Отчёт по лабораторной работе №6

Дисциплина: Архитектура компьютера

Машков Илья Евгеньевич

Содержание

# 1 Цель работы

Цель данной лабораторной работы - освоение арифметческих инструкций языка ассемблера NASM.

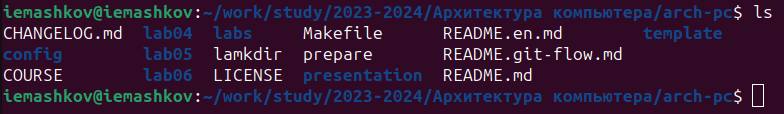
# 2 Задание

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

# 3 Выполнение лабораторной работы

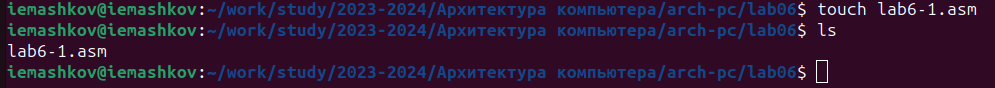
## 3.1 Символьные и численные данные в NASM

Ещё в прошлой лабораторной работе я по ошибке создал папку **lab06** (рис. [??]).



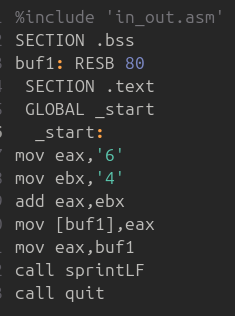
Директория lab06.

Далее я перехожу в эту директорию и создаю файл **lab6-1.asm** с помощью команды **‘touch’** (Рис. [??]).



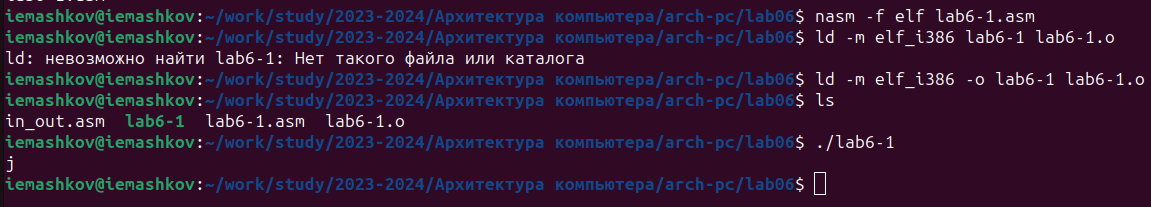
Создание рабочего файла.

Затем я ввожу код в .asm файл (Рис. [??]).



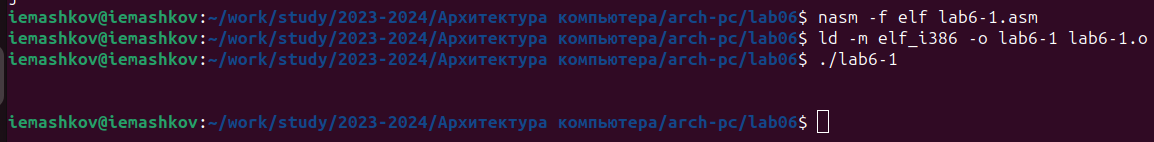
Код программы.

Создаю исполняемый файл и запускаю программу (Рис. [??]). В выводе программы я получаю символ **‘j’**, который по системе ASCII соответствует сумме двоичных кодов символов **6** и **4**.



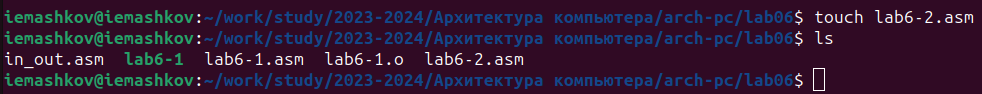
Результат выполнения программы.

Теперь я убираю кавычки у символов **6** и **4**, создаю исполняемый файл и запускаю программу (Рис. [??]). В выводе я получаю неотображающийся символ с кодом **10** - это **символ перевода строки**.



