Лабораторная работа №9

НКАбд-02-23

Выборнов Дмитрий Валерьевич

# 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. Добавление точек останова.
4. Обработка аргументов командной строки в GDB.
5. Задание для самостоятельной работы.

# 3 Теоретическое введение

## 3.1 Отладка.

Отладка — это процесс поиска и исправления ошибок в программе. В общем случае его можно разделить на четыре этапа: • обнаружение ошибки; • поиск её местонахождения; • определение причины ошибки; • исправление ошибки.

Наиболее часто применяют следующие методы отладки: • создание точек контроля значений на входе и выходе участка программы (например, вывод промежуточных значений на экран — так называемые диагностические сообщения); • использование специальных программ-отладчиков. Отладчики позволяют управлять ходом выполнения программы, контролировать и изменять данные. Это помогает быстрее найти место ошибки в программе и ускорить её исправление. Наиболее популярные способы работы с отладчиком — это использование точек останова и выполнение программы по шагам.

Пошаговое выполнение — это выполнение программы с остановкой после каждой строчки, чтобы программист мог проверить значения переменных и выполнить другие действия. Точки останова — это специально отмеченные места в программе, в которых программаотладчик приостанавливает выполнение программы и ждёт команд.

GDB (GNU Debugger — отладчик проекта GNU) работает на многих UNIX-подобных системах и умеет производить отладку многих языков программирования. GDB предлагает обширные средства для слежения и контроля за выполнением компьютерных программ. Отладчик не содержит собственного графического пользовательского интерфейса и использует стандартный текстовый интерфейс консоли. Однако для GDB существует несколько сторонних графических надстроек, а кроме того, некоторые интегрированные среды разработки используют его в качестве базовой подсистемы отладки. Отладчик GDB (как и любой другой отладчик) позволяет увидеть, что происходит «внутри» программы в момент её выполнения или что делает программа в момент сбоя.

## 3.2 Подпрограммы.

Подпрограмма — это, как правило, функционально законченный участок кода, который можно многократно вызывать из разных мест программы. В отличие от простых переходов из подпрограмм существует возврат на команду, следующую за вызовом. Если в программе встречается одинаковый участок кода, его можно оформить в виде подпрограммы, а во всех нужных местах поставить её вызов. При этом подпрограмма будет содержаться в коде в одном экземпляре, что позволит уменьшить размер кода всей программы.

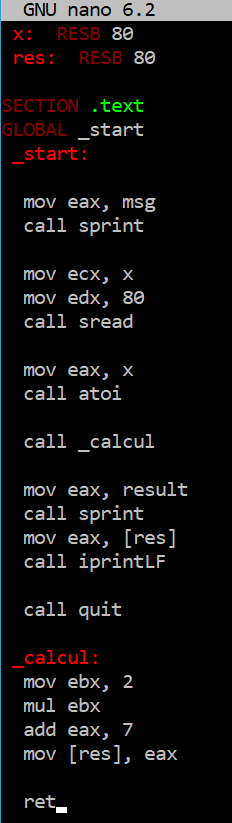
Для вызова подпрограммы из основной программы используется инструкция call, которая заносит адрес следующей инструкции в стек и загружает в регистр eip адрес соответствующей подпрограммы, осуществляя таким образом переход. Затем начинается выполнение подпрограммы, которая, в свою очередь, также может содержать подпрограммы. Подпрограмма завершается инструкцией ret, которая извлекает из стека адрес, занесённый туда соответствующей инструкцией call, и заносит его в eip. После этого выполнение основной программы возобновится с инструкции, следующей за инструкцией call.

Подпрограмма может вызываться как из внешнего файла, так и быть частью основной программы.

# 4 Выполнение лабораторной работы

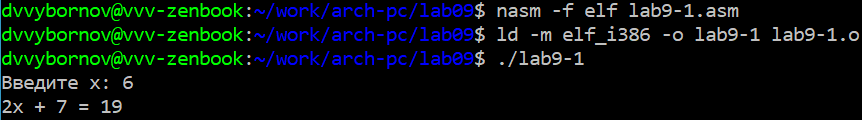
## 4.1 Реализация подпрограмм в NASM.

