Отчет по лабораторной работе №9

Понятие подпрограммы. Отладчик GDB.

Паласиос Фелипе

Содержание

# 1 Цель работы

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

# 2 Задание

1. Создайте каталог для выполнения лабораторной работы No 9, перейдите в него и сздайте файл lab09-1.asm
2. 1.Введите в файл lab09-1.asm текст программы из листинга 9.1. Создайте исполняемый файл и проверьте его работу.

2.Замените текст программы, добавив подпрограмму \_subcalcul в подпрограмму \_calcul,для вычисления выражения 𝑓(𝑔(𝑥)), где 𝑥 вводится с клавиатуры, 𝑓(𝑥) = 2𝑥 + 7, 𝑔(𝑥) = 3𝑥 − 1.

1. Создайте файл lab09-2.asm с текстом программы из Листинга 9.2. (Программа печати сообщения Hello world!). 1.Проверьте работу программы, запустив ее в оболочке GDB с помощью команды run (сокращённо r) 2.для более подробного анализа программы установите брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запустите её.

3.Посмотрите дисассимилированный код программы с помощью команды disassemble начиная с метки \_start

4.Переключитесь на отображение команд с Intel’овским синтаксисом, введя команду set disassembly-flavor intel

5.Включите режим псевдографики для более удобного анализа программы

1. Работа с данными программы в GDB

1.Посмотреть содержимое регистров также можно с помощью команды info registers (или i r)

2.Посмотрите значение переменной msg1 по имени (gdb) x/1sb &msg1 0x804a000 : “Hello,”

3.Посмотрите дисассимилированный код программы с помощью команды disassemble начиная с метки \_start

4.С помощью команды set измените значение регистра ebx

1. Скопируйте файл lab8-2.asm, созданный при выполнении лабораторной работы No8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем lab09-3.asm. Создайте исполняемый файл. Для загрузки в gdb программы с аргументами необходимо использовать ключ –args. Загрузите исполняемый файл в отладчик, указав аргументы

1.Исследуем расположение аргументов командной строки в стеке после запуска программы с помощью gdb. Для начала установим точку останова перед первой инструкцией в программе и запустим ее. (gdb) b \_start (gdb) run

2.Адрес вершины стека храниться в регистре esp и по этому адресу располагается число равное количеству аргументов командной строки (включая имя программы) Посмотрите остальные позиции стека – по адесу [esp+4] располагается адрес в памяти где находиться имя программы, по адесу [esp+8] храниться адрес первого аргумента, поаресу [esp+12] – второго и т.д

1. Задание для самостоятельной работы

1.Преобразуйте программу из лабораторной работы No8 (Задание No1 для самостоятельной работы), реализовав вычисление значения функции 𝑓(𝑥) как подпрограмму

2.В листинге 9.3 приведена программа вычисления выражения (3 + 2) ∗ 4 + 5. При запуске данная программа дает неверный результат. Проверьте это. С помощью отладчика GDB, анализируя изменения значений регистров, определите ошибку и исправьте ее.

# 3 Выполнение лабораторной работы

1. Создайте каталог для выполнения лабораторной работы No 9, перейдите в него и сздайте файл lab09-1.asm (рис. [1](#fig:001)).

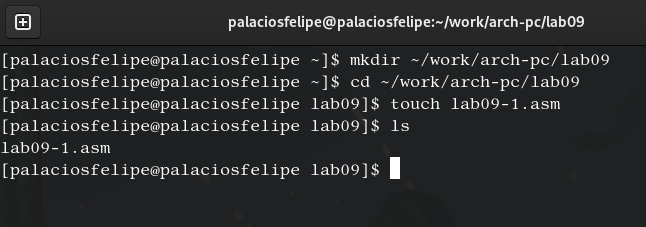


Figure 1: каталог

1. 1.Введите в файл lab09-1.asm текст программы из листинга 9.1. Создайте исполняемый файл и проверьте его работу (рис. [2](#fig:002)).

