Отчёт по лабораторной работе 9

дисциплина: Архитектура компьютера

Фархад Ахамд Камран

Содержание

# 1 Цель работы

Целью работы является приобретение навыков написания программ с использованием подпрограмм. Знакомство с методами отладки при помощи GDB и его основными возможностями.

# 2 Выполнение лабораторной работы

## 2.1 Реализация подпрограмм в NASM

Сначала я создал новую папку для выполнения лабораторной работы №9 и перешел в нее. Затем создал файл с именем lab9-1.asm.

В качестве примера рассмотрел программу, которая вычисляет арифметическое выражение с использованием подпрограммы calcul. В этой программе значение переменной вводится с клавиатуры, а вычисление выражения осуществляется внутри подпрограммы. (рис. 1, 2)

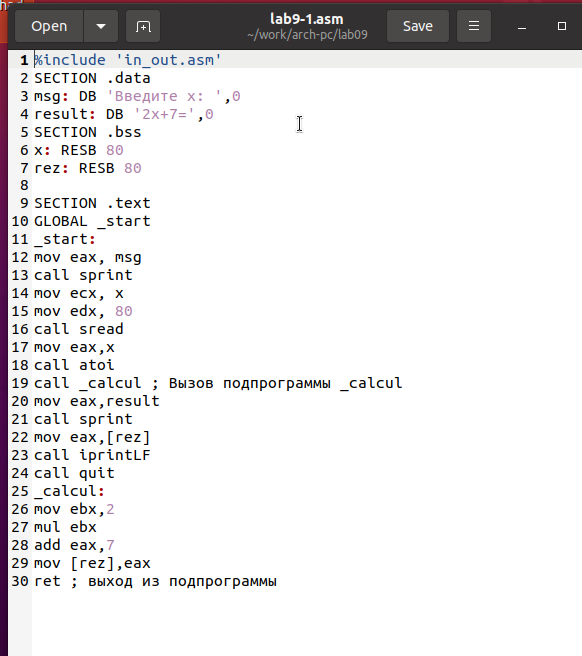


Рис. 1: Программа lab9-1.asm

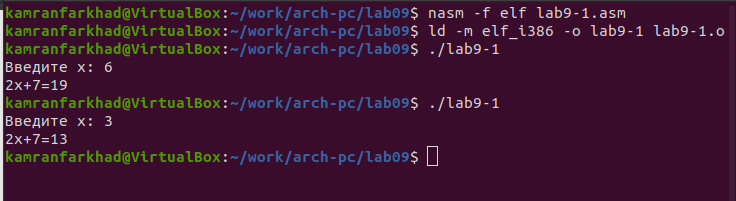


Рис. 2: Запуск программы lab9-1.asm

После этого я модифицировал программу, добавив подпрограмму subcalcul внутри calcul. Это позволило вычислить составное выражение , где значение также вводится с клавиатуры. Определения функций: , . (рис. 3, 4)

