# РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

## *Факультет физико-математических и естественных наук*

## направление: *Компьютерные и информационные науки*

## Лабораторная работа №6

### дисциплина: *Архитектура компьютеров и операционные системы*

#### студент: Гробман Александр Евгеньевич

#### Группа: НКАбд-02-23

### Цель работы

Освоение арифметических инструкций языка ассемблера NASM

### Задание

1. Символьные и численные данные в NASM.
2. Выполнение арифметических операций в NASM.
3. Ответы на вопросы по листингу 6.4
4. Задания для самостоятельной работы

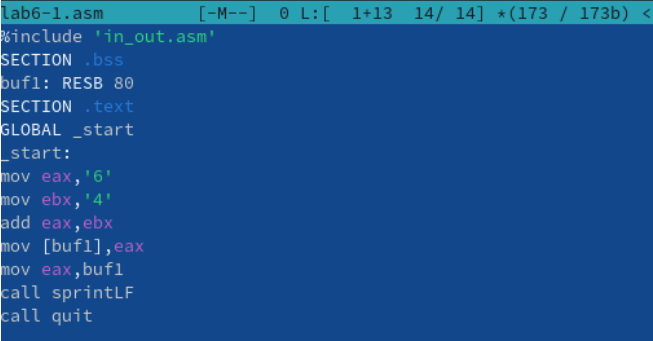
### Теория

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. 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. Схема команды целочисленного сложения add (от англ. addition - добавление) выполняет сложение двух операндов и записывает результат по адресу первого операнда. Допустимые сочетания операндов для команды add аналогичны сочетаниям операндов для команды mov. Так, например, команда add eax,ebx прибавит значение из регистра eax к значению из регистра ebx и запишет результат в регистр eax. Довольно часто при написании программ встречается операция прибавления или вычитания единицы. Прибавление единицы называется инкрементом, а вычитание — декрементом. Для этих операций существуют специальные команды: inc (от англ. increment) и dec (от англ. decrement), которые увеличивают и уменьшают на 1 свой операнд. Команда neg рассматривает свой операнд как число со знаком и меняет знак операнда на противоположный. Операндом может быть регистр или ячейка памяти любого размера. Для деления, как и для умножения, существует 2 команды div (от англ. divide - деление) и idiv. Например, в табл. 3.1 приведено краткое описание стандартных каталогов Unix.