Создаю новый файл и ввожу в него текст программы.



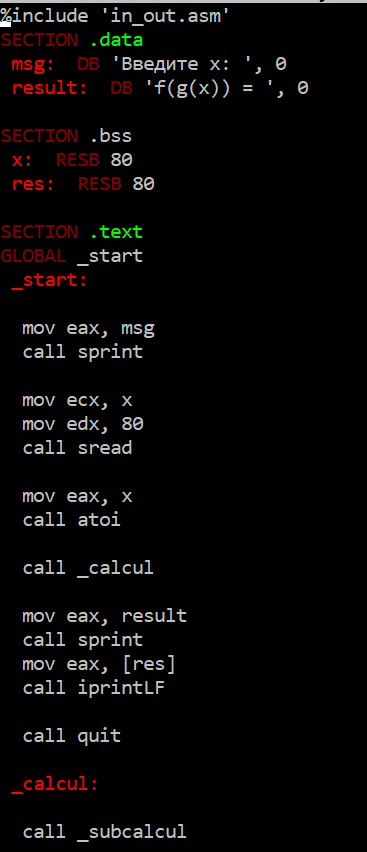
Первый шаг.

Создаю исполняемый файл и проверяю его работу.



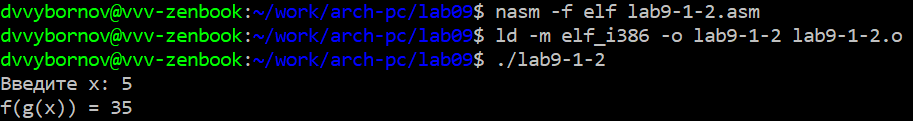
Второй шаг.

Изменяю текст программы.



Третий шаг.

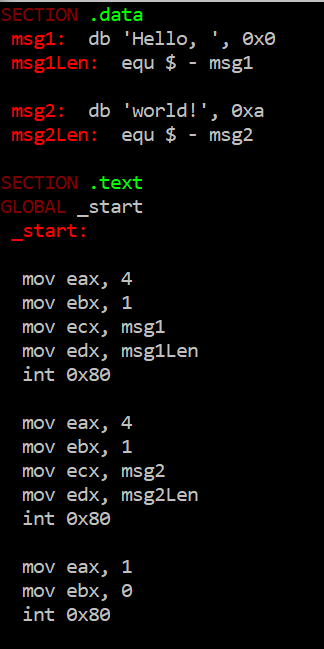
Запускаю изменённую программу и проверяю её работу.



Четвёртый шаг.

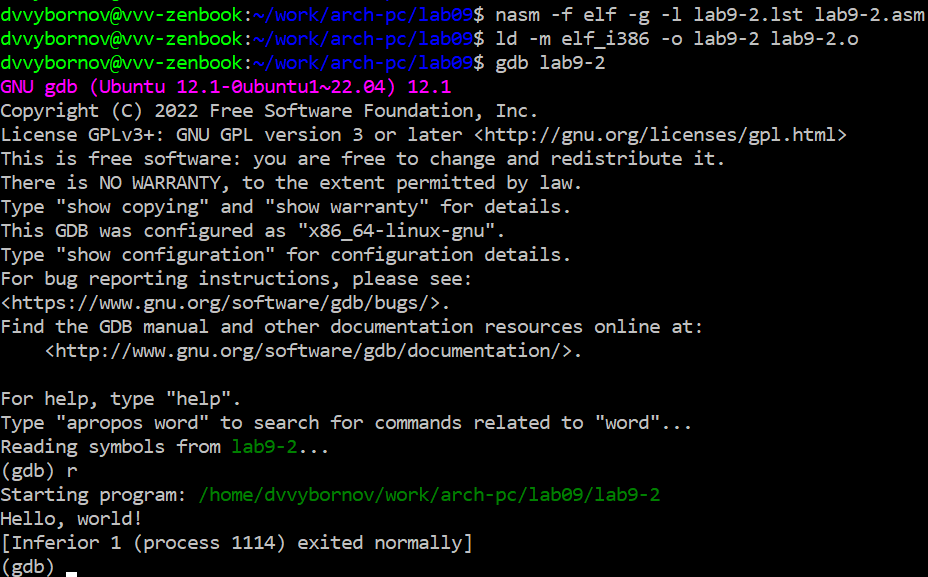
## 4.2 Отладка программам с помощью GDB.