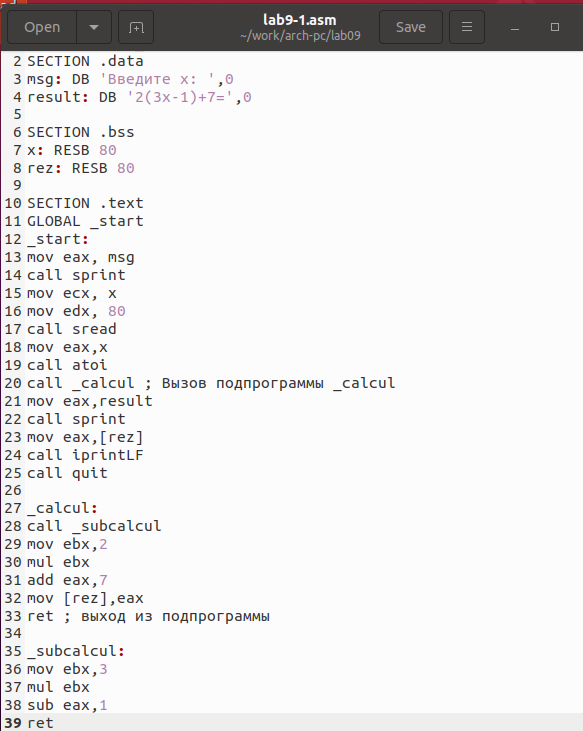


Рис. 3: Программа lab9-1.asm

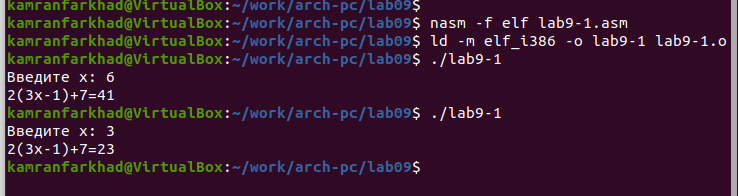


Рис. 4: Запуск программы lab9-1.asm

## 2.2 Отладка программы с помощью GDB

Создал файл lab9-2.asm, содержащий программу из Листинга 9.2, которая выводит сообщение “Hello world!” на экран. (рис. 5)

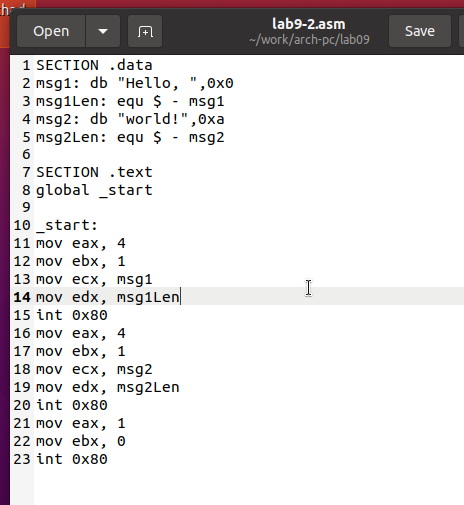


Рис. 5: Программа lab9-2.asm

Скомпилировал файл и создал исполняемый файл, добавив ключ -g для включения отладочной информации. Загрузил исполняемый файл в отладчик GDB и запустил программу с помощью команды run. (рис. 6)

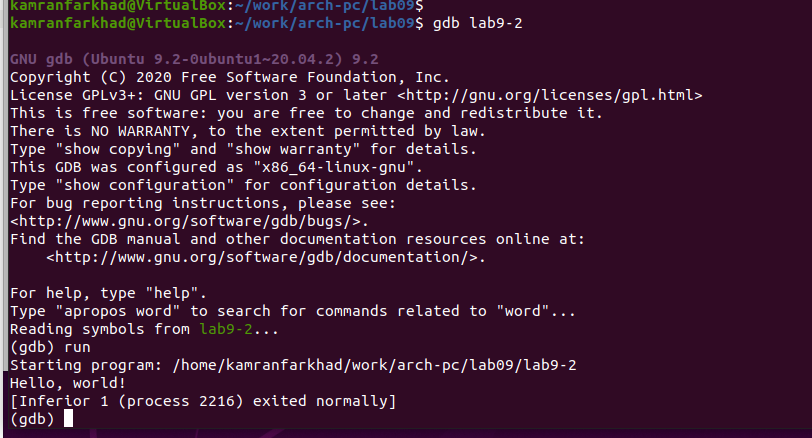


Рис. 6: Запуск программы lab9-2.asm в отладчике

Для детального анализа установил точку остановки на метке \_start и изучил дизассемблированный код программы. (рис. 7, 8)

