Отчёт по лабораторной работе 9

Понятие подпрограммы. Отладчик GDB.

Калашникова Дарья Викторовна

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 создала новую папку для выполнения лабораторной работы №9 и перешла в нее. Затем создала файл с именем lab9-1.asm.

В качестве примера я реализовала программу, вычисляющую арифметическое выражение с использованием подпрограммы calcul. Значение переменной вводится с клавиатуры, а само выражение вычисляется в подпрограмме.

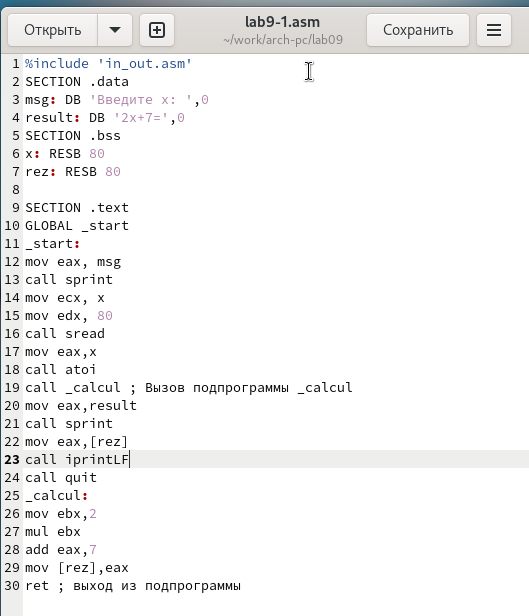


Рис. 1: Исходный код программы lab9-1.asm

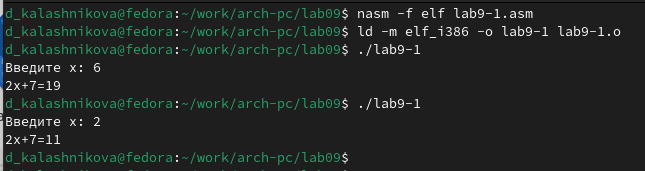


Рис. 2: Результат выполнения программы

Затем я изменила текст программы, добавив подпрограмму subcalcul внутрь подпрограммы calcul. Это позволило вычислять составное выражение , где , . Значение также вводится с клавиатуры.

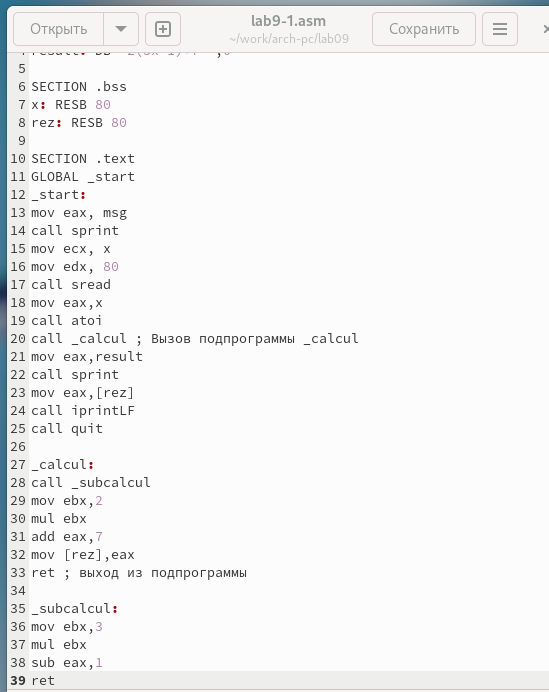


Рис. 3: Модифицированный код программы

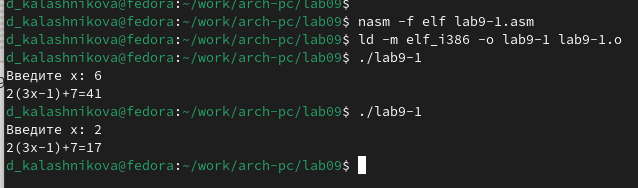


Рис. 4: Результат выполнения модифицированной программы

## 2.2 Отладка программы с помощью GDB

Я создала файл lab9-2.asm, содержащий программу для вывода сообщения “Hello, world!” (Листинг 9.2).

