Отчёт по лабораторной работе №6

Дисциплина: Архитектура компьютеров

Лазарев Даниил Михайлович

Содержание

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

# 2 Задание

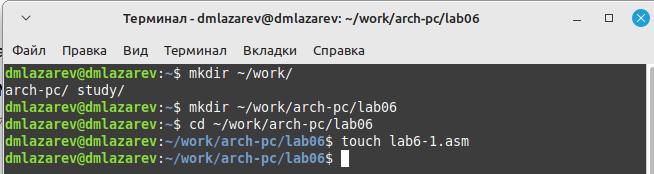
1. Выполнение лабораторной работы №6
2. Заполнение отчета по выполнению лабораторной работы №6 с помощью языка разметки Markdown
3. Выполнение заданий для самостоятельной работы

# 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.

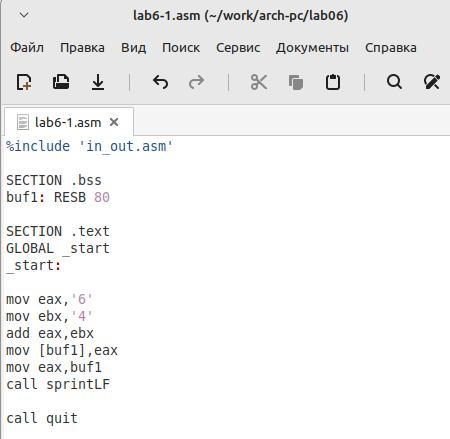
# 4 Выполнение лабораторной работы

Создадим каталог для программ лаб. работы н.6, перейдем в него и создадим файл “lab6-1.asm” (рис. ??)



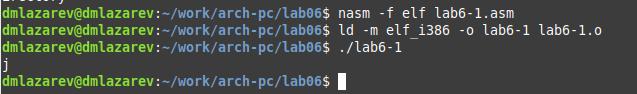
Создание файла в каталоге

Введем в созданный файл текст программы из предложенного нам листинга 6.1(рис. ??)



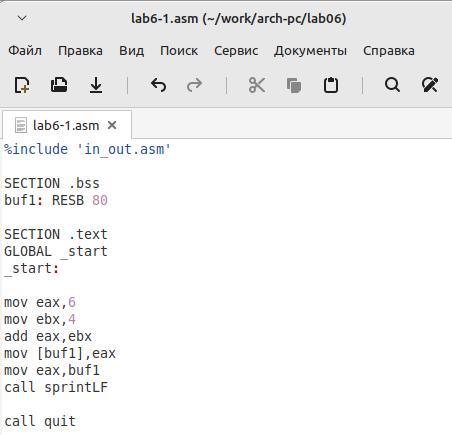
Код программы в файле

Создадим исполняемый файл и запустим его, предварительно скопировав из предыдущей лаб. работы файл “in\_out.asm” для корректной работы (рис. ??)



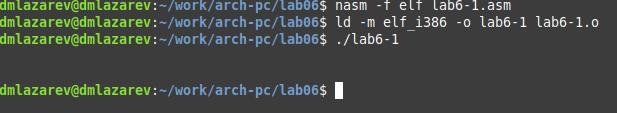
Преобразование в исполняемый файл

Далее изменим часть строк в исходном файле. (рис. ??)



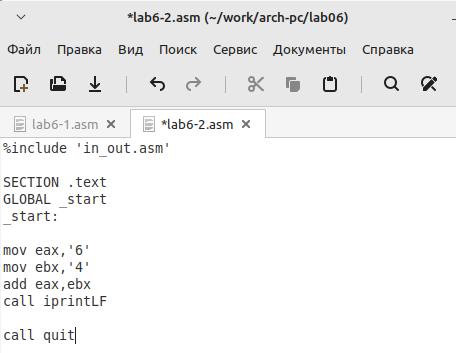
Измененный код программы

Преобразуем в исполняемый файо и проверим правильность выполнения. (рис. ??)



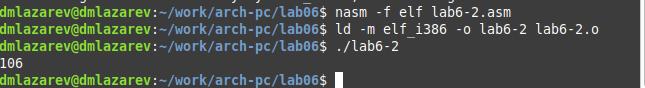
Преобразование измененного файла

Предварительно создав файл “lab6-2.asm” в нашем каталоге, вставим в него предложенный нам листинг 6.2 (рис. ??)



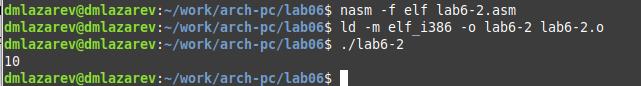
Проверка наличия текста

Преобразуем файл в исполняемый и проверим правильность выполнения. (рис. ??)



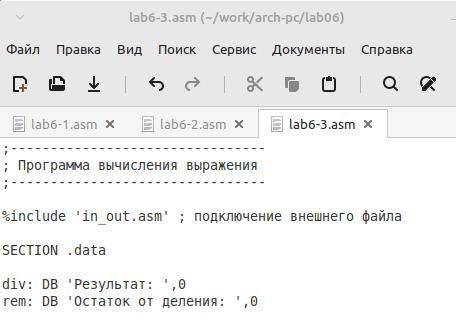
Преобразование файла

Изменим часть строк в коде программы, преобразуем в исполняемый файл и проверим правильность выполнения. (рис. ??)



