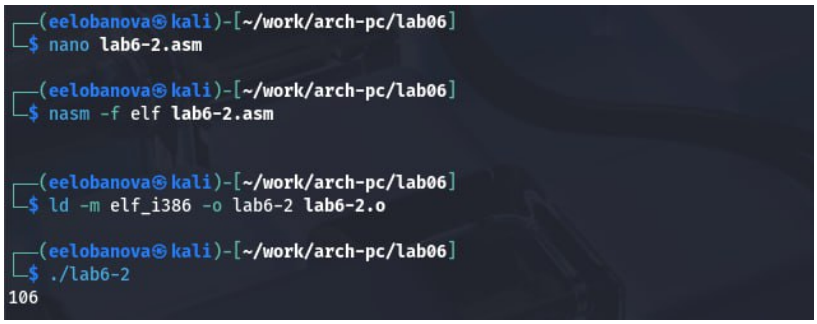

Рисунок 2.8: Запускаем файл и сморим на результат

Программы выводит число 106.

Каки в прошлом случаем убираем ковычки из двух строк меняя текст программы.(рис. 2.9).

Рисунок 2.9: Меняем программу

Создаем исполняемый файл и запускаем его.(рис. 2.10).

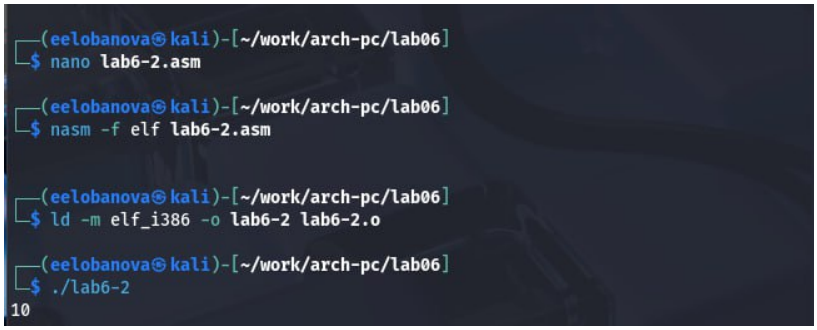

Рисунок 2.10: Запускаем файл

Результатом работы программы является число 10.

Заменяем подпрограмму iprintLF на iprint.(рис. 2.11).

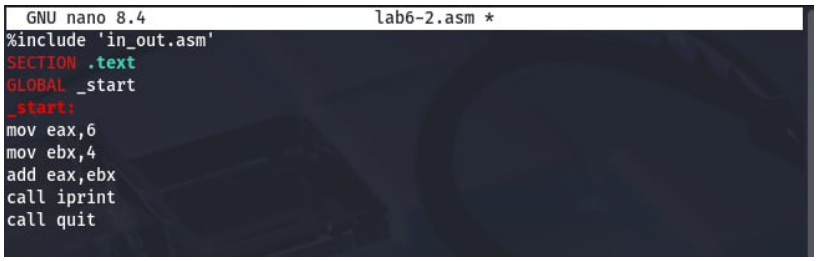

Рисунок 2.11: Замена подпрограммы

Создаем исполняемый файл и запускаем его. (рис. 2.12)

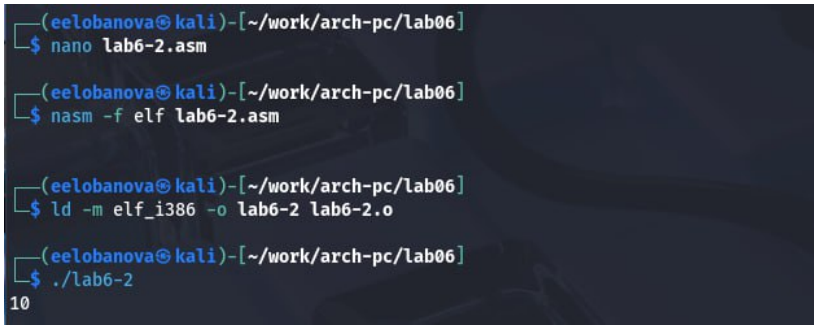

Рисунок 2.12: Запускаем файл и смотрим что изменилось

При использовании iprintLF выводимые строчки видны с новой сторки, а с iprint на той же самой.