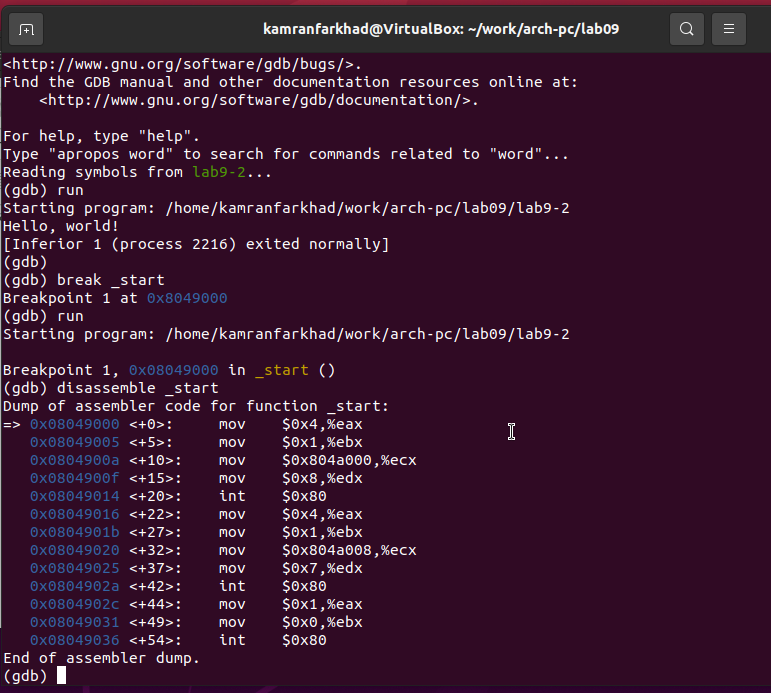


Рис. 7: Дизассемблированный код

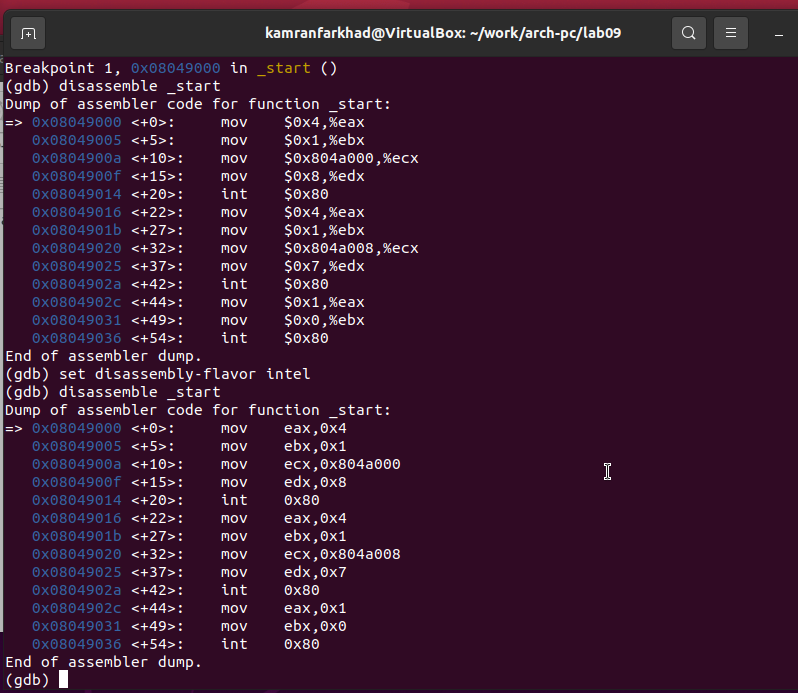


Рис. 8: Дизассемблированный код в режиме Intel

Установил точку останова по имени метки \_start с помощью команды info breakpoints и добавил еще одну точку остановки по адресу предпоследней инструкции mov ebx, 0x0. (рис. 9)