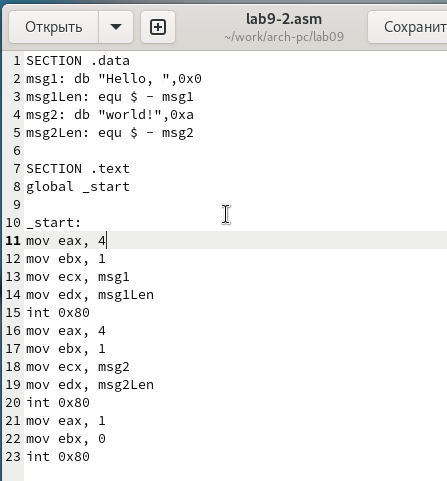


Рис. 5: Код программы lab9-2.asm

Скомпилировала файл с ключом -g для добавления отладочной информации и загрузила его в GDB. Затем запустила программу командой run.

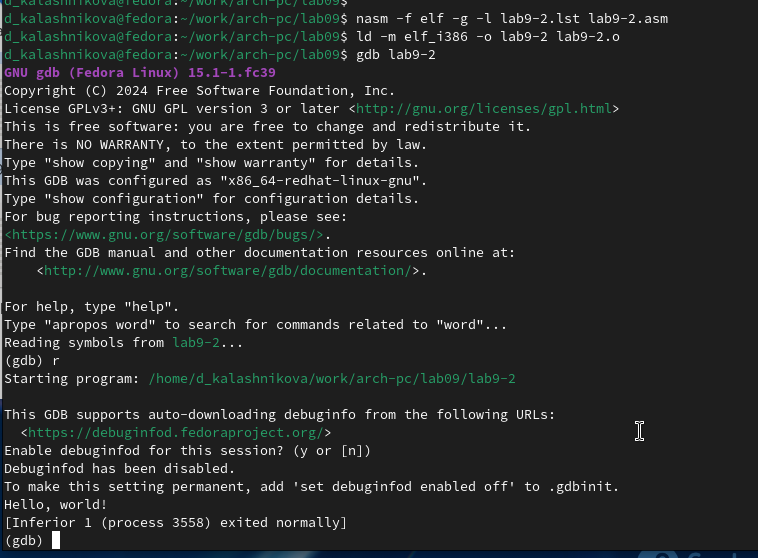


Рис. 6: Запуск программы в отладчике

Установила точку останова на метке \_start, запустила программу, а затем просмотрела дизассемблированный код.

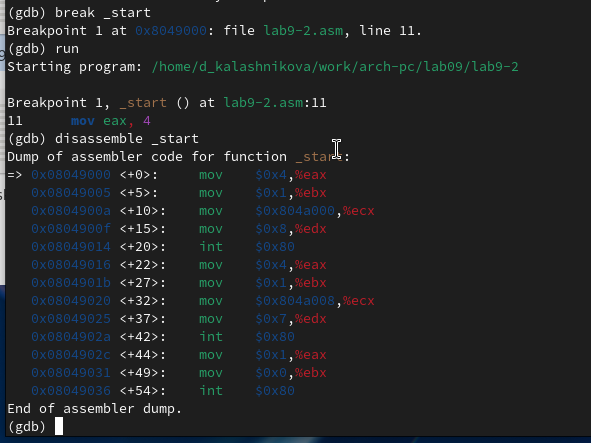


Рис. 7: Дизассемблированный код

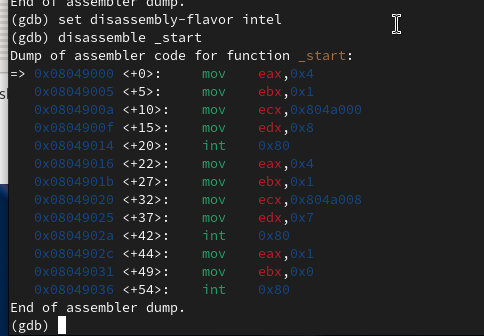


Рис. 8: Дизассемблированный код в режиме Intel

Установила дополнительные точки останова, используя команды info breakpoints и break. Например, добавила точку на инструкции mov ebx, 0x0.