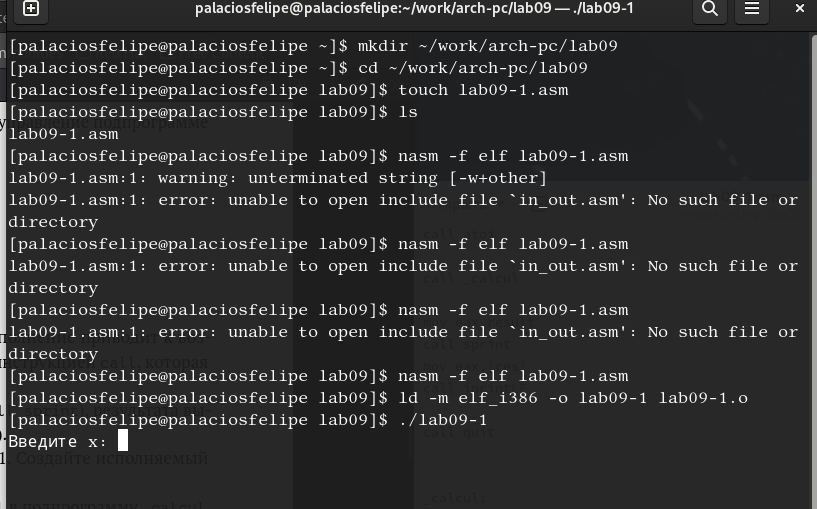


Figure 2: исполняемый файл

2.Измените текст программы, добавив подпрограмму \_subcalcul в подпрограмму \_calcul,для вычисления выражения 𝑓(𝑔(𝑥)), где 𝑥 вводится с клавиатуры, 𝑓(𝑥) = 2𝑥 + 7, 𝑔(𝑥) = 3𝑥 − 1 (рис. @fig:003).

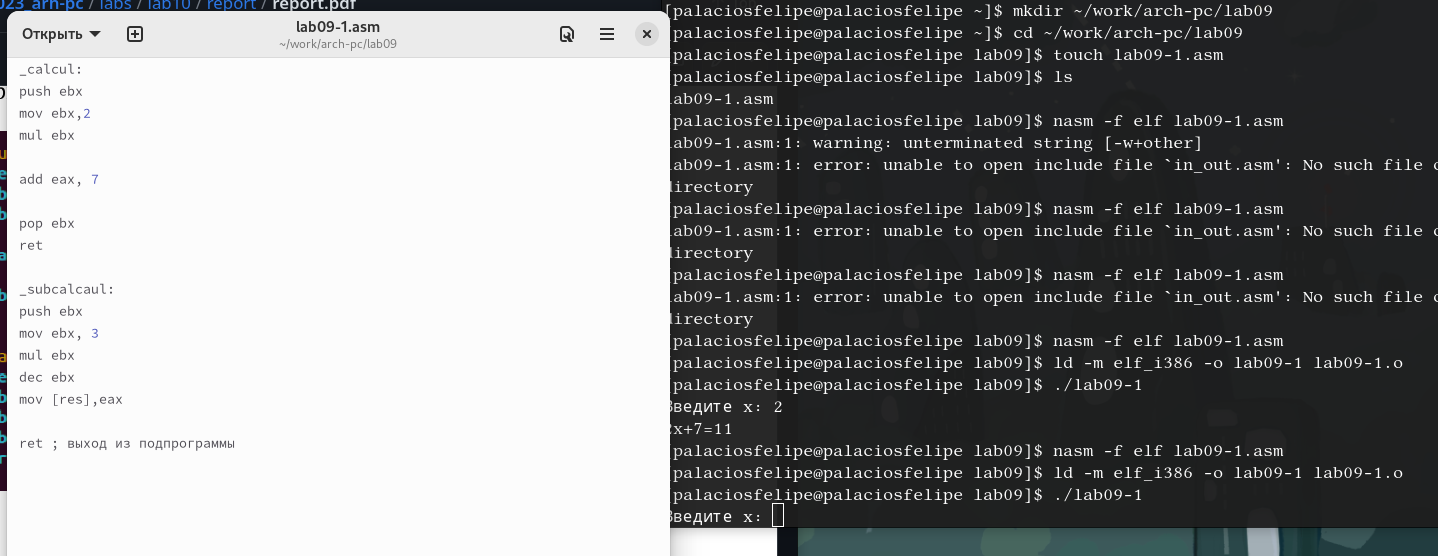


Figure 3: добавив подпрограмму \_subcalcul

1. Создайте файл lab09-2.asm с текстом программы из Листинга 9.2. (Программа печати сообщения Hello world!) (рис. [4](#fig:004)).