Создаю новый файл и ввожу в него текст программы вывода Hello, world!



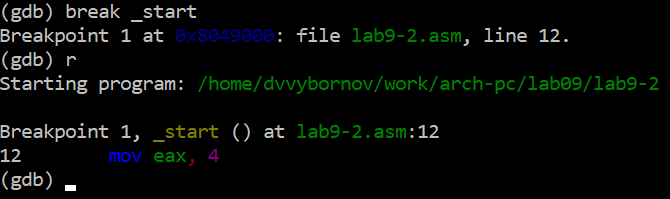
Пятый шаг.

Получаю исполняемый файл, загружаю его в GDB и проверяю его при помощи команды run.



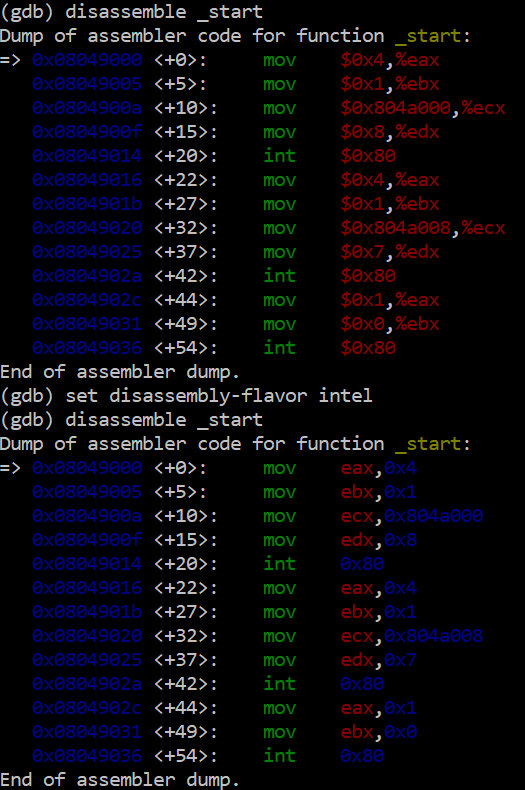
Шестой шаг.

Запускаю прогамму с брейкпоинтом.



Седьмой шаг.

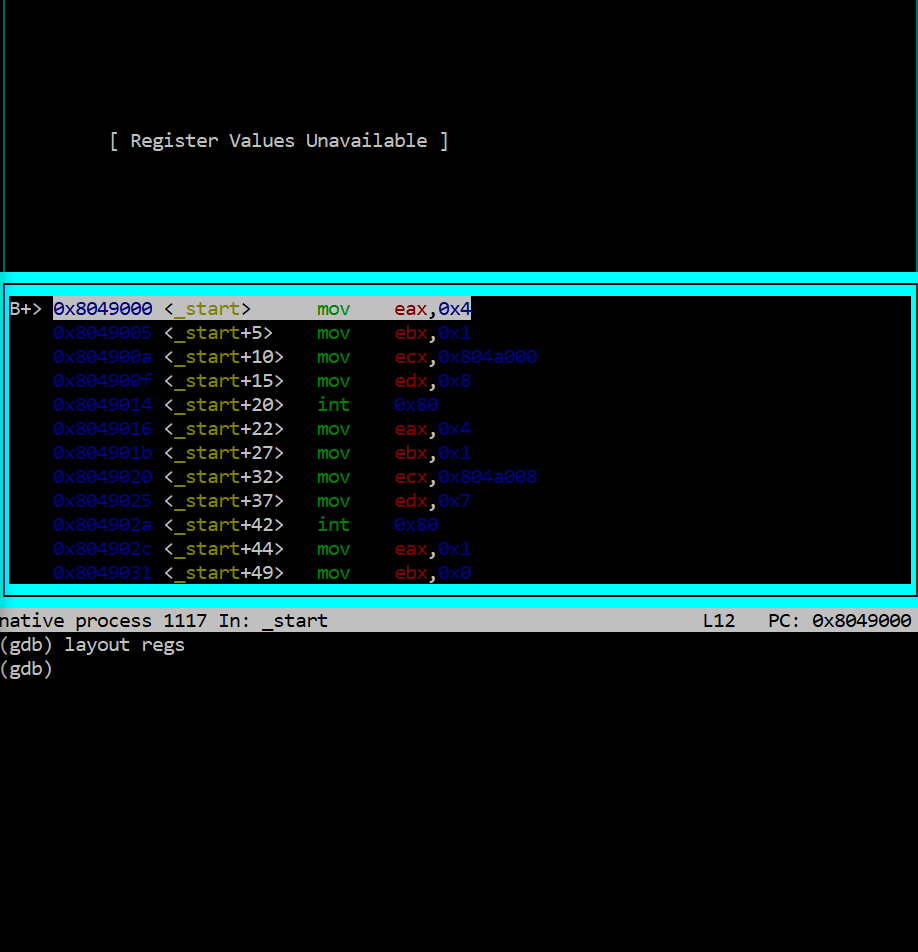
Изучаю диссимилированный код программы с двумя разными синтаксисами.