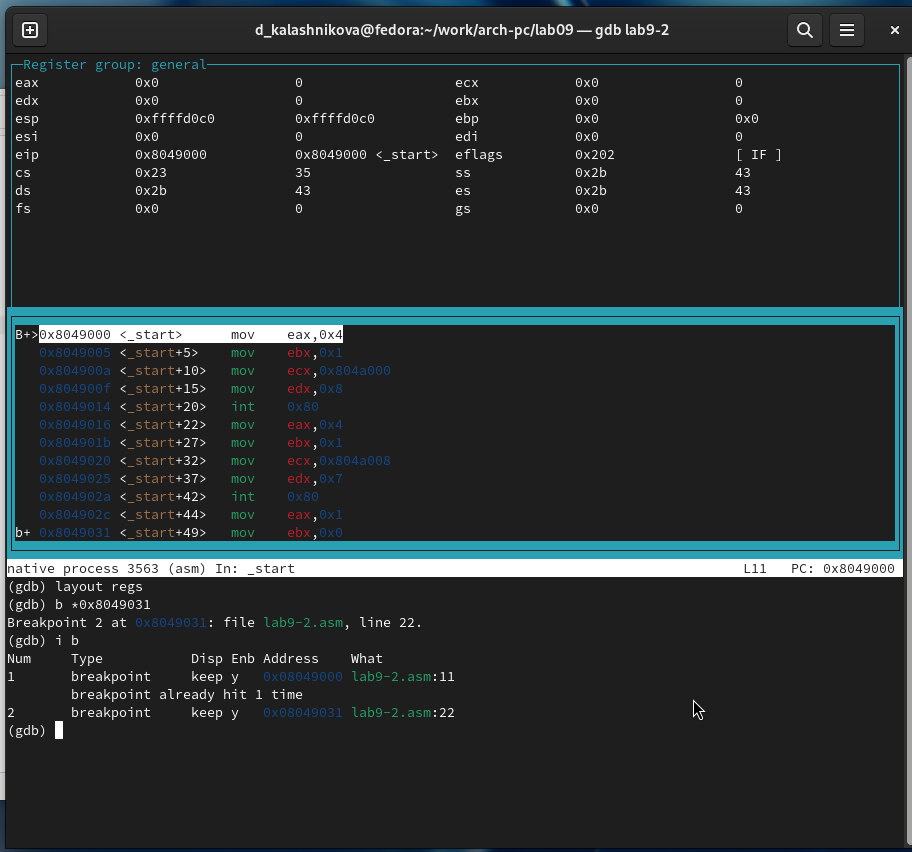


Рис. 9: Установка точки останова

С помощью команды stepi (или si) я пошагово выполняла инструкции, отслеживая изменения регистров.