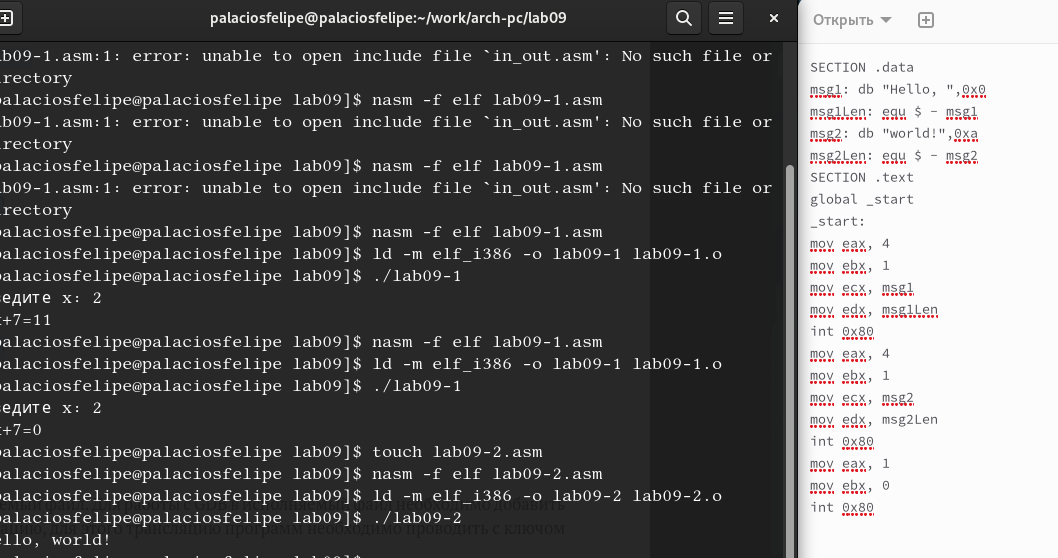


Figure 4: lab09-2

1.Проверьте работу программы, запустив ее в оболочке GDB с помощью команды run (сокращённо r) (рис. [5](#fig:005)).

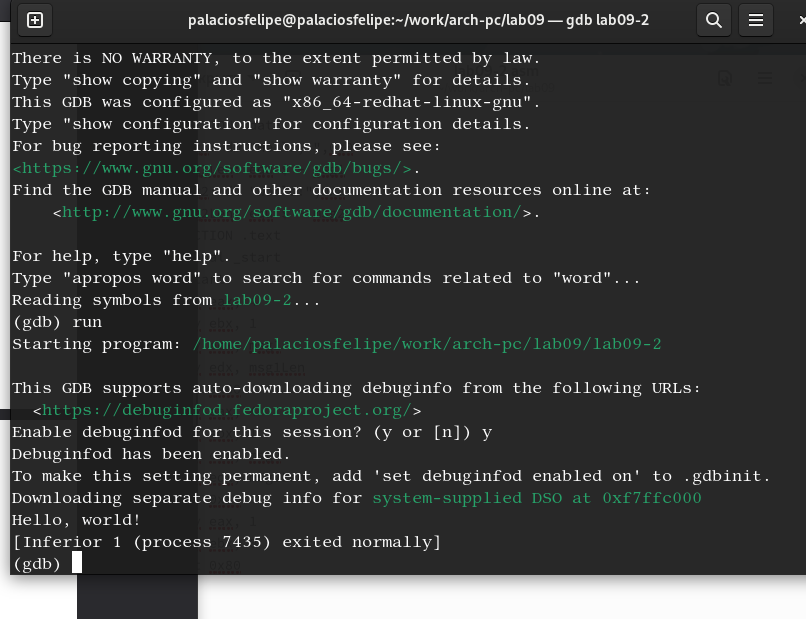


Figure 5: lab09-2. run

2.для более подробного анализа программы установите брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запустите её (рис. [6](#fig:006)).