Восьмой шаг.

Основные различия отображения синтаксиса машинных команд в режимах ATT и Intel заключаются в отображении третьего столбца данных.

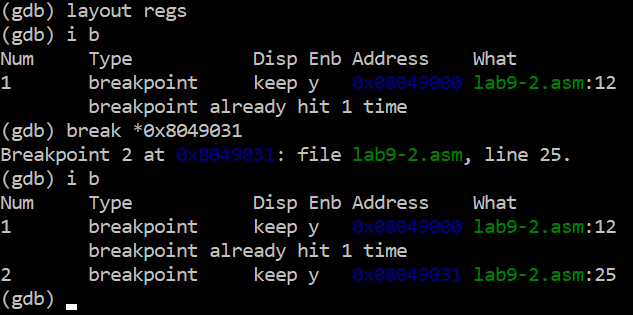
Включаю режим псевдографики.



Девятый шаг.

## 4.3 Добавление точек останова.

Создаю ещё одну точку останова и проверяю информацию о них.

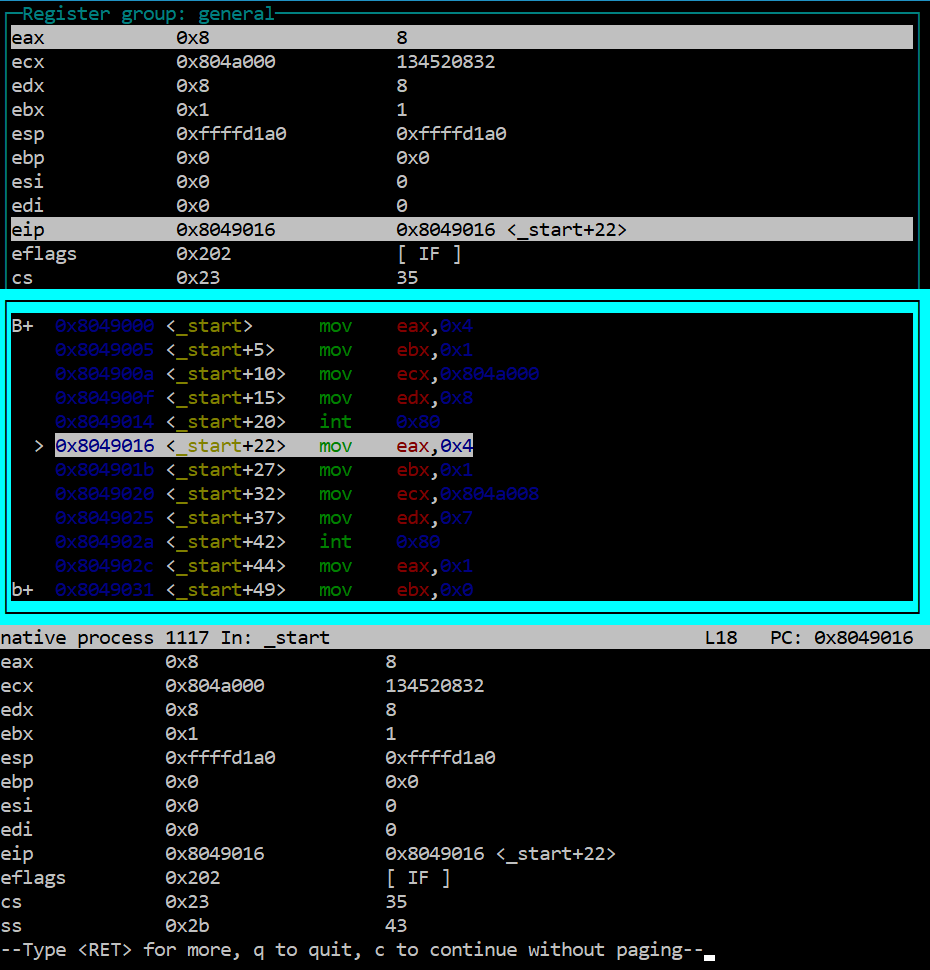


Десятый шаг.

## 4.4 Работа с данными программы в GDB.

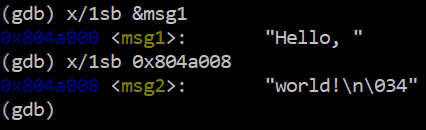
Во время работы программы изменяются значения тех регистров, с которыми она взаимодействует, а также значение регистра eip.

