Отчет по лабораторной работе №6

Освоение арифметических инструкций языка ассемблера NASM

Карапетян Мари Рафаеловна

Содержание

# 1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

# 2 Задание

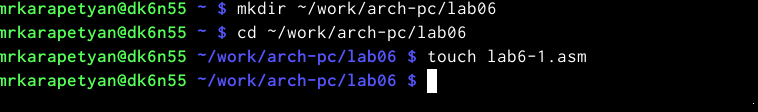
1.Написать программу вычисления выражения 𝑦 = 𝑓(𝑥). Вид функции 𝑓(𝑥) выбрать из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. 2.Загрузите файлы на GitHub.

# 3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес опе- ранда 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 ко- манде, Например: mov ax,2. • Адресация памяти – операнд задает адрес в памяти. В команде указывается символи- ческое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию. Например, определим переменную intg DD 3 – это означает, что задается область памяти размером 4 байта, адрес которой обозначен меткой intg. В таком случае, команда mov eax,[intg] копирует из памяти по адресу intg данные в регистр eax. В свою очередь команда mov [intg],eax запишет в память по адресу intg данные из регистра eax. Также рассмотрим команду mov eax,intg В этом случае в регистр eax запишется адрес intg. Допустим, для intg выделена память начиная с ячейки с адресом 0x600144, тогда команда mov eax,intg аналогична команде mov eax,0x600144 – т.е. эта команда запишет в регистр eax число 0x600144.

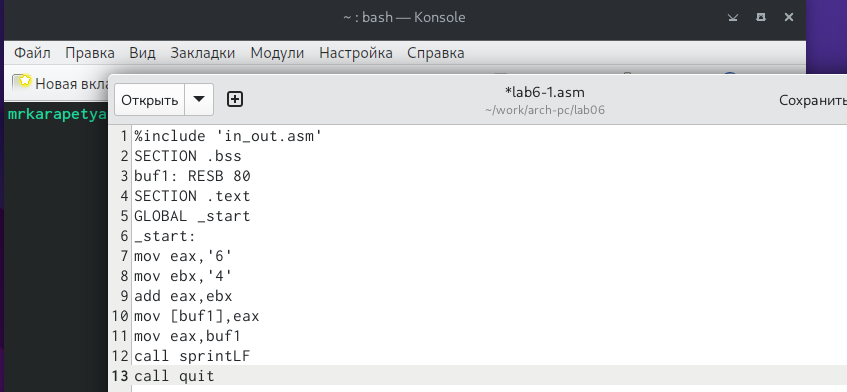
# 4 Выполнение лабораторной работы

Создайте каталог для программам лабораторной работы No 6, перейдите в него и создайте файл lab6-1.asm (рис. ??).



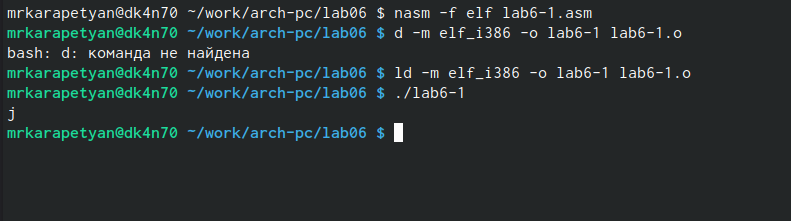
Создание каталога и файла лаб 6-1.asm

Рассмотрим примеры программ вывода символьных и численных значений. Програм- мы будут выводить значения записанные в регистр eax. (рис. ??).



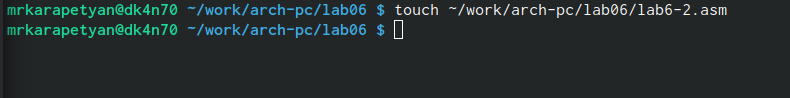
Программа 1

Создайте исполняемый файл и запустите его. (рис. ??).



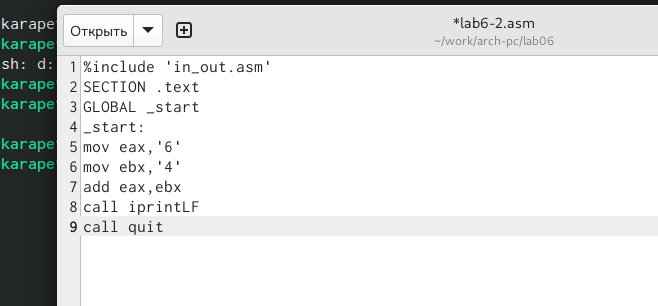
Запуск программы

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



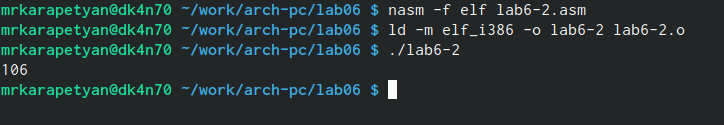
Создание файла lab6-2.asm

Введите в него текст программы из листинга 6.2. (рис. ??).