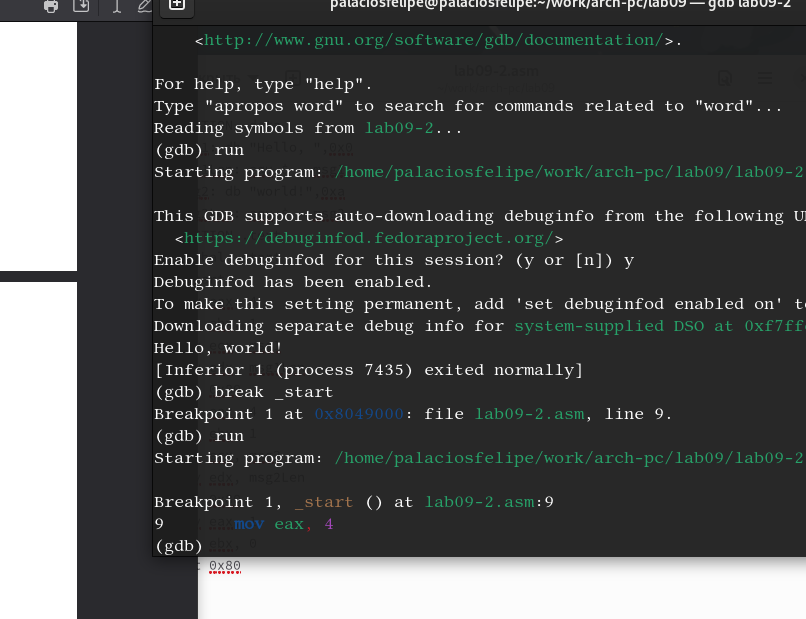


Figure 6: \_start

3.Посмотрите дисассимилированный код программы с помощью команды disassemble начиная с метки \_start (рис. [7](#fig:007)).

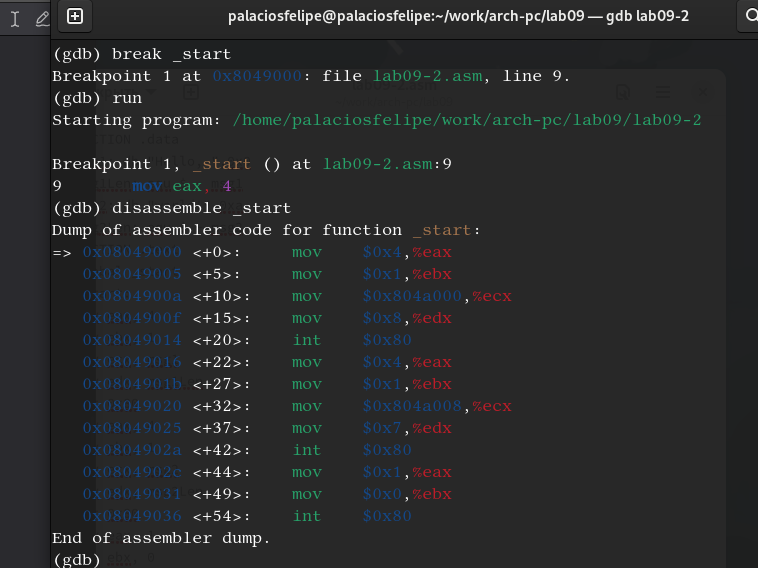


Figure 7: disassemble \_\_start

4.Переключитесь на отображение команд с Intel’овским синтаксисом, введя команду set disassembly-flavor intel (рис. [8](#fig:008)).

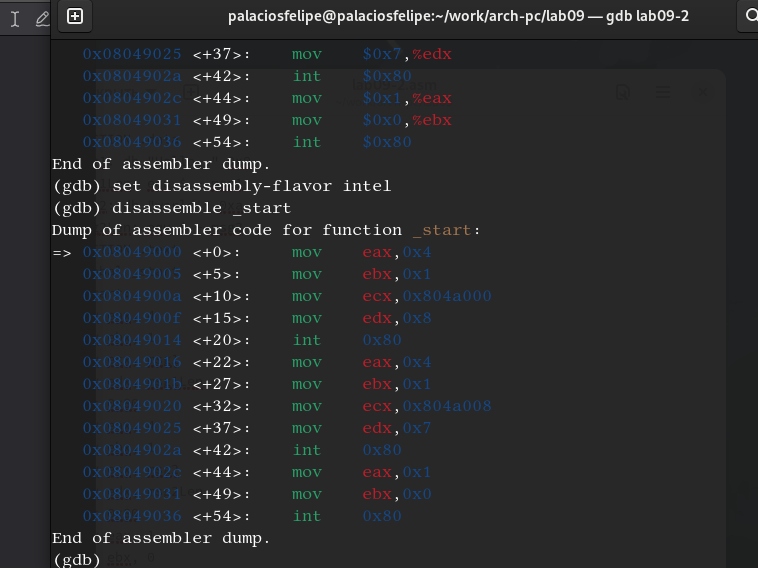


Figure 8: Название рисунка

5.Включите режим псевдографики для более удобного анализа программы (рис. [9](#fig:009)).