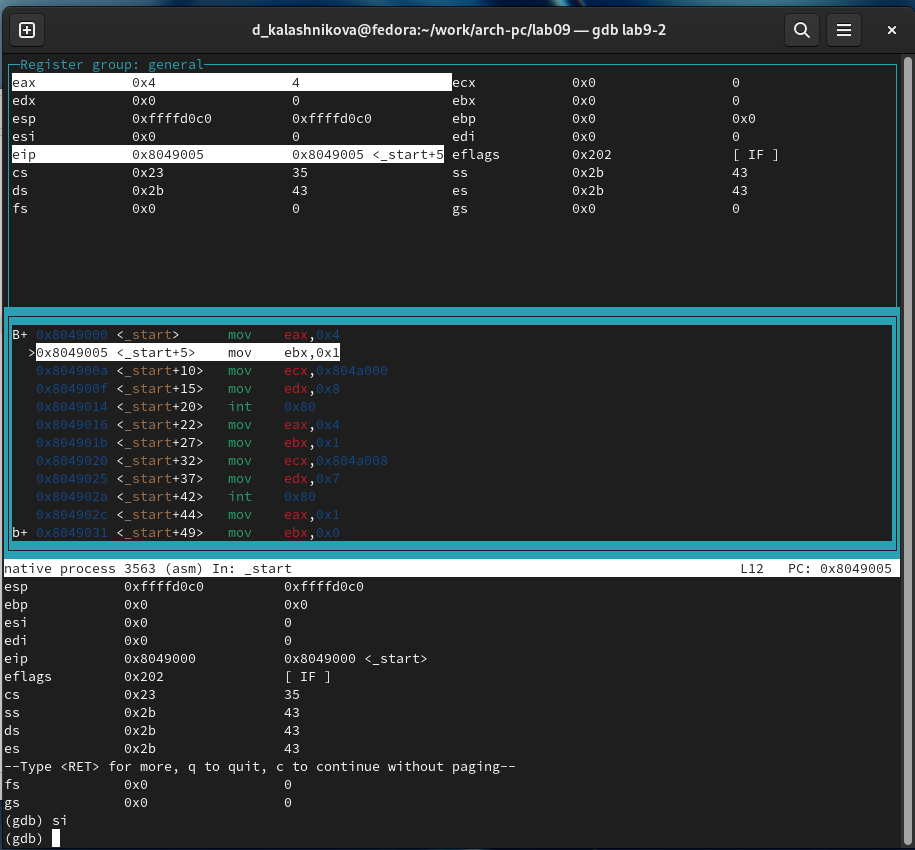


Рис. 10: Изменение значений регистров

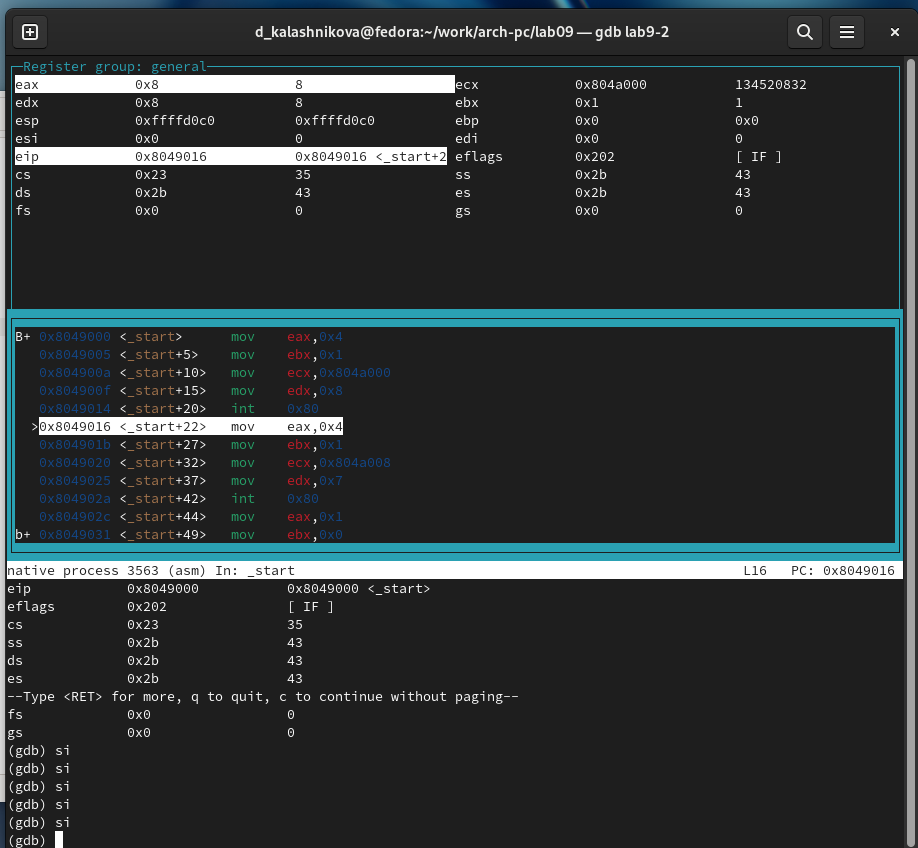


Рис. 11: Отслеживание изменений регистров

Я также изменила значение переменной msg1 и регистров, используя команду set.