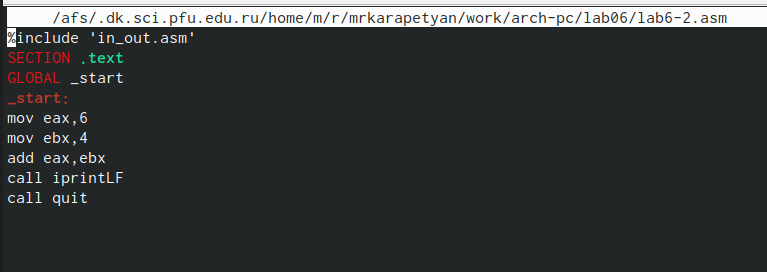
Введение текста

Создайте исполняемый файл и запустите его. (рис. ??).



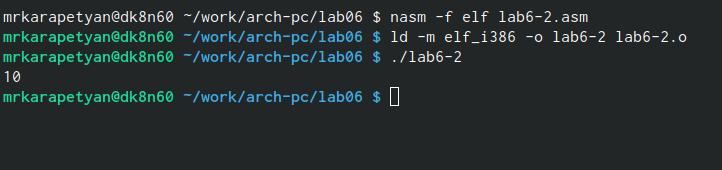
Запуск файла

Аналогично предыдущему примеру изменим символы на числа. Замените строки (рис. ??).



Замена символов на числа

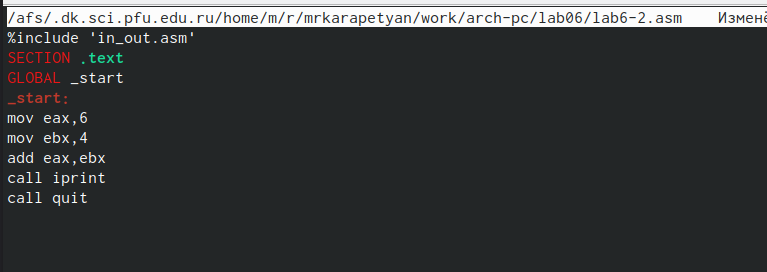
Создайте исполняемый файл и запустите его. (рис. ??).



Запуск файла

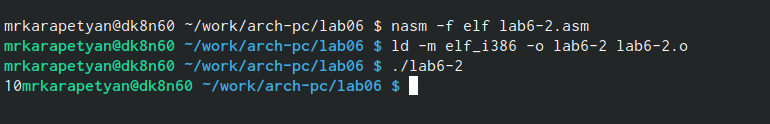
В результате полчили число 10.

Замените функцию iprintLF на iprint (рис. ??).



Замена функции

Создайте исполняемый файл и запустите его. (рис. ??).



Запуск файла

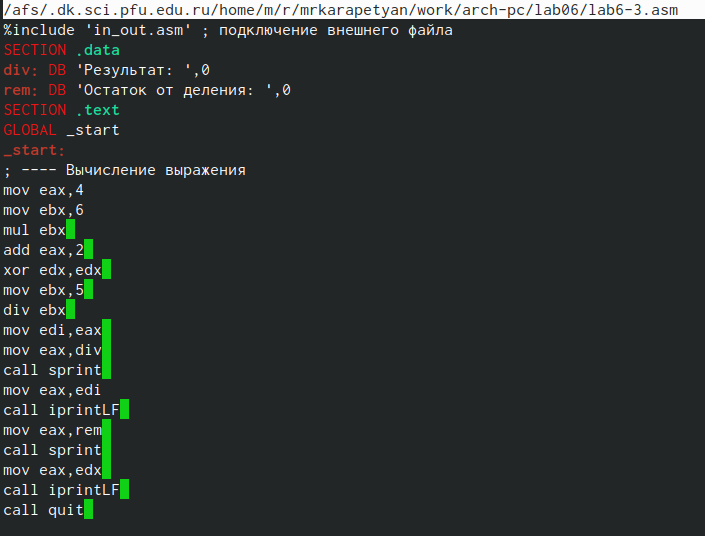
В качестве примера выполнения арифметических операций в NASM приведем про- грамму вычисления арифметического выражения 𝑓(𝑥) = (5 ∗ 2 + 3)/3

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

Создание файла lab6-3.asm

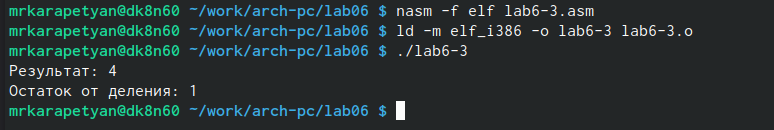
Создание файла lab6-3.asm

Внимательно изучите текст программы из листинга 6.3 и введите в lab6-3.asm. (рис. ??).