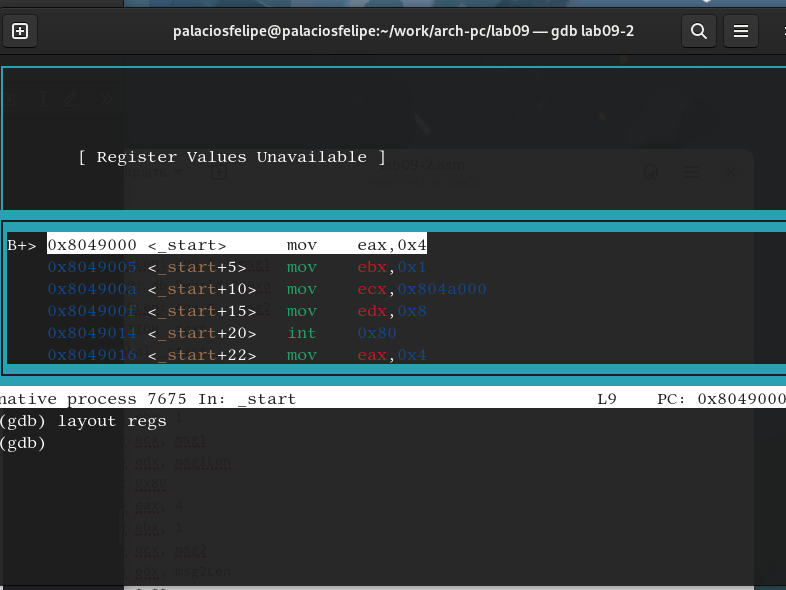


Figure 9: layout

1. Работа с данными программы в GDB

1.Посмотреть содержимое регистров также можно с помощью команды info registers (или i r) (рис. [10](#fig:010)).

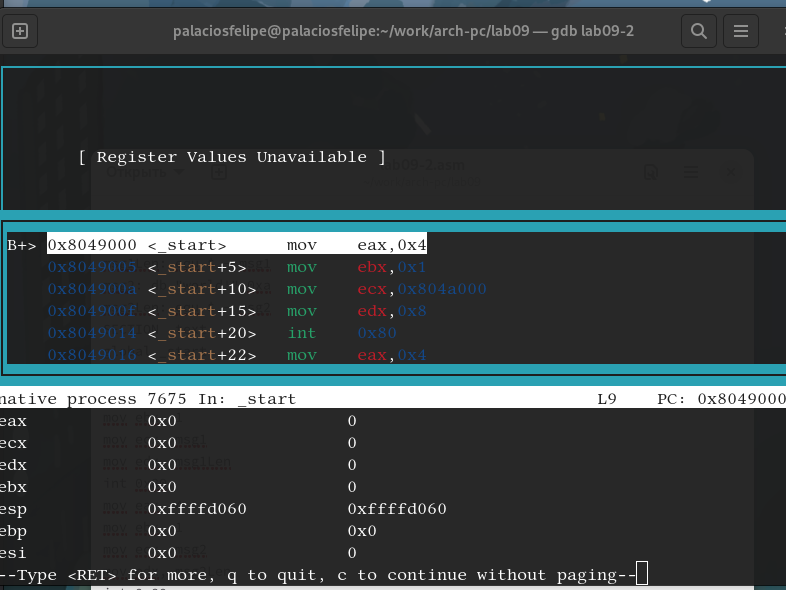


Figure 10: содержимое регистров

2.Посмотрите значение переменной msg1 по имени (gdb) x/1sb &msg1 0x804a000 : “Hello,” (рис. [11](#fig:011)).