Рис. 12: Изменение переменной

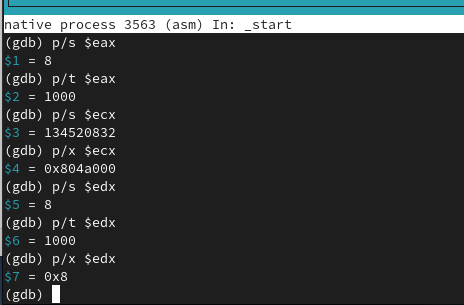


Рис. 13: Отображение измененного регистра

Используя аналогичные команды, я изменила значение регистра ebx.

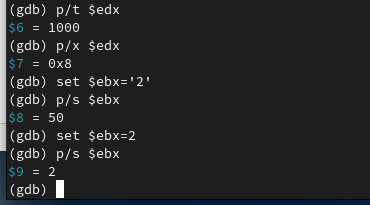


Рис. 14: Изменение регистра ebx

## 2.3 Работа с аргументами командной строки

Для работы с аргументами командной строки я использовала файл lab8-2.asm (из лабораторной работы №8), создав из него исполняемый файл. Затем загрузила программу в GDB с аргументами, используя ключ –args.

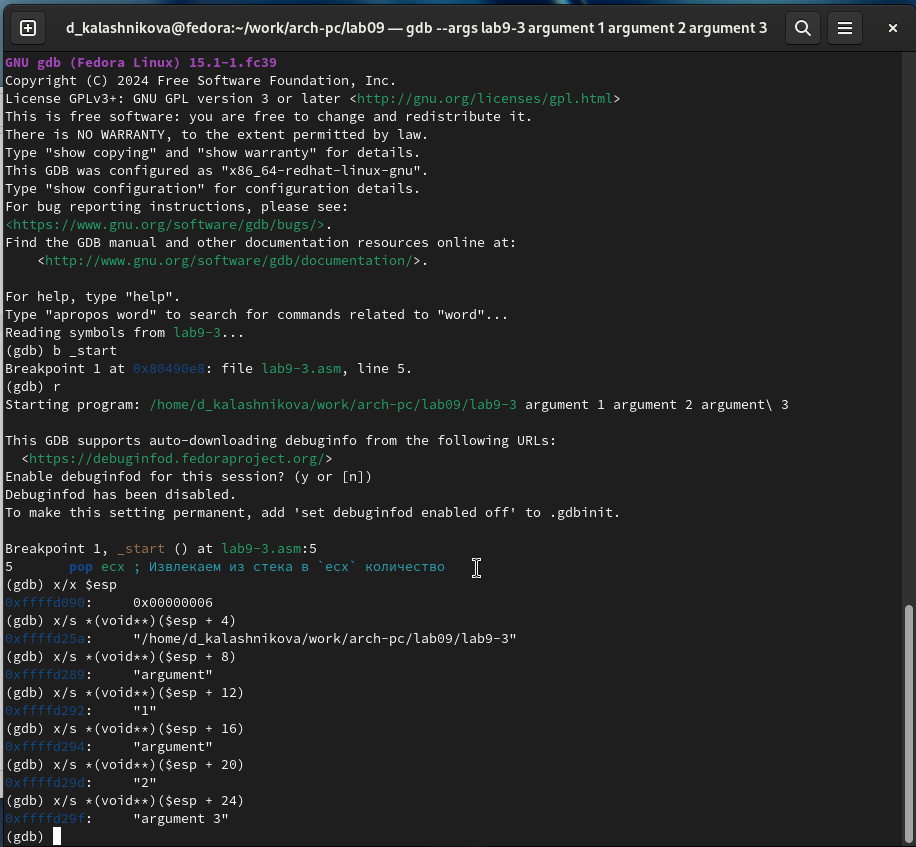


Рис. 15: Просмотр аргументов командной строки

## 2.4 Задание для самостоятельной работы

В рамках задания я модифицировала программу из лабораторной работы №8, добавив подпрограмму для вычисления функции .

