Отчёт по лабораторной работе №9

Дисциплина:архитектура компьютера

Мирзоян Ян Игоревич

Содержание

# 1 Цель работы

Приобретение навыков написания программ с использованием подпрограмм. Знакомство с методами отладки при помощи GDB и его основными возможностями.

# 2 Задание

1. Реализация подпрограмм в NASM
2. Отладка программам с помощью GDB

# 3 Теоретическое введение

Подпрограмма — поименованная или иным образом идентифицированная часть компьютерной программы, содержащая описание определённого набора действий.

# 4 Выполнение лабораторной работы

##Реализация подпрограмм в NASM

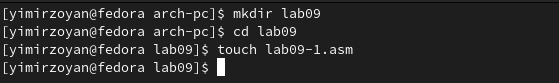


Figure 1: Создаю рабочую директорию и файл в ней

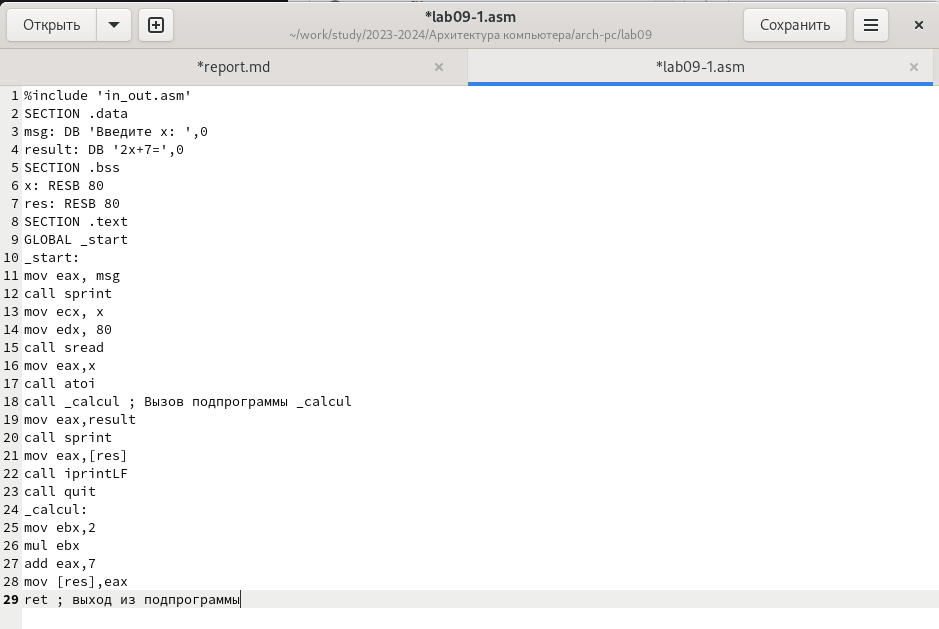


Figure 2: Редактирю файл вводя текст листинга

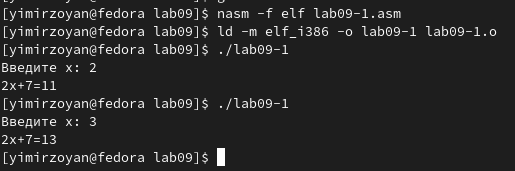