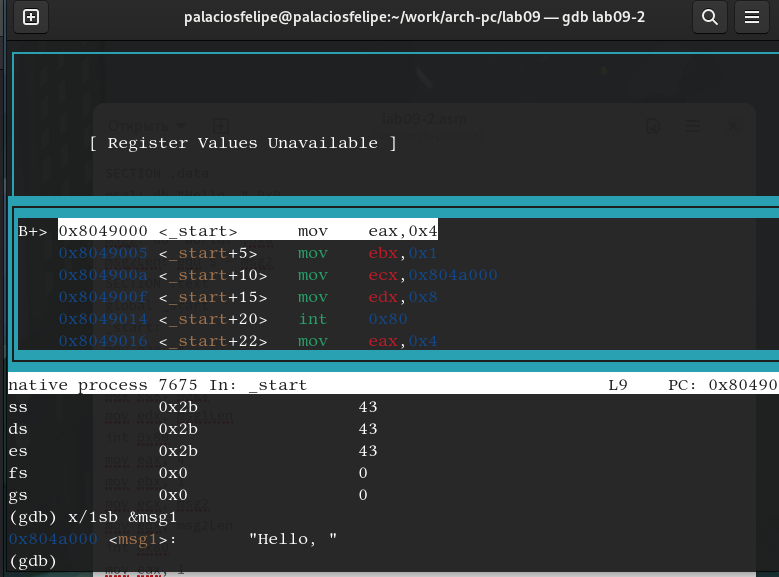


Figure 11: значение переменной msg1 по имени

3.Измените первый символ переменной msg1 (рис. [12](#fig:012)).

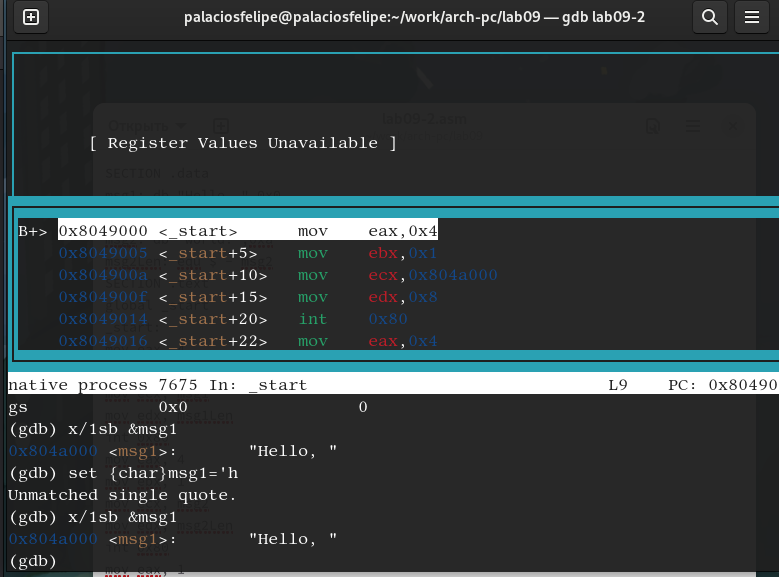


Figure 12: Изменен первый символ переменной msg1

4.С помощью команды set измените значение регистра ebx (рис. [13](#fig:013)).

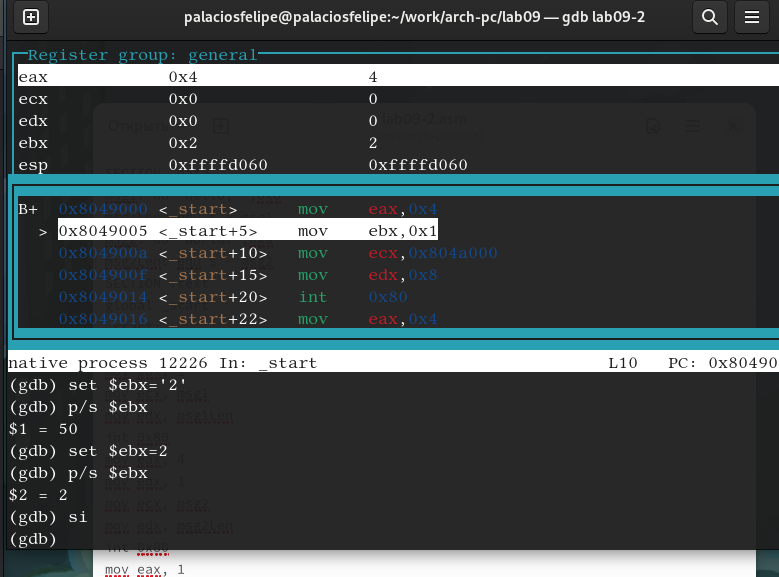


Figure 13: команда set

5.Завершите выполнение программы с помощью команды continue (сокращенно c) или stepi (сокращенно si) и выйдите из GDB с помощью команды quit (сокращенно q). (рис. [14](#fig:014)).