Проверяю информацию о регистрах.



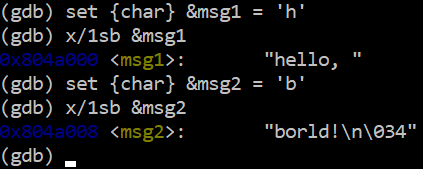
Одиннадцатый шаг.

Просматриваю значение регистров msg1 и msg2.



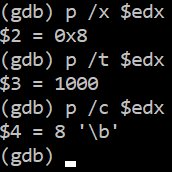
Двенадцатый шаг.

Изменяю несколько символов в msg1 и msg2.



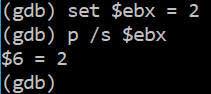
Тринадцатый шаг.

Вывожу значение регистра eax в разных форматах.



Четырнадцатый шаг.

Изменяю значение регистра ebx.

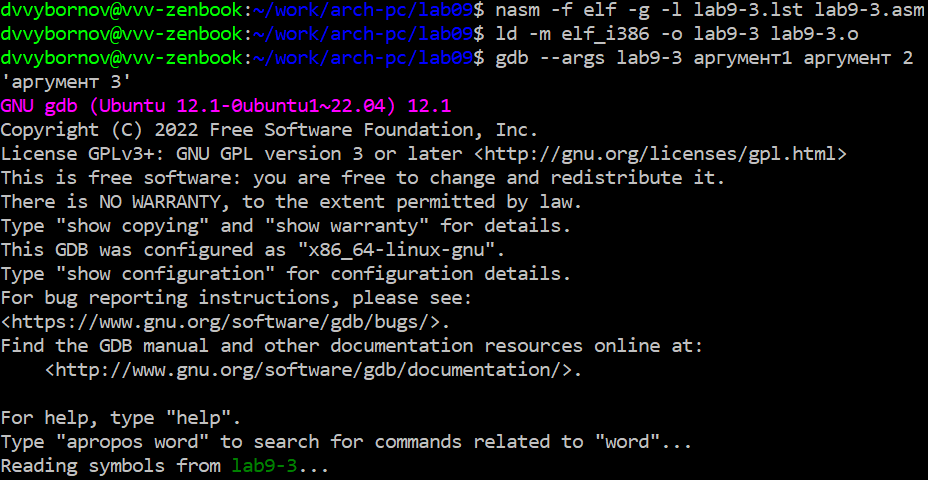


Пятнадцатый шаг.

В первый раз изменение не сработало, так как в ebx хранилось численное значение.