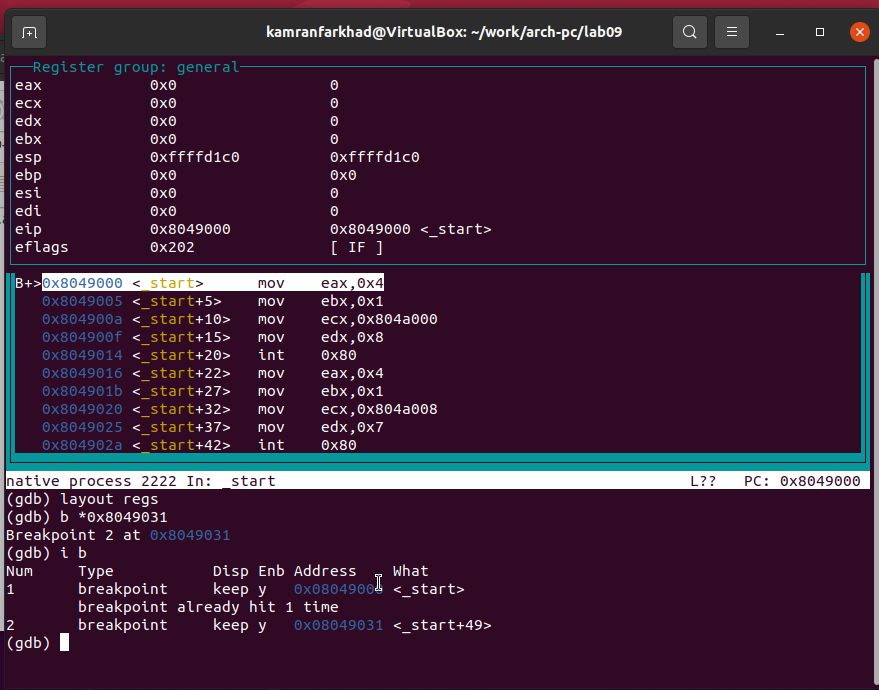


Рис. 9: Точка остановки

С помощью команды stepi выполнил пошаговое выполнение первых пяти инструкций, наблюдая за изменениями в регистрах. (рис. 10, 11)