Figure 3: Создаю исполняемый файл. Проверяю корректность работы програмы

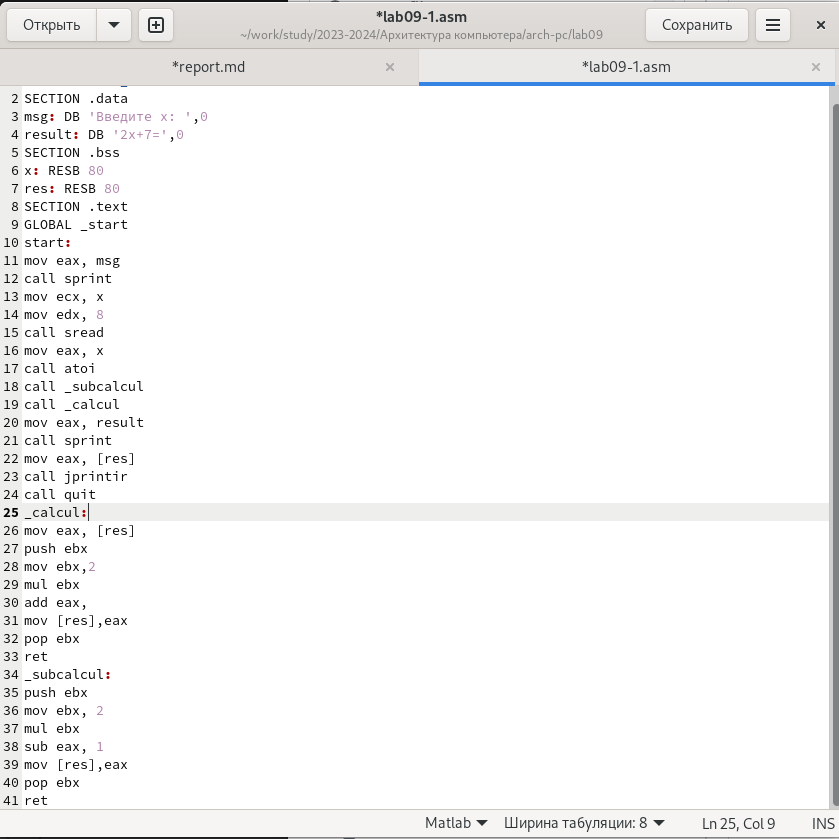


Figure 4: Изменяю текст программы чтобы она считала f(g(x))

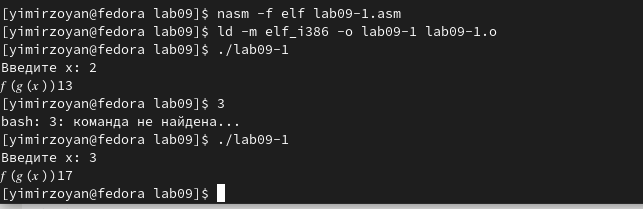


Figure 5: Создаю исполняемый файл. Проверяю корректность работы програмы

##Отладка программам с помощью GDB

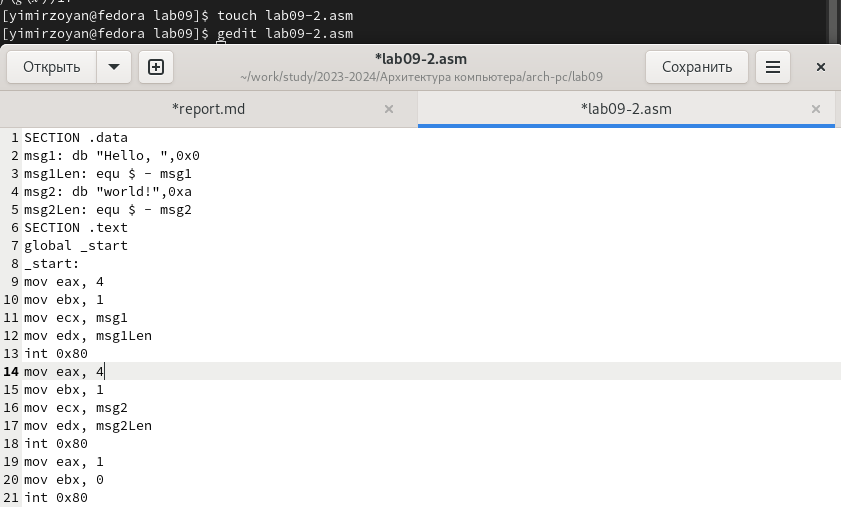


Figure 6: Создаю файл. Ввожу в него текст листинга

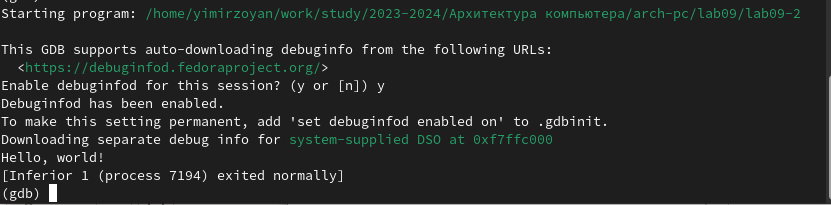


Figure 7: Запускаю программу в отладочной оболочке GDB

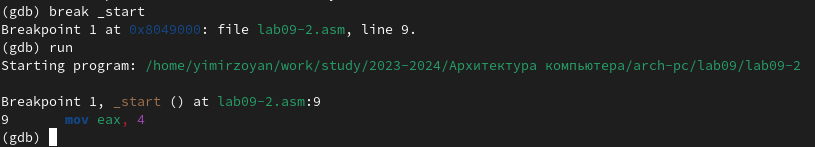


Figure 8: Для более подробного анализа программы устанавливаю брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запускаю её

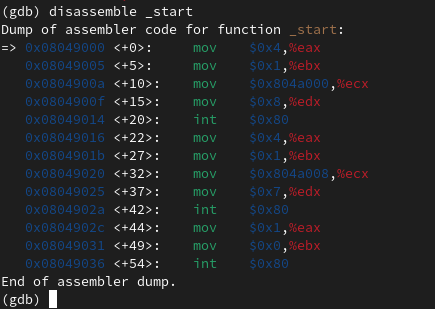


Figure 9: Смотрю дисассимилированный код программы с помощью команды disassemble начиная с метки \_start