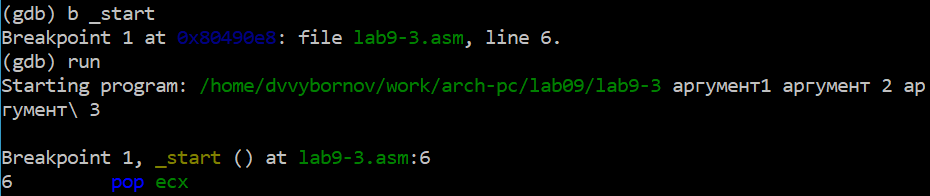
## 4.5 Обработка аргументов командной строки в GDB.

Копирую нужный файл из восьмой лабораторной работы, создаю исполняемый файл и загружаю его в GDB.



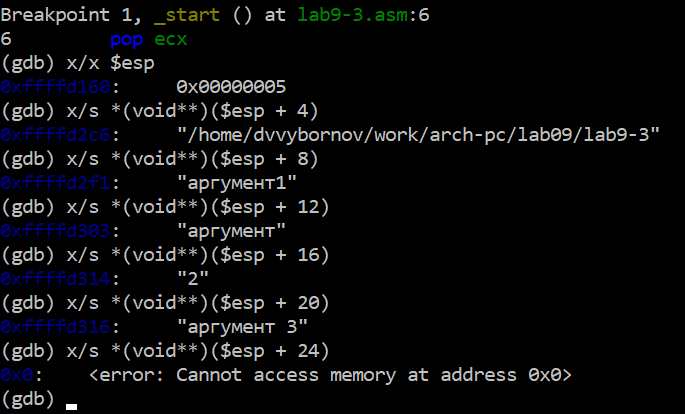
Шестнадцатый шаг.

Запускаю программу.



Семнадцатый шаг.

Просматриваю остальные позиции стека.



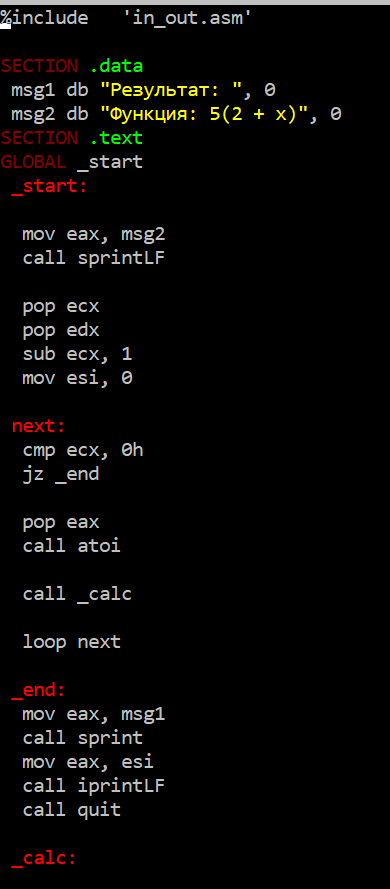
Восемнадцатый шаг.

Шаг изменения адреса равен 4, так как для каждого обьекта в стеке отводится 4 бита.

## 4.6 Задание для самостоятельной работы.

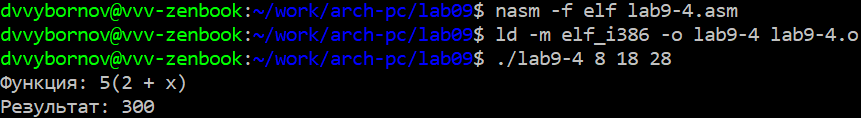
### 4.6.1 №1

Преобразовываю программу из восьмой лабораторной работы.



Первый шаг первого задания.

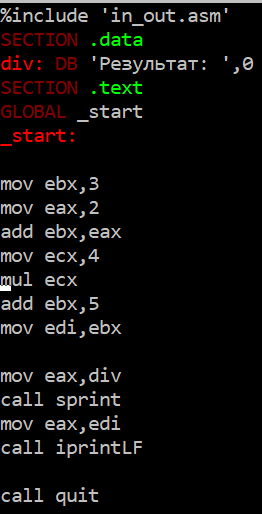
Проверяю работу изменённой программы.



Второй шаг первого задания.