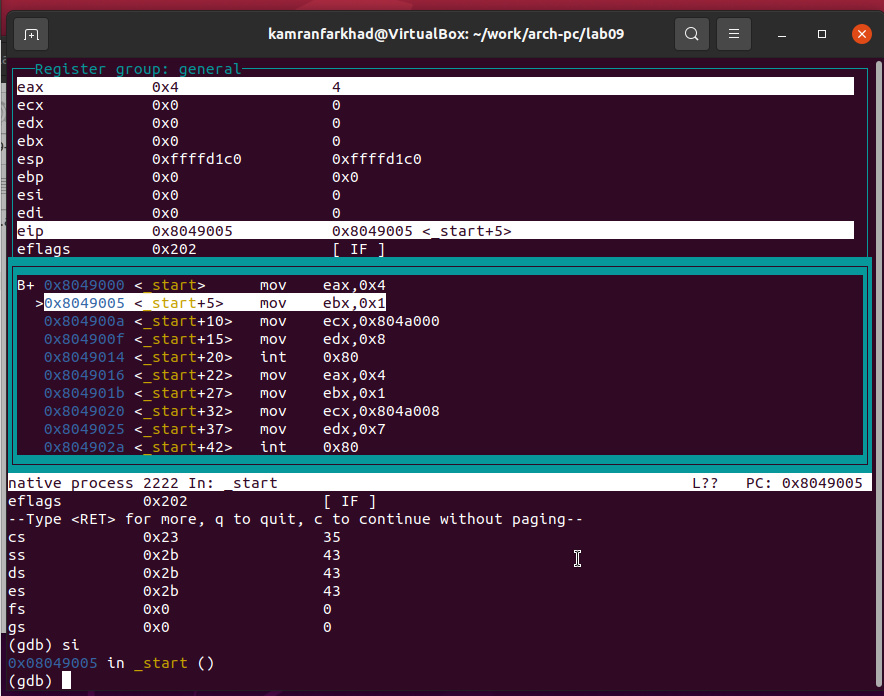


Рис. 10: Изменение регистров

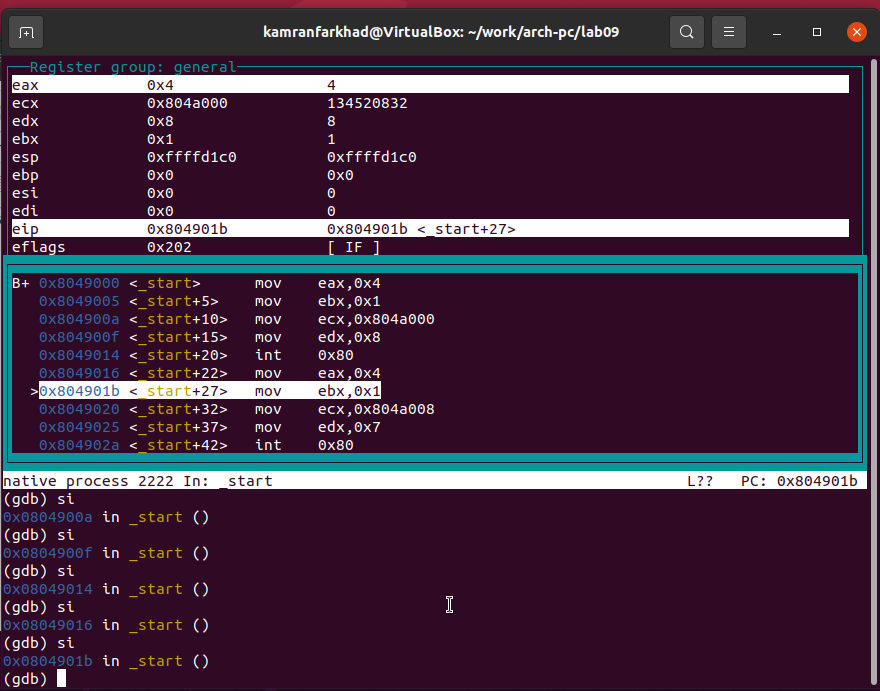


Рис. 11: Изменение регистров

Для анализа переменных использовал команду set, изменив первый символ переменной msg1. (рис. 12, 13)