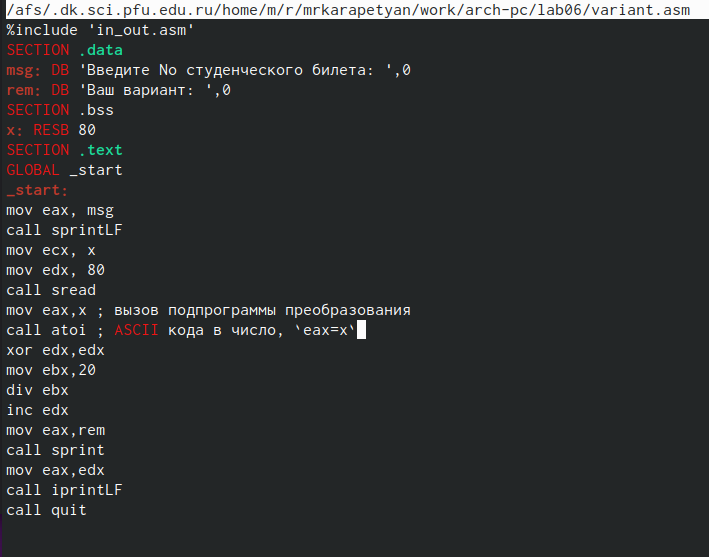
Введение текста

Создайте исполняемый файл и запустите его. (рис. ??).



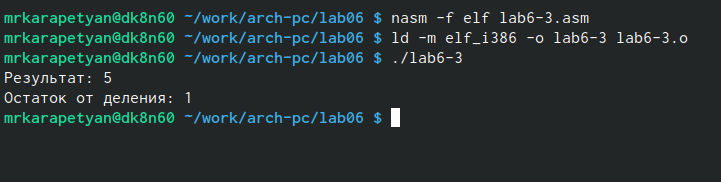
Запуск файла

Введите в него текст программы variant.asm (рис. ??).



Введение текста

Создайте исполняемый файл и проверьте его работу. (рис. ??).



Запуск файла

# 5 Ответы на вопросы

1 “mov eax, rem” и “call sprint” в секции кода отвечают за вывод сообщения “Ваш вариант:” на экран. 2 “mov ecx, x” и “mov edx, 80” загружают адрес буфера (x) и длину буфера (80) соответственно в регистры ecx и edx для вызова подпрограммы sread, которая считывает строку из консоли. 3 “call atoi” вызывает подпрограмму atoi для преобразования ASCII кодов символов в число, результат которого сохраняется в регистре eax. 4 Код для вычисления варианта начинается с “xor edx, edx” и “mov ebx, 20”, после чего происходит деление числа, сохраненного в eax, на 20 с помощью инструкции “div ebx”. Результат деления (цифра варианта) записывается в нижнюю часть регистра AX (AL). Затем происходит увеличение цифры варианта на единицу с помощью инструкции “inc edx”. 5 Результат деления (цифра варианта) записывается в нижнюю часть регистра AX (AL). 6 Инструкция “inc edx” увеличивает цифру варианта на единицу для того, чтобы результат деления не оказывался равным нулю. 7 “mov eax, edx” загружает цифру варианта из AL (нижней части регистра AX)в регистр eax для вывода результата на экран с помощью подпрограммы iprintLF.

# 6 Выводы

В ходе выполнения работы, я освоил работу с арифметическими операциями на языке assebly.

# Список литературы

GDB: The GNU Project Debugger. — URL: https://www.gnu.org/software/gdb/. GNU Bash Manual. — 2016. — URL: https://www.gnu.org/software/bash/manual/. Midnight Commander Development Center. — 2021. — URL: https://midnight-commander.org/. NASM Assembly Language Tutorials. — 2021. — URL: https://asmtutor.com/. Newham C. Learning the bash Shell: Unix Shell Programming. — O’Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: http://www.amazon.com/Learning- bash-Shell-Programming-Nutshell/dp/0596009658. Robbins A. Bash Pocket Reference. — O’Reilly Media, 2016. — 156 с. — ISBN 978-1491941591. The NASM documentation. — 2021. — URL: https://www.nasm.us/docs.php. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс,2017. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016. Расширенный ассемблер: NASM. — 2021. — URL: https://www.opennet.ru/docs/RUS/nasm/. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВ- Петербург, 2010. — 656 с. — ISBN 978-5-94157-538-1. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm\_unix. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science). Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер,2015 — 1120 с. — (Классика Computer Science). ::: {#refs} :::