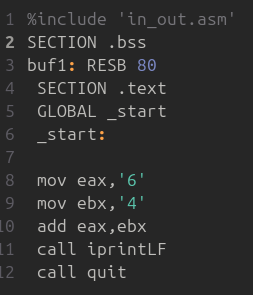
Результат второго выполнения программы.

Создаю файл **lab6-2.asm** (Рис. [??]).



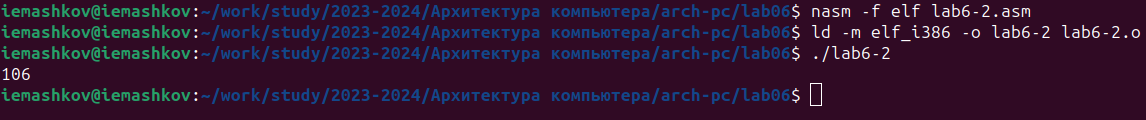
Создание файла lab6-2.asm.

Ввожу код программы (Рис. [??]).



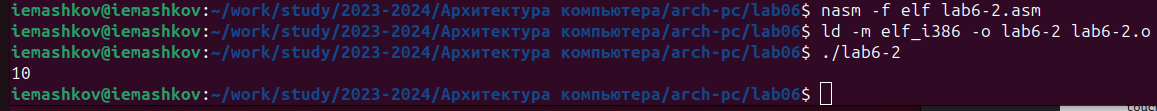
Код программы.

Создаю исполняемый файл и запускаю программу (Рис. [??]). В выводе получаю число **106**, т.к. программа позволяет вывести число, а не символ, хоть и по-прежнему происходит сложение кодов символов **6** и **4**.



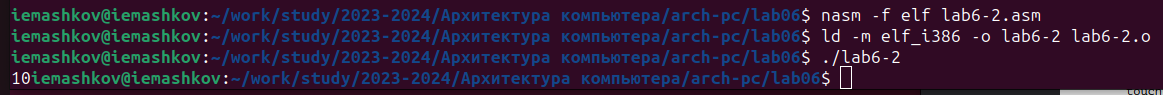
Результат выполнения программы.

Теперь убираю кавычки и заново создаю исполняемый файл и запускаю программу и в выводе получаю **10**, т.к. в этом случае программа складывает сами числа, а не их коды (Рис. [??]).



Результат второго выполнения программы.

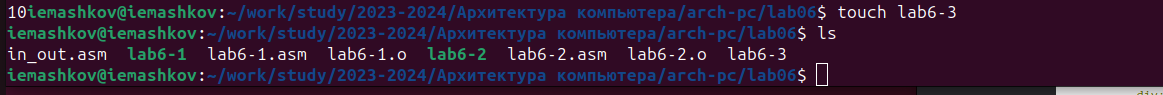
Затем меняю параметр **‘iprintLF’** на **‘iprint’**, создаю исполняемый файл и запускаю программу (Рис. [??]). В выводе получаю то же число, но без символа переноса строки.



Результат третьего выполнения программы.

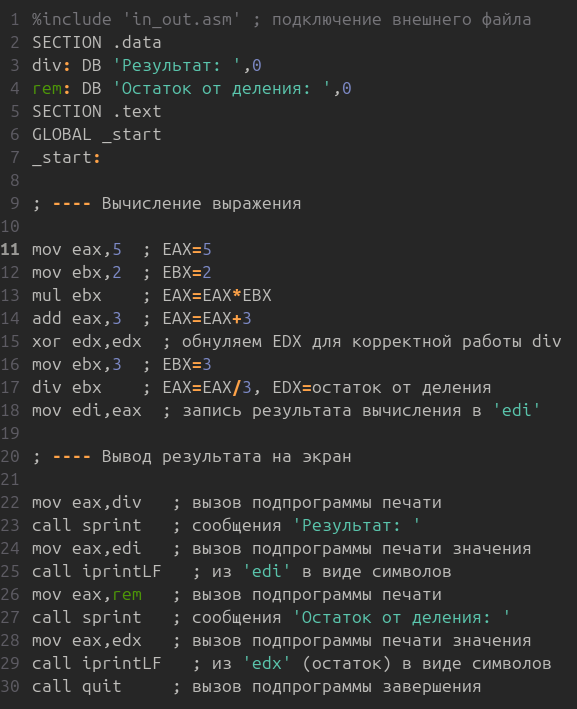
## 3.2 Выполнение арифметических операций в NASM

Создаю файл **lab6-3.asm** (при создании я забыл дать ему расширение .asm, но потом переименовал его) (Рис. [??]).