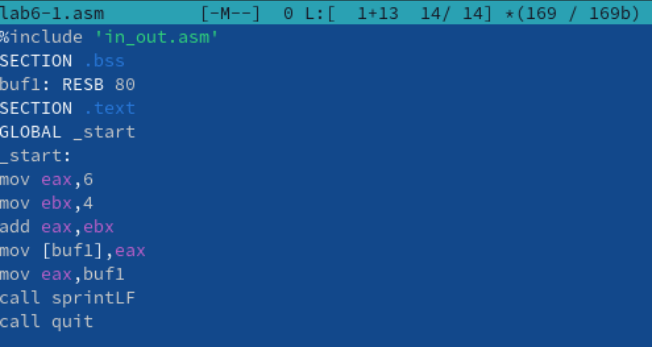
### Выполнение лабораторной работы

#### Символьные и численные данные в NASM

создаю файл lab6-1.asm и ввожу туда код из листинга

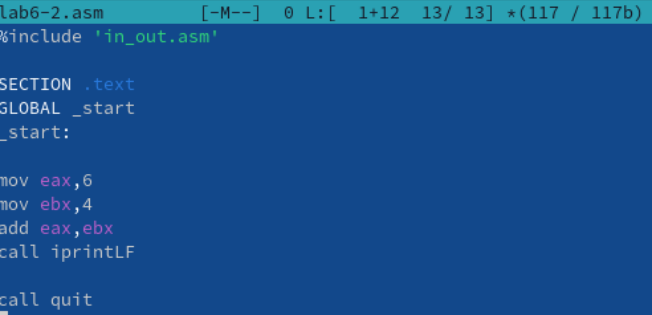


im 0

Изменяем текст программы по образцу 

по сути, программа выполняет переход на следущую строку

Создаю файл lab6-2.asm в каталоге и ввожу туда код из листинга



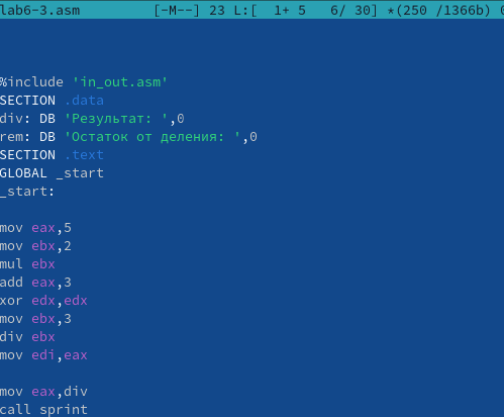
im2

im3

im3

#### Выполнение арифметических операций в NASM.

Создаю файл lab6-3.asm и ввожу в него текст из листинга

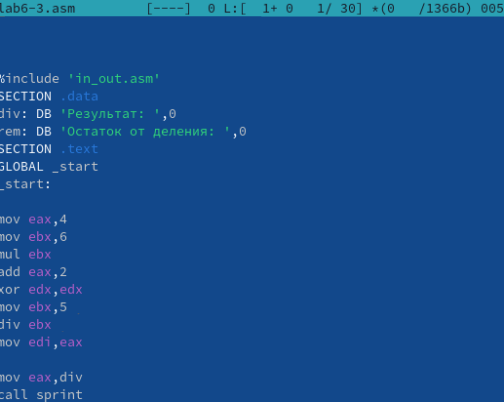


im 4

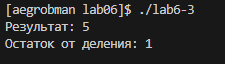
im 5

im 5

Изменяю текст программы для вычисления выражения f(x) = (4 \* 6 + 2)/5,

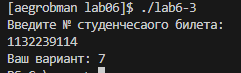


im 6



im 7

Текст программы из листинга 6.4 ввожу в файл variant.asm, создаю исполняемый файл и запускаю его. Проверяю результат работы программы, вычислив номер варианта



im8

#### Ответы на вопросы по листингу 6.4

За вывод сообщения “Ваш вариант” отвечают строки кода: mov eax,rem call sprint

mov ecx, x - Используется, чтобы положить адрес вводимой строки x в регистр. mov edx, 80 - Используется для записи в регистр edx длины вводимой строки. call sread - Используется для вызова подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры.

“call atoi” используется для вызова подпрограммы из внешнего файла, которая преобразует ascii-код символа в целое число и записывает результат в регистр eax.

За вычисления варианта отвечают строки: xor edx,edx mov ebx,20 div ebx inc edx

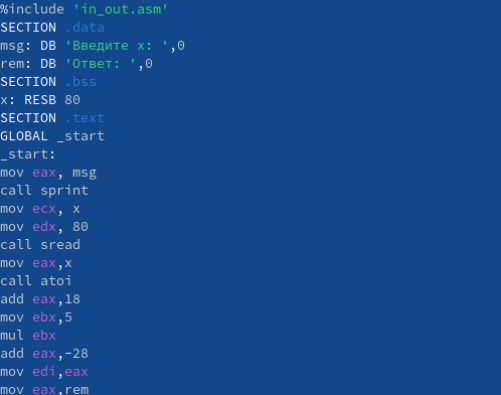
При выполнении инструкции div ebx остаток от деления записывается в регистр edx.

Инструкция “inc edx” увеличивает значение регистра edx на 1.

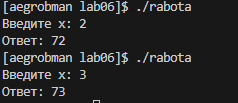
За вывод на экран результатов вычислений отвечают строки: mov eax,edx

call iprintLF

#### Задания для самостоятельной работы



im 9



im 10

#### Вывод

С помощью данной лабораторной работы я освоила арифметические инструкции языка ассемблер NASM, что пригодится мне при выполнении последующих лабораторных работ.

### Отправляем файлы на гитхаб.

Ссылка на отчёт <https://github.com/DaOneme/AEGrobman_study_2023-2024_arhpc/tree/main/Labs/Lab06>