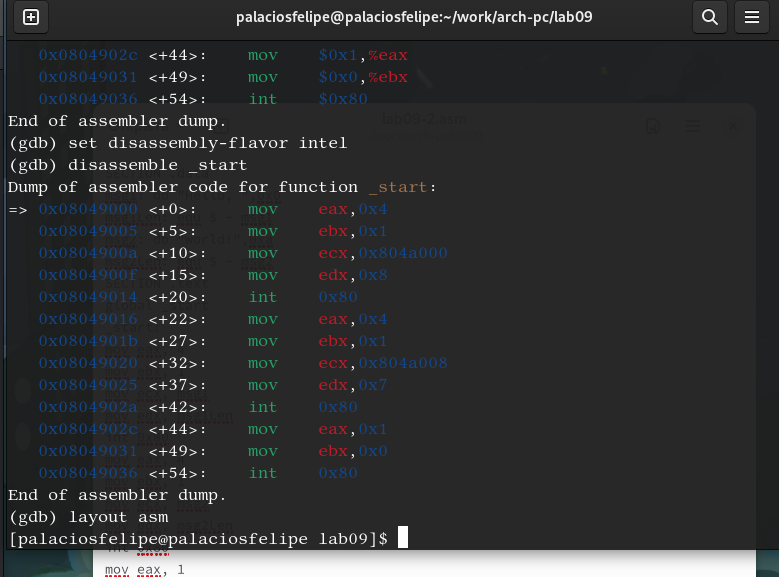


Figure 14: quit

1. Скопируйте файл lab8-2.asm, созданный при выполнении лабораторной работы No8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем lab09-3.asm. Создайте исполняемый файл. Для загрузки в gdb программы с аргументами необходимо использовать ключ –args. Загрузите исполняемый файл в отладчик, указав аргументы (рис. [15](#fig:015)).

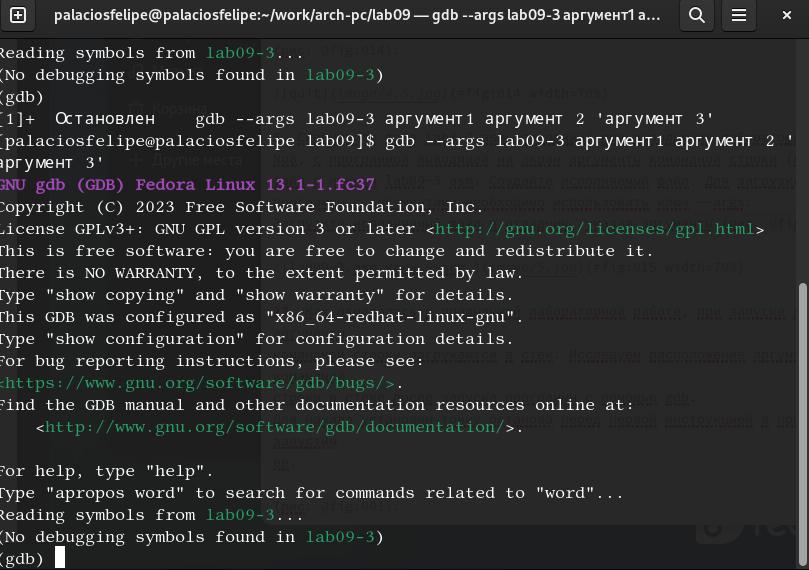


Figure 15: lab09-3.asm.ключ –arg

1.Исследуем расположение аргументов командной строки в стеке после запуска программы с помощью gdb.Для начала установим точку останова перед первой инструкцией в программе и запустим ее (gdb) b \_start (gdb) run (рис. [16](#fig:016)).