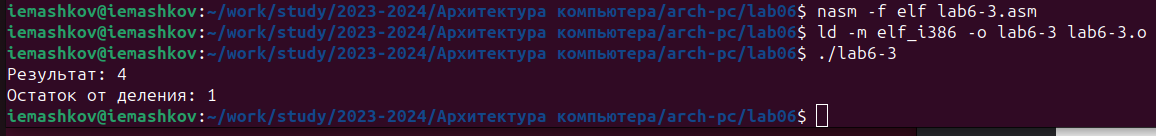
Создание файла lab6-3.asm.

Теперь ввожу в него код программы, который будет вычислять выражение f(x) = (5\*2 + 3)/3 (Рис. [??]).



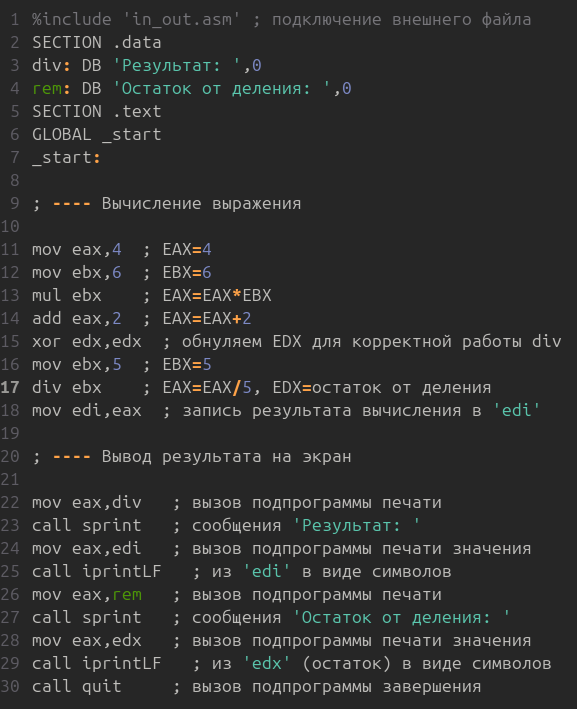
Код программы.

Далее создаю исполняемый файл и запускаю программу (Рис. [??]).



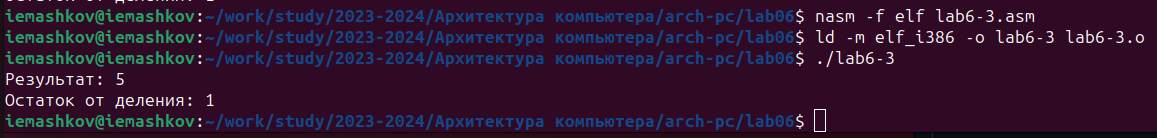
Результат выполнения программы.

Изменяю код программы так, чтобы она вычисляла выражение f(x) = (4\*6 + 2)/5 (Рис. [??]).



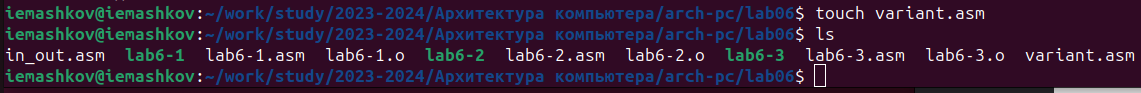
Изменённый код программы.

Создаю исполняемый файл и запускаю программу (Рис. [??]).