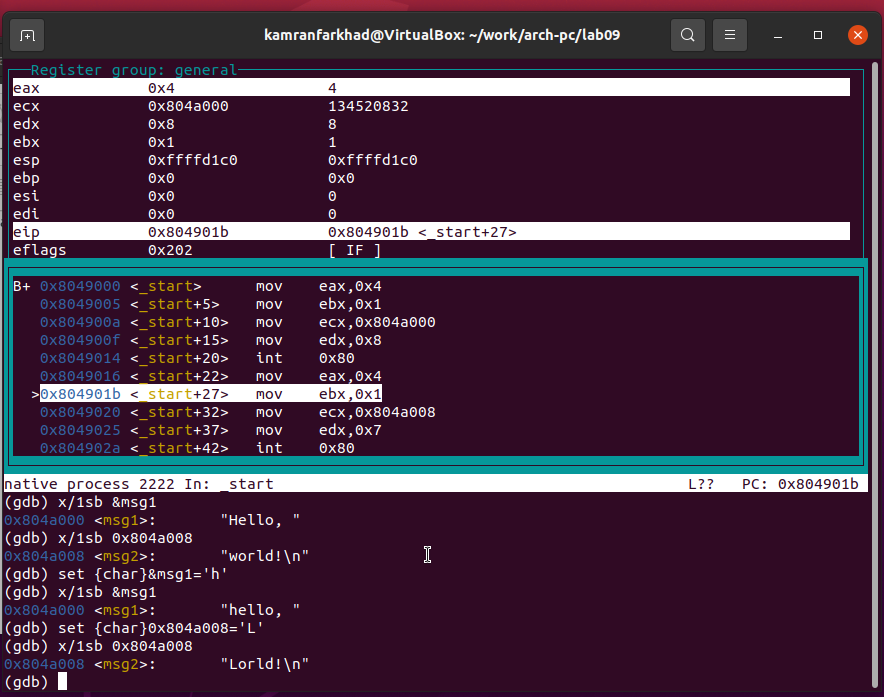


Рис. 12: Изменение значения переменной

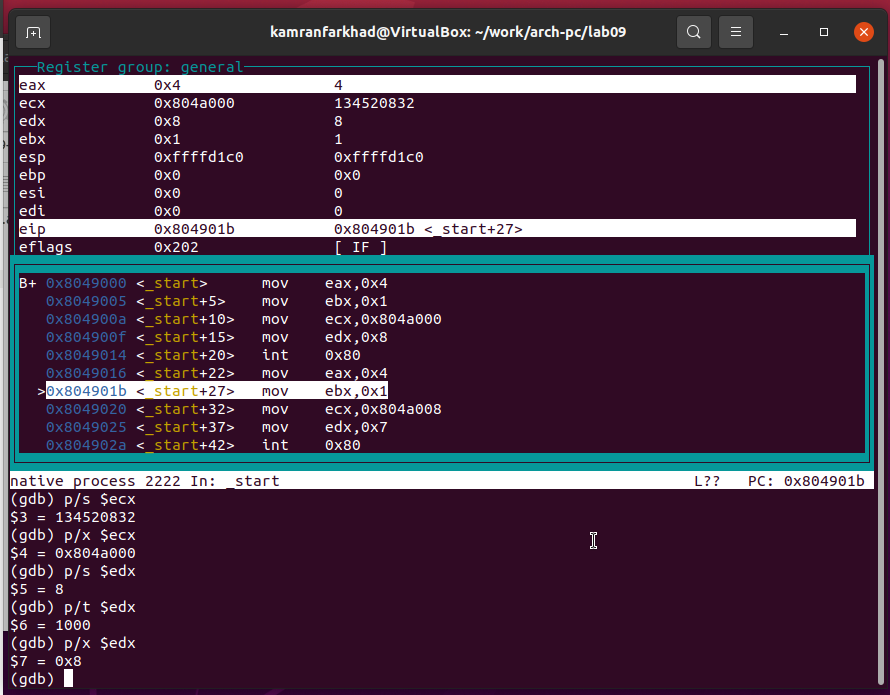


Рис. 13: Вывод значения регистра

Также изменил значение регистра ebx на нужное. (рис. 14)

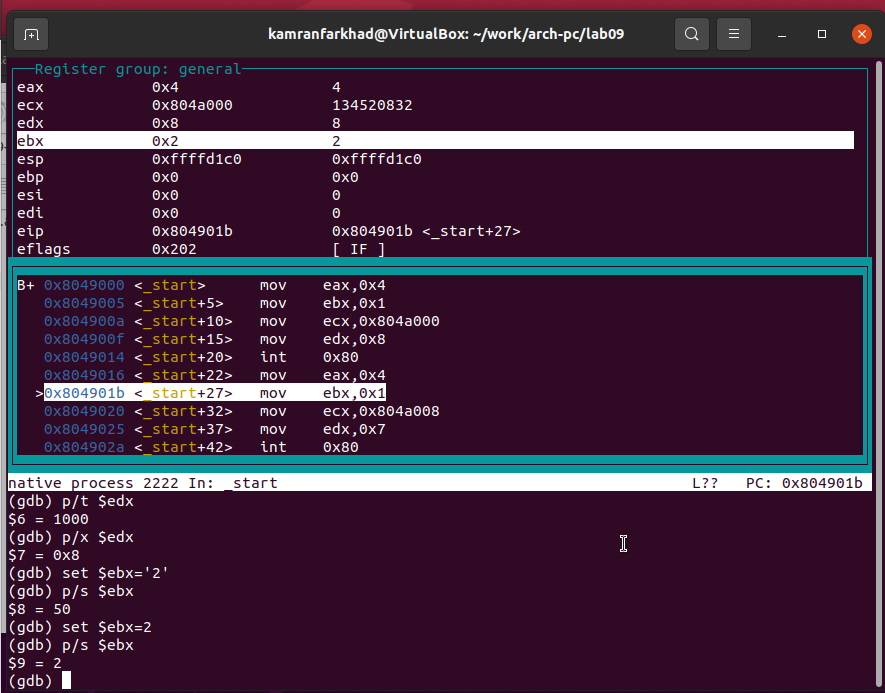


Рис. 14: Вывод значения регистра

Скопировал файл lab8-2.asm из лабораторной работы №8 и создал исполняемый файл. Использовал ключ –args для передачи аргументов в программу при запуске через GDB. Исследовал содержимое стека, где в esp находится количество аргументов, а остальные позиции содержат указатели на строки. (рис. 15)