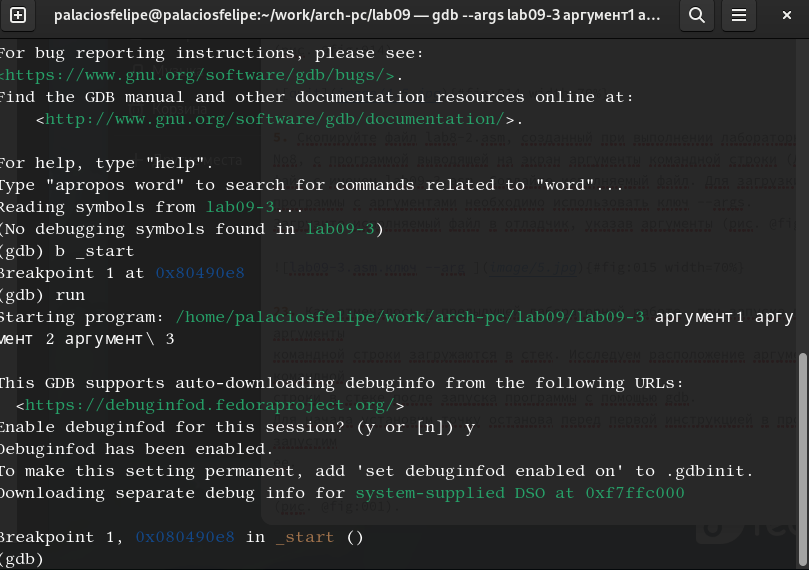


Figure 16: установим точку

2.Адрес вершины стека храниться в регистре esp и по этому адресу располагается число равное количеству аргументов командной строки (включая имя программы) Посмотрите остальные позиции стека – по адесу [esp+4] располагается адрес в памяти где находиться имя программы, по адесу [esp+8] храниться адрес первого аргумента, поаресу [esp+12] – второго и т.д (рис. [17](#fig:017)).

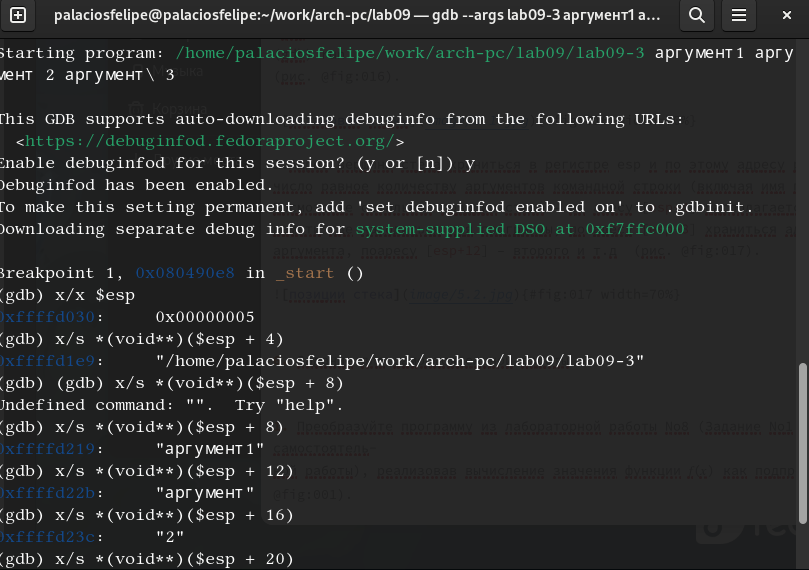


Figure 17: позиции стека

1. Задание для самостоятельной работы

1.Преобразуйте программу из лабораторной работы No8 (Задание No1 для самостоятельной работы), реализовав вычисление значения функции 𝑓(𝑥) как подпрограмму (рис. [18](#fig:018)).

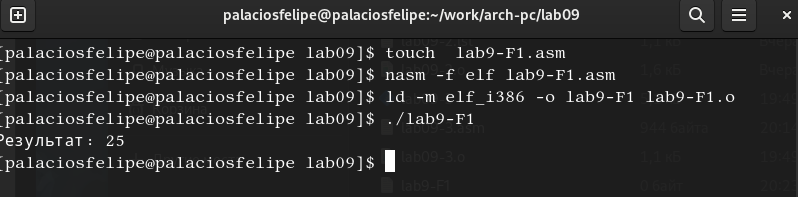


Figure 18: 1

2.В листинге 9.3 приведена программа вычисления выражения (3 + 2) ∗ 4 + 5. При запуске данная программа дает неверный результат. Проверьте это. С помощью отладчика GDB, анализируя изменения значений регистров, определите ошибку и исправьте ее (рис. [19](#fig:019)).

Figure 19: 2

Figure 19: 2

# 4 Выводы

Приобретены навыки написания программ с использованием подпрограмм. Ознакомлен с методами отладки при помощи GDB и его основными возможностями.

# Список литературы