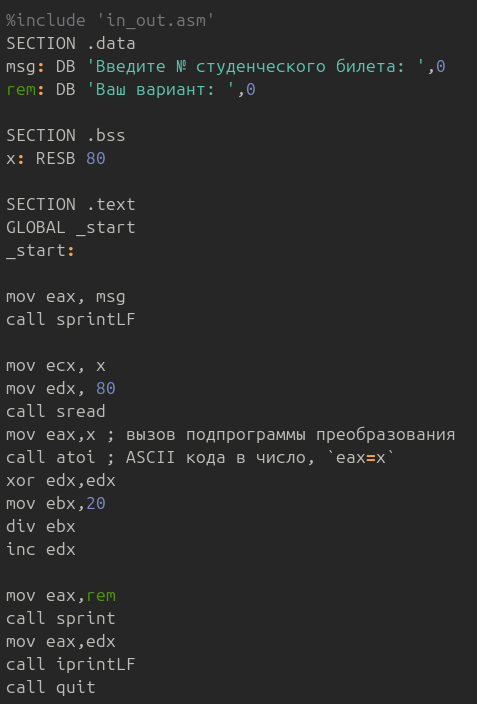
Результат второго выполнения программы.

Создаю файл **variant.asm** с помощью команды **‘touch’** (Рис. [??]).



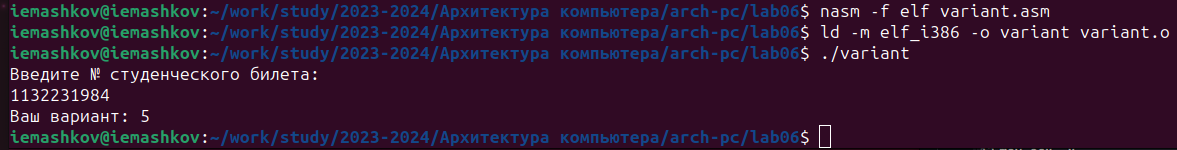
Создание файла variant.asm.

Теперь ввожу в него код программы, которая по номеру студенческого должна мне выдать номер моего варианта (Рис. [??]).



Результат второго выполнения программы.

Создаю исполняемый файл и запускаю его (Рис. [??]). Мой вариант - **5**.



Код программы.

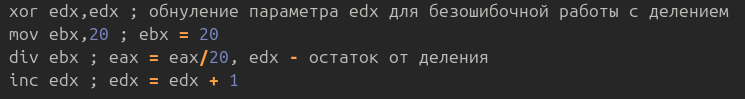
# 4 Ответы на вопросы по программе

1. За вывод сообщения **“Ваш вариант:”** отвечают эти строки кода (Рис. [??]):

Строки кода, выводящие сообщение о моём варианте.

Строки кода, выводящие сообщение о моём варианте.

1. **‘mov ecx, x’** используется для того, чтобы внести адрес вводимой строки **‘x’** в регистр **‘ecx’**. **‘mov edx, 80’** - запись в регистр **‘edx’** длинны вводимой строки. **‘call sread’** - вызов подпрограммы из внешнего файла, которая обеспечивает ввод сообщения с клавиатуры.
2. **‘call atoi’** - подпрограмма из внешнего файла, преобразующая код символа в целое число и записывающая его в регистр **‘eax’**.
3. За вычисление варианта отвечают строки (Рис. [??]):



Строки кода, вычисляющие вариант.

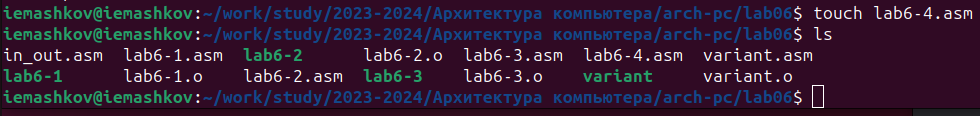
1. При выполнении инструкции **‘div ebx’** остаток от деления записывается в регистр **‘edx’**.
2. Инструкция **‘inc edx’** увеличивает значение регистра **‘edx’** на **1**
3. За вывод на экран результатов вычислений отвечают строки (Рис. [??]):

Строки кода, выводящие результат вычислений.