Figure 10: Переключаюсь на отображение команд с Intel’овским синтаксисом, введя команду setdisassembly-flavor intel. В представлении ATT в виде 16-ричного числа записаны первые аргументы всех комманд, а в представлении intel так записываются адреса вторых аргументов

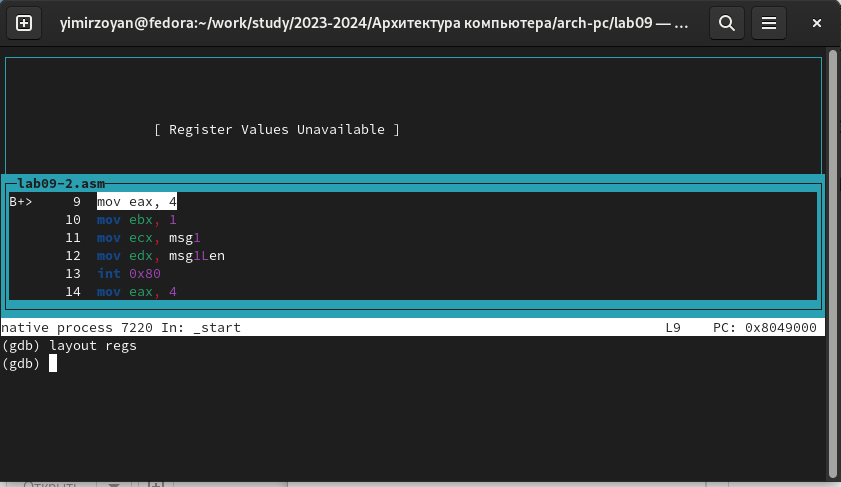


Figure 11: Включаю режим псевдографики, с помощью которго отбражается код программы и содержимое регистров

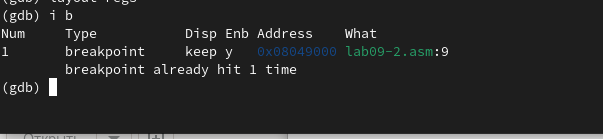


Figure 12: Проверяю наличие точки останова

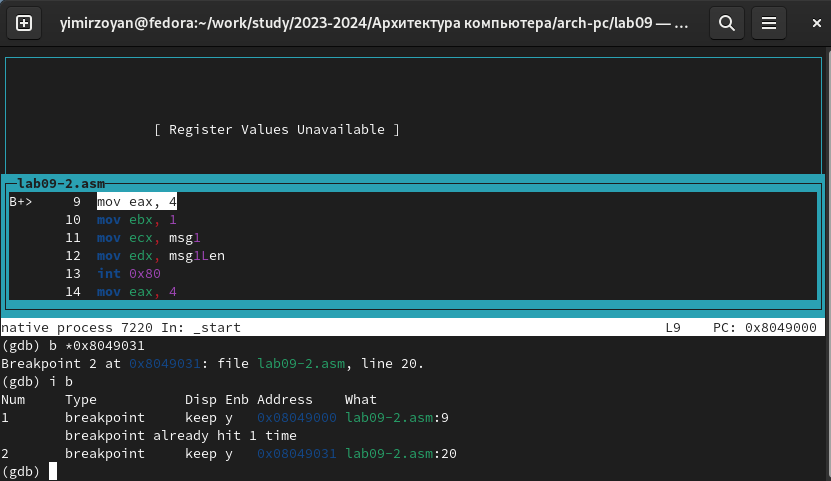


Figure 13: Добавляю ещё одну точку останова и проверяю сколько точек останова есть

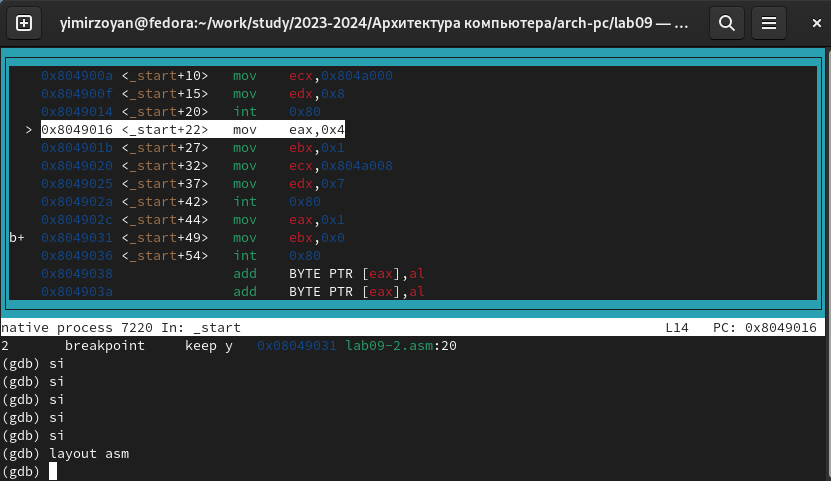


Figure 14: Выполняю 5 инструкций с помощью команды si. Изменились значения регистров eax ecx edx ebx