2.2 2.1 Выполнение арифметических операций в NASM

Создаем файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 (рис. 2.13)

Рисунок 2.13: Используя touch создаем lab6-3.asm

Вводим в новый файл текст листинга 3.(рис. 2.14).

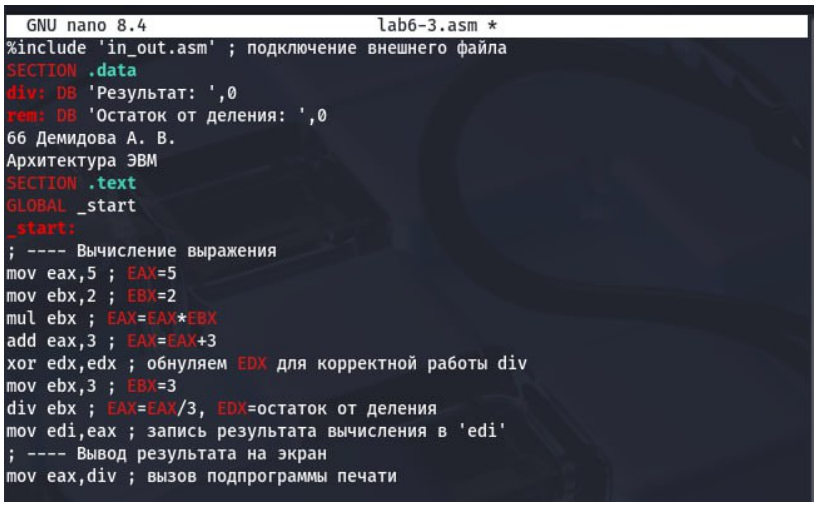

Рисунок 2.14: Заполняем файл

Создаем исполняемый файл и запускаем его. (рис. 2.15)

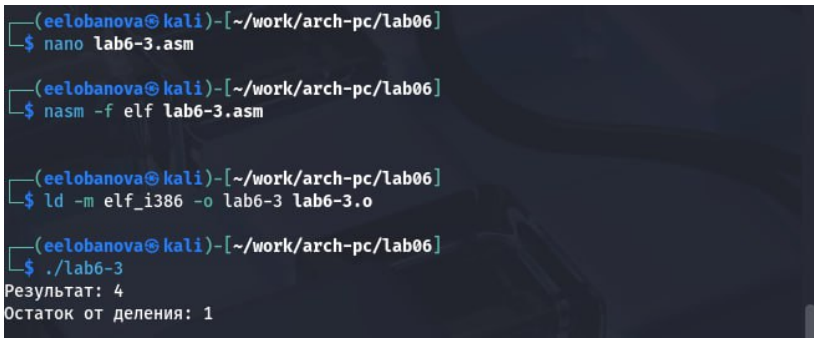

Рисунок 2.15: Рис 15

Программа работает корректно.

Вносим изменения в код, чтобы программв вычисляла выражение □(□) = (4 ∗ 6 + 2)/5. (рис. 2.15)