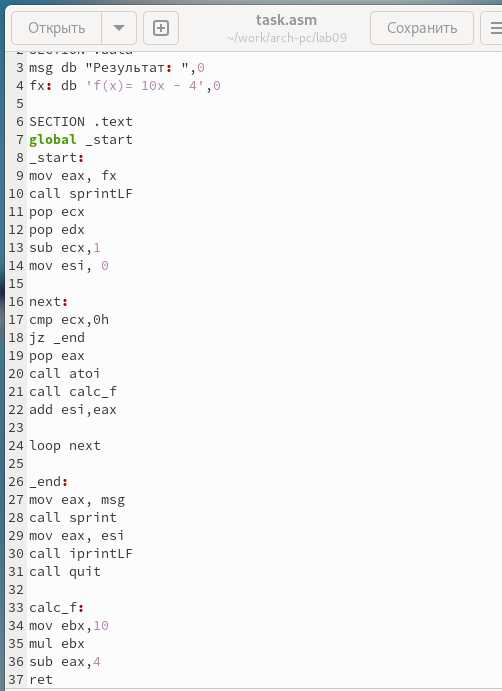


Рис. 16: Код программы prog-1.asm

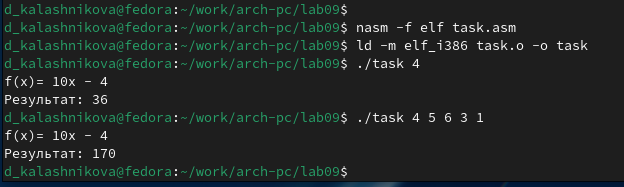


Рис. 17: Результат выполнения программы

В процессе выполнения программы я обнаружила ошибку: порядок аргументов в инструкции add был перепутан, а регистр ebx вместо eax отправлялся в edi.

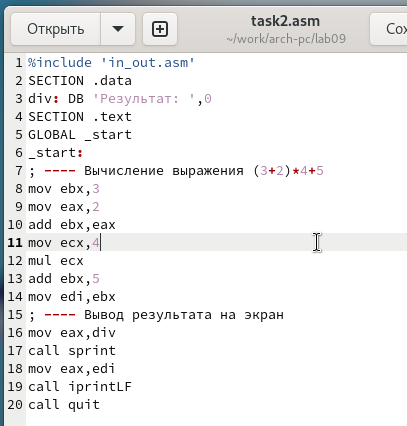


Рис. 18: Код с ошибками

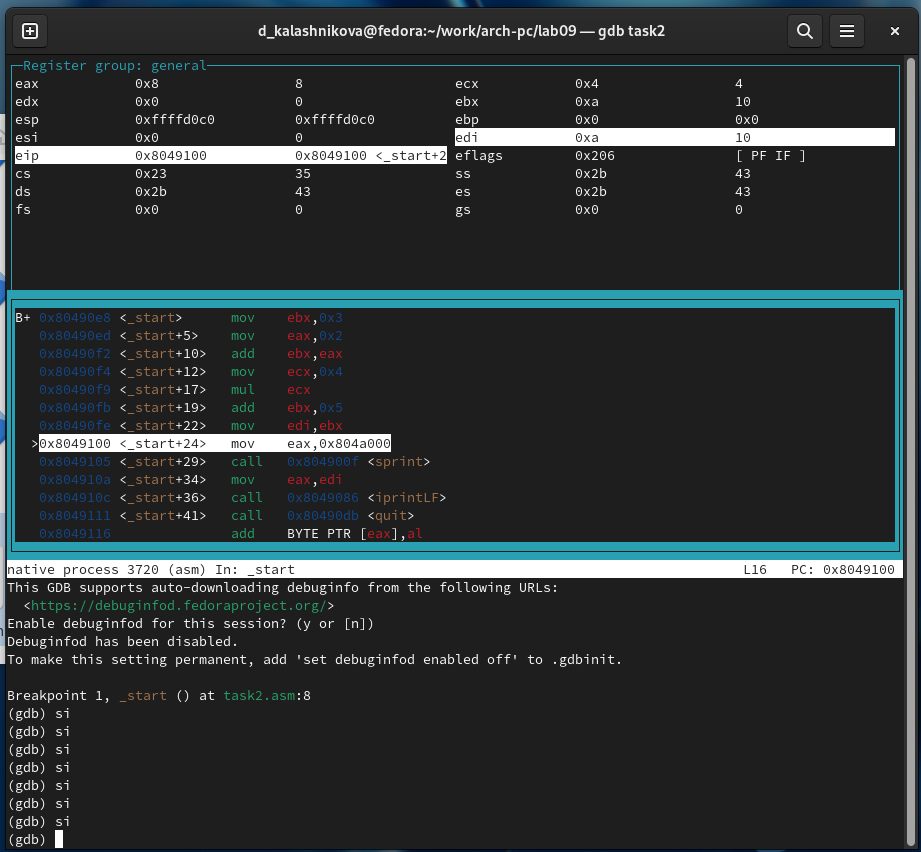


Рис. 19: Результат отладки

После исправления ошибок программа заработала корректно.