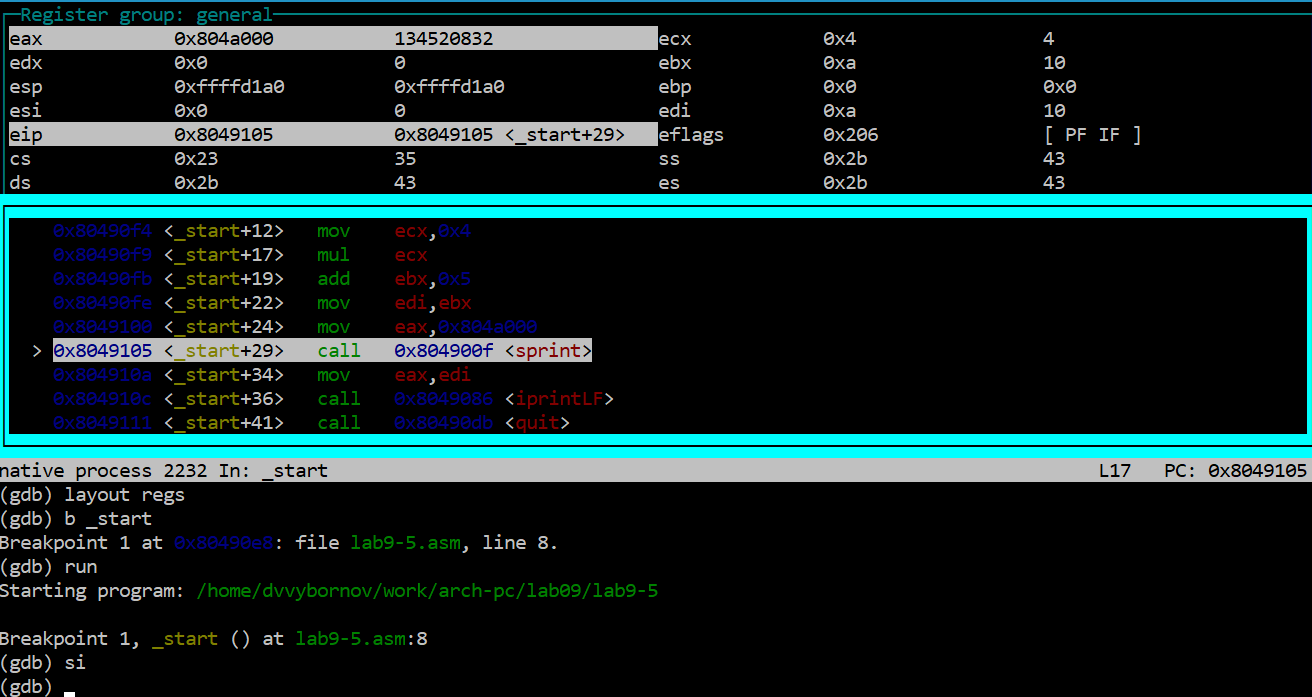
### 4.6.2 №2

Копирую текст нужной программы в новый файл.



Первый шаг второго задания.

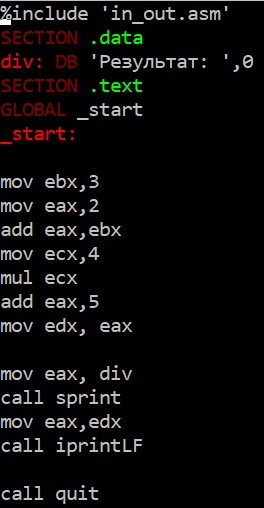
Запускаю эту программу при помощи GDB.



Второй шаг второго задания.

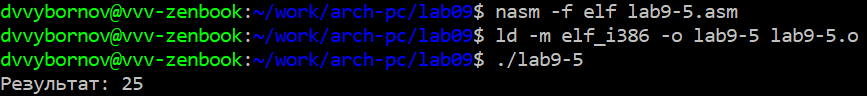
Все проблемы этой программы связаны с тем, как в ней использованы регистры.

Исправляю текст программы.



Третий шаг второго задания.

Проверяю работу программы.



Четвёртый шаг второго задания.

Теперь прогдамма даёт правильный результат.

# 5 Выводы

Выполнив эту лабораторную работу, я приобрёл навыки написания программ с использованием подпрограмм и познакомился с методами отладки при помощи GDB и его основными возможностями.