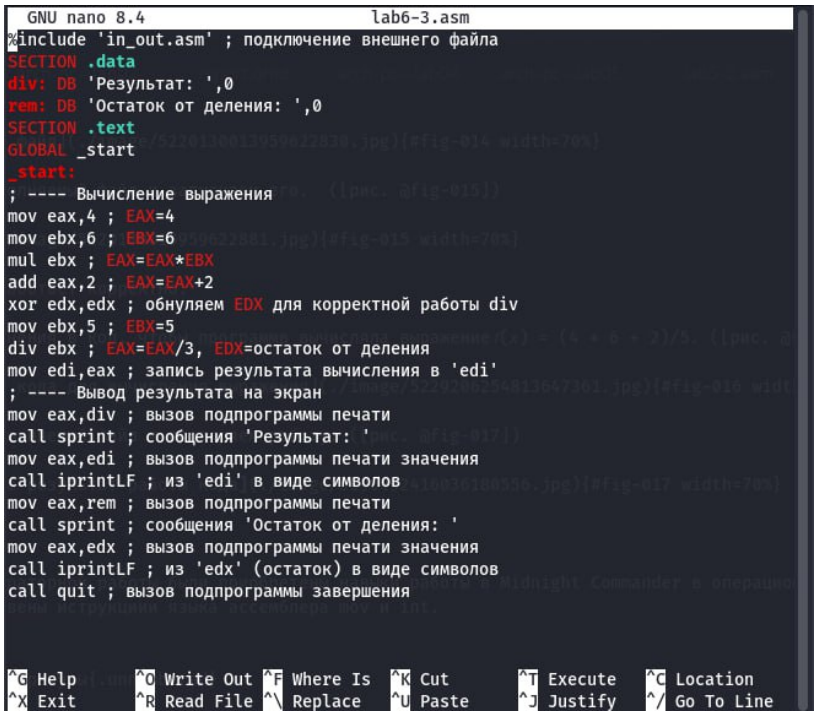

Рисунок 2.16: Изменение кода для вычисления выражения

Создаем исполняемый файл и проверяем работу. (рис. 2.17)

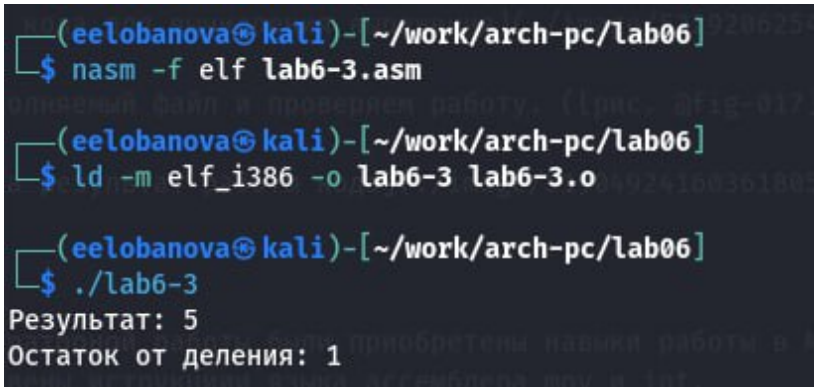

Рисунок 2.17: Смотрим на результат работы кода

Создаем файл variant.asm в каталоге ~/work/arch-pc/lab06 (рис. 2.18)

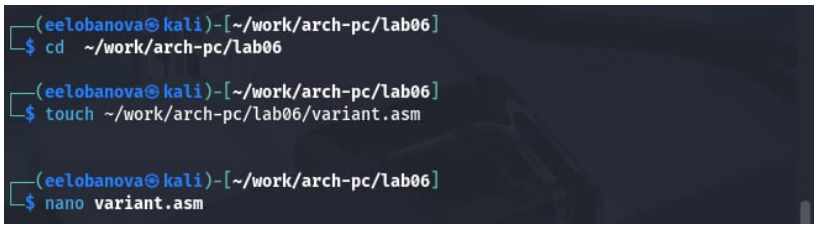

Рисунок 2.18: Создеам новый файл

Вводим текст листинга 6.4 в созданный файл.(рис. 2.19)

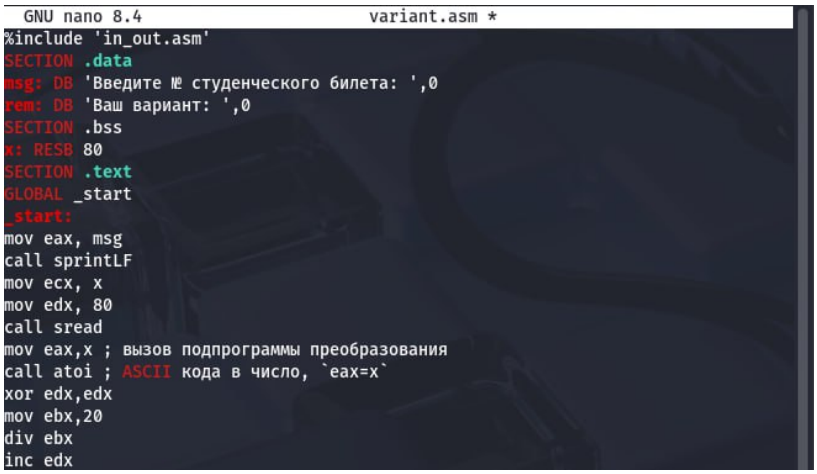

Рисунок 2.19: Редактируем файл

Компелируем файл и сморитм на результат его работы. (рис. 2.20)