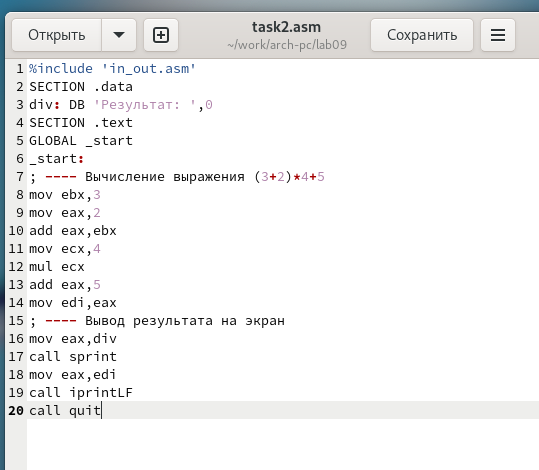


Рис. 20: Исправленный код программы

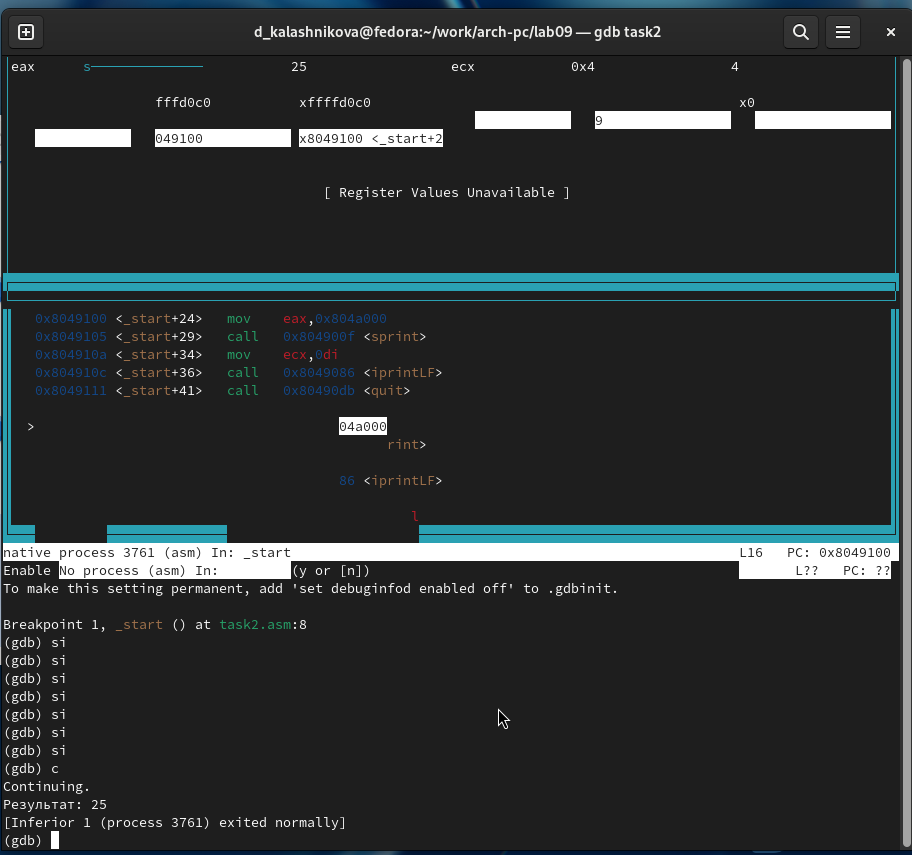


Рис. 21: Результат проверки

## 2.5 Выводы

В ходе лабораторной работы я научилась работать с подпрограммами и отладчиком GDB, а также диагностировать и исправлять ошибки в ассемблерных программах.