Строки кода, выводящие результат вычислений.

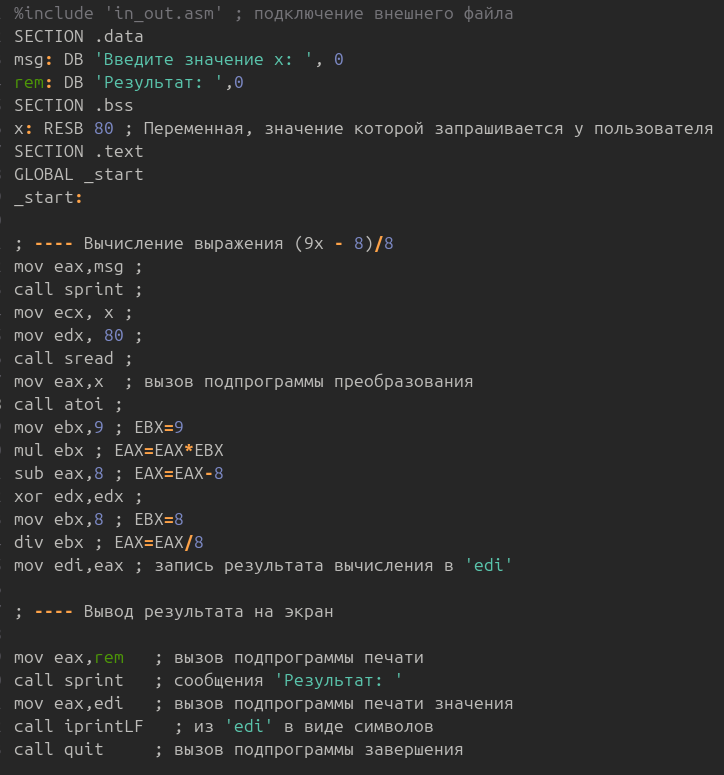
## 4.1 Выполнение заданий для самостоятельной работы

1. Создаю файл **lab6-4.asm** (Рис. [??]).



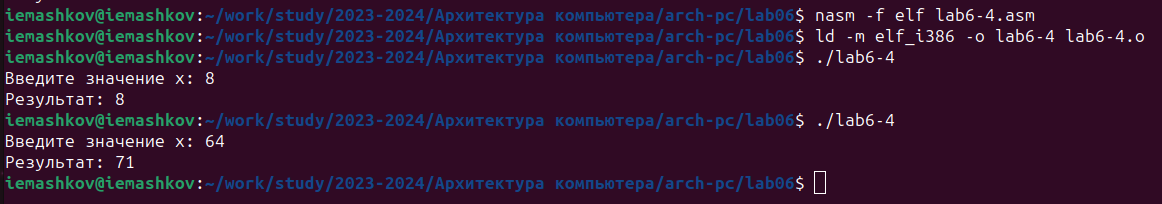
Создание файла lab6-4.asm.

Ввожу код программы, которая будет вычислять выражение **‘f(x) = (9x - 8)/8’**. Именно это выражение стоит под номером **5** в таблице (Рис. [??]).



Получение и запуск исполняемого файла.

Создаю исполняемый файл и запускаю его со значением **‘8’**, а потом и со значением **‘64’** (Рис. [??]).



Результаты выполненй программы.

Эти результаты я проверил, а это значит, что программа отработала верно. # Выводы

При выполнении данной лабораторной работы я освоил арифметические инструкции языка ассемблера NASM.

# 5 Список литературы

[Архитектура ЭВМ](https://esystem.rudn.ru/pluginfile.php/2089086/mod_resource/content/0/%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%966.%20%D0%90%D1%80%D0%B8%D1%84%D0%BC%D0%B5%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5%20%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%B8%20%D0%B2%20NASM..pdf)

[Таблица кодов ASCII](https://www.rapidtables.com/code/text/ascii-table.html)