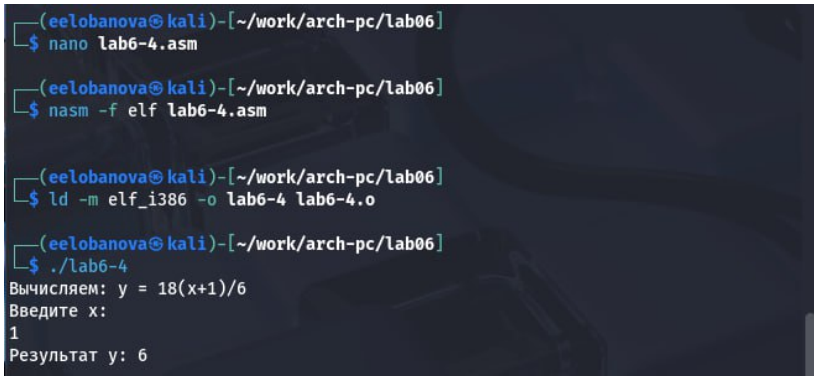

Рисунок 2.20: Проверяем корректность работы программы

Ответы на вопросы:

1. За вывод сообщения „Ваш вариант“ отвечают строки: mov eax,rem call sprint
2. Конструкции отвечают за чтение строки и ввод строки от пользователя. mov ecx, x - загрузка адреса буфера для сохранения вводимых данных

mov edx, 80 - установка максимального размера вводимых данных (80 символов)
call sread - вызов функции ввода строки из стандартного потока ввода

3. Инструкция call atoi используется для преобразования строки в число, принимая адрес строки в регистре eax и возвращает полученное число в eax.
4. За вычисение выриантa отвечают следующие строки: xor edx,edx ; Обнуление edx для деления mov ebx,20 ; Загрузка делителя (20) в ebx div ebx ; Деление (eax) / (ebx). Частное в eax, остаток в edx. inc edx ; Увеличение остатка на 1
5. При выполнении инструкции div ebx остаток от деления записывается в регистр edx
6. Инструкция inc edx увеличивает значение в регистре edx на 1.
7. Строка „mov eax,edx“ передает значение остатка от деления в eax. Строка „call iprintLF“ вызывает процедуру iprintLF для вывода значения на экран вместе с переводом строки.

3 Задание для самостоятельной работы

Создаем новый файл lab6-4.asm использую touch.(рис. 3.1)

Рисунок 3.1: Создаем файл

Редактируем его и вводим программу которая будет вычислять заданное выражение.(рис. 3.2)