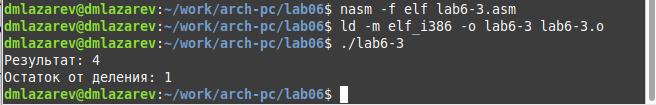
Преобразование исправленного файла

Создадим файл “lab6-3.asm” в каталоге и вставим в него предложенный нам листинг 6.3 (рис. ??)



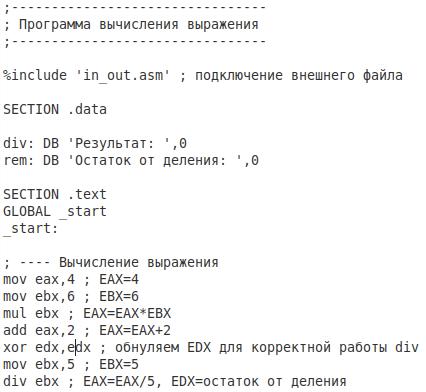
Создание файла “lab6-3.asm”

Преобразуем его в исполняемый файл и проверим правильность выполнения (рис. ??)



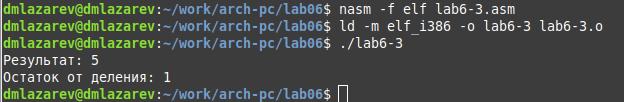
Проверка правильности работы файла

Исправим листинг 6.3 так, чтобы выполнялась функция (4\*6+2)/5 (рис. ??)



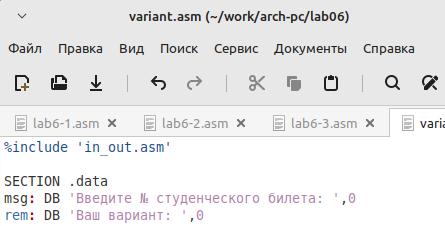
Исправленный файл

Преобразуем файл в исполняемый и проверим правильность работы (рис. ??)



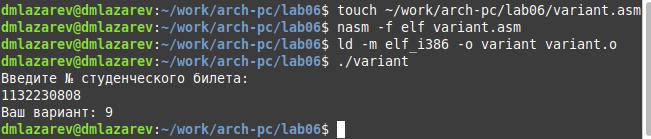
Преобразование файла в исполняемый

Создадим файл “variant.asm” и вставим в него предложенный нам листинг 6.4 (рис. ??)



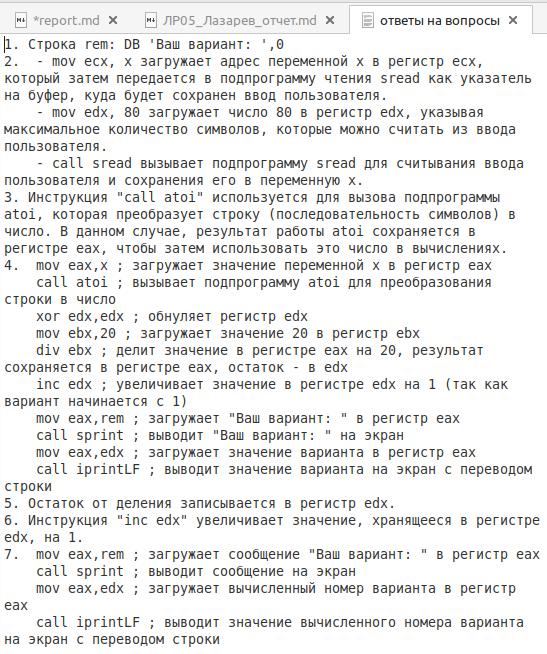
Код программы в файле

Преобразуем созданный нами файл в исполняемый и проверим работоспособность (рис. ??)



Преобразование файла и проверка работы

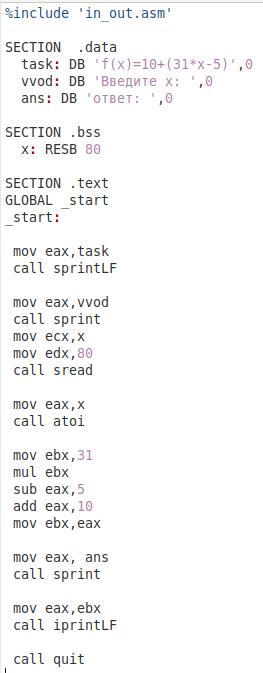
Ответим на вопросы, поставленные посли листинга 6.4 (рис. ??)



Ответы на вопросы

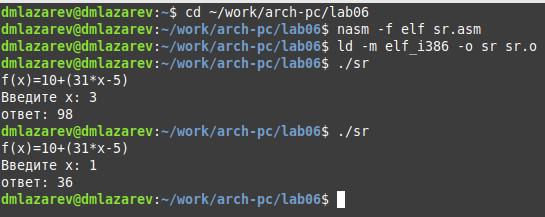
# 5 Выполнение самостоятельной работы

Основываясь на результате файла “variant.asm” выберем функцию из таблицы, в нашем случае это - 10+(31\*х-5). Самостоятельно напишем код программы, который будет проводить вычисления относительно введенного х. (рис. ??)



Код программы для вычисления функции

Преобразуем файл в исполняемый и проверим правильность выполнения работы относительно заданных нам х (3 и 1) (рис. ??)



Проверка работоспособности

# 6 Выводы

В ходе лабораторной работы мы освоили арифметические операции на языке NASM.