Figure 15: Просматриваю значение переменной msg1 по имени

Figure 15: Просматриваю значение переменной msg1 по имени

Figure 16: Просматриваю значение переменной msg2 по адресу

Figure 16: Просматриваю значение переменной msg2 по адресу

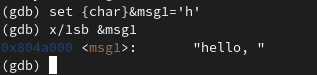


Figure 17: Изменяю первый символ переменной msg1

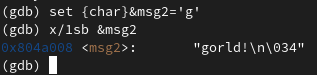


Figure 18: Изменяю первый символ переменной msg2

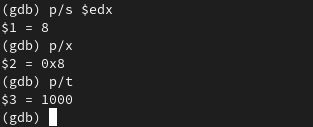


Figure 19: Вывожу значение edx в разных форматах: строчном, 16-ричном, двоичном

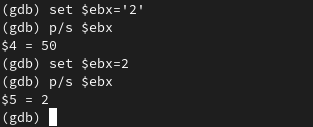


Figure 20: С помощью команды set изменяю значение регистра ebx. Разница вывода из-за того что в первом случае 2 это символ а во втором число.

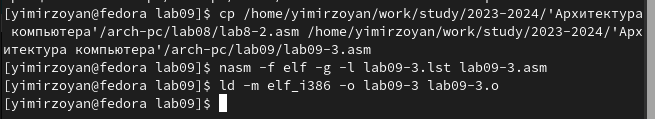


Figure 21: Скопировал файл lab8-2.asm, созданный при выполнении лабораторной работы No8, с программой выводящей на экран аргументы командной строки. в файл с именем lab09-3.asm, создал исполняемый файл

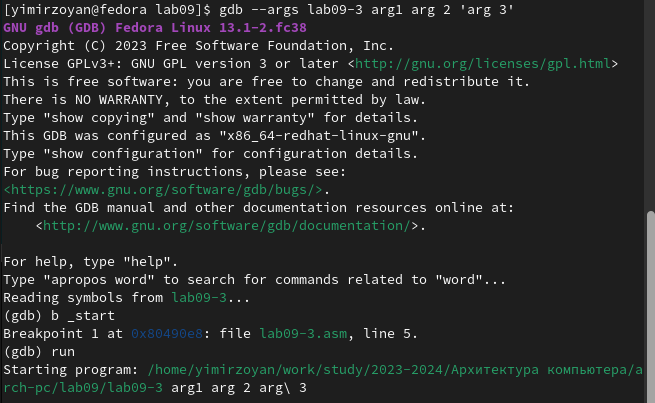


Figure 22: Запускаю программу в оболочке GDB

Figure 23: Узнаю количество аргументов

Figure 23: Узнаю количество аргументов

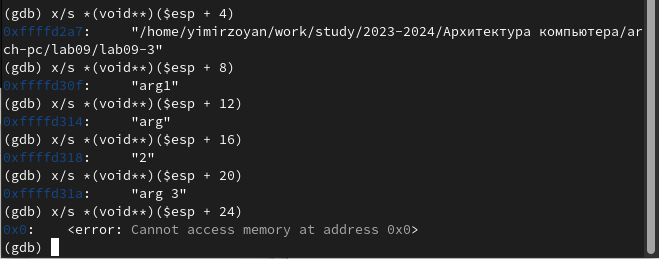


Figure 24: мотрю все позиции стека. Их адреса распологаются в 4 байтах друг от друга (именно столько заниемает элемент стека)

# 5 Выводы

В результате выполнения работы, я научился организовывать код в подпрограммы и познакомился с базовыми функциями отладчика gdb.

# Список литературы

1. [Лабораторная работа 9](https://esystem.rudn.ru/pluginfile.php/2089551/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%969.%20%D0%9F%D0%BE%D0%BD%D1%8F%D1%82%D0%B8%D0%B5%20%D0%BF%D0%BE%D0%B4%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D1%8B.%20%D0%9E%D1%82%D0%BB%D0%B0%D0%B4%D1%87%D0%B8%D0%BA%20..pdf)