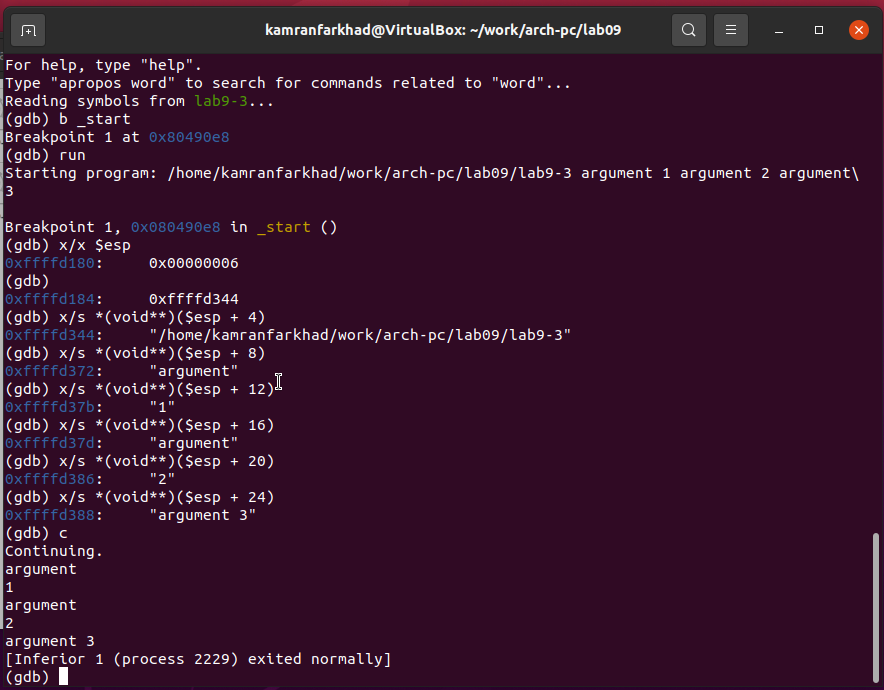


Рис. 15: Вывод значений стека

## 2.3 Задание для самостоятельной работы

Преобразовал программу из лабораторной работы №8, добавив вычисление функции в виде подпрограммы. (рис. 16, 17)

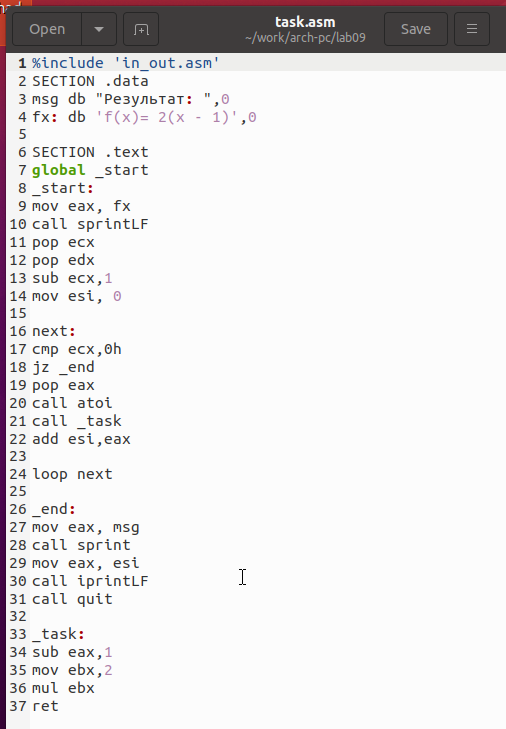


Рис. 16: Программа task-1.asm

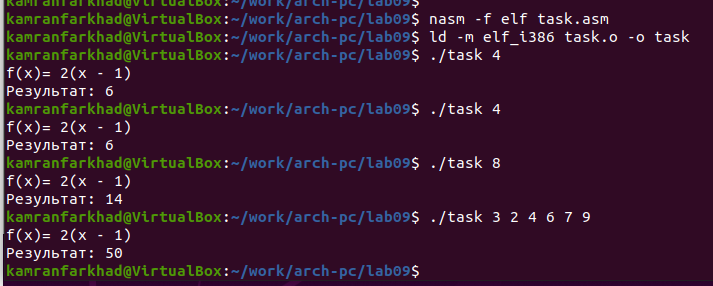


Рис. 17: Запуск программы task-1.asm

В процессе анализа обнаружил ошибки: перепутан порядок аргументов у инструкции add и отправка ebx вместо eax в конце. Исправил ошибки. (рис. 18, 19)