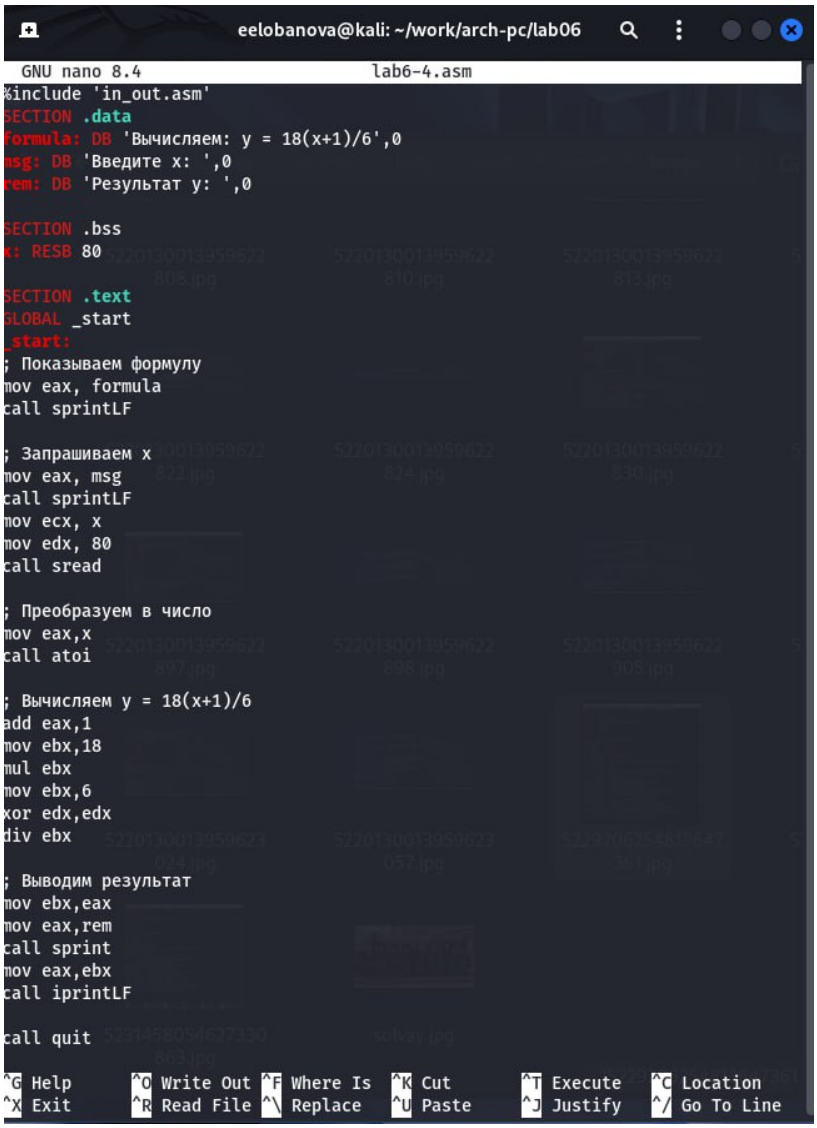

Рисунок 3.2: Заполняем файл

Запускаем программу и проверяем ее работу для x=3. (рис. 3.3)

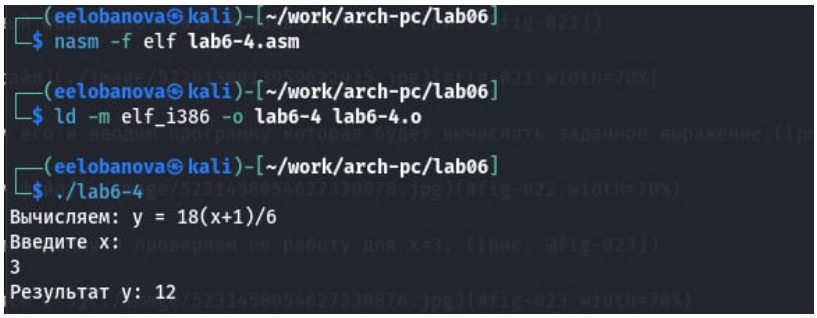

Рисунок 3.3: Подставляем x=3

Запускаем программу и проверяем ее работу для x=1. (рис. 3.4)

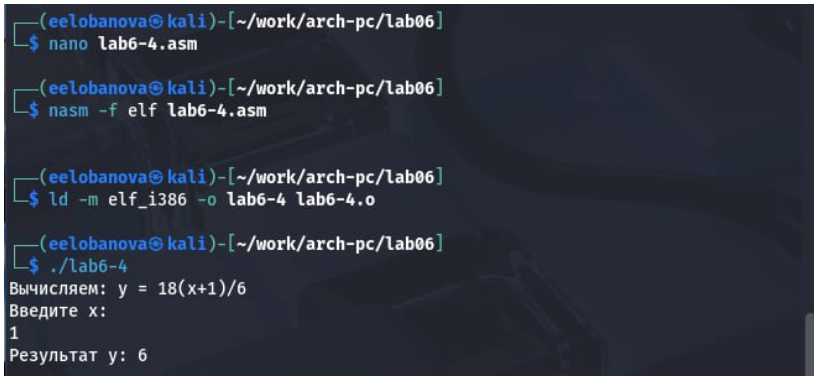

Рисунок 3.4: Подставляем x=1

4 Выводы

В ходе лабораторной работы были освоены арифметические инструкции NASM и их практическое применение. Изучены операции сложения, вычитания, умножения и деления с учетом особенностей работы с регистрами. Освоено преобразование данных между символьным и числовым представлением. Разработаны программы вычисления арифметических выражений и определения варианта задания.