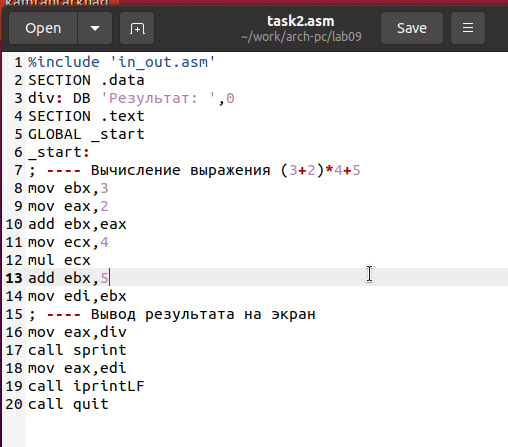


Рис. 18: Код с ошибкой

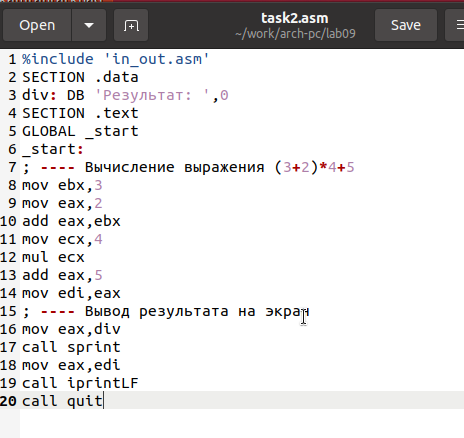


Рис. 19: Исправленный код

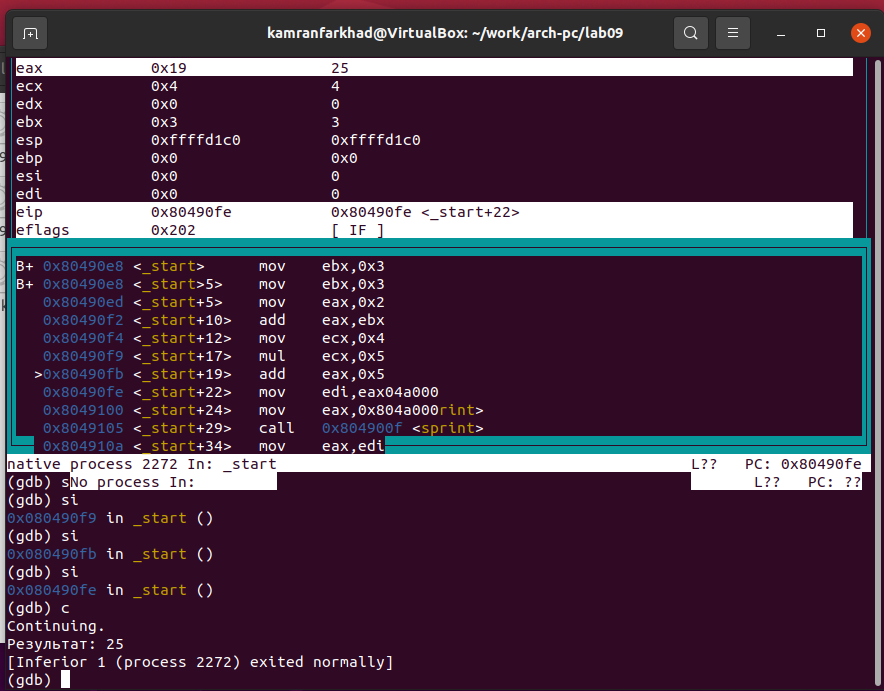


Рис. 20: Проверка работы

# 3 Выводы

В ходе лабораторной работы освоил работу с подпрограммами на NASM и изучил методы отладки с использованием GDB.