Отчёт по лабораторной работе №9

По теме: Понятие подпрограммы. Отладчик GDB.

Выполнил: Пателепень Филипп Максимович, НММбд-04-24.

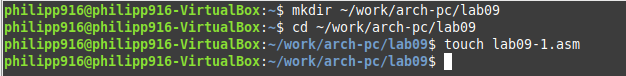
# 9.1. Цель работы

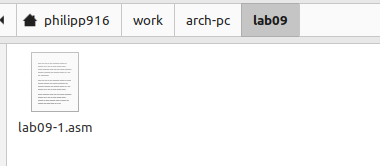
Приобретение навыков написания программ с использованием подпрограмм. Знакомство с методами отладки при помощи GDB и его основными возможностями.

# 9.4. Ход выполнения лабораторной работы

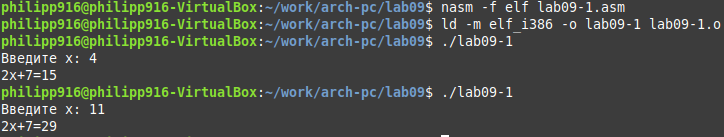
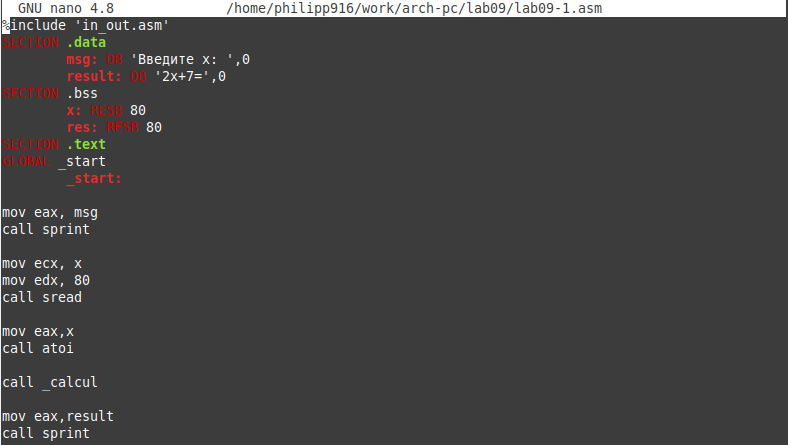
9.4.1. Реализация подпрограмм в NASM

1. Я создал каталог для выполнения лабораторной работы № 9, перешёл в него и создал файл lab09-1.asm:

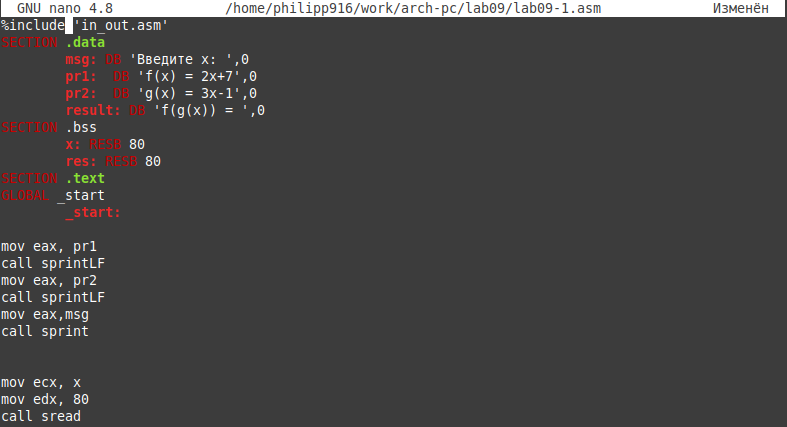
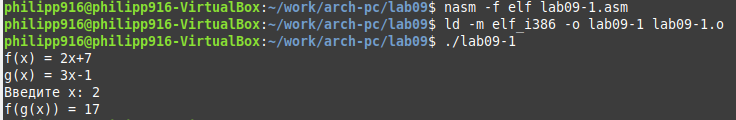




1. Я ввел в файл lab09-1.asm текст программы из листинга 9.1. Создал исполняемый файл и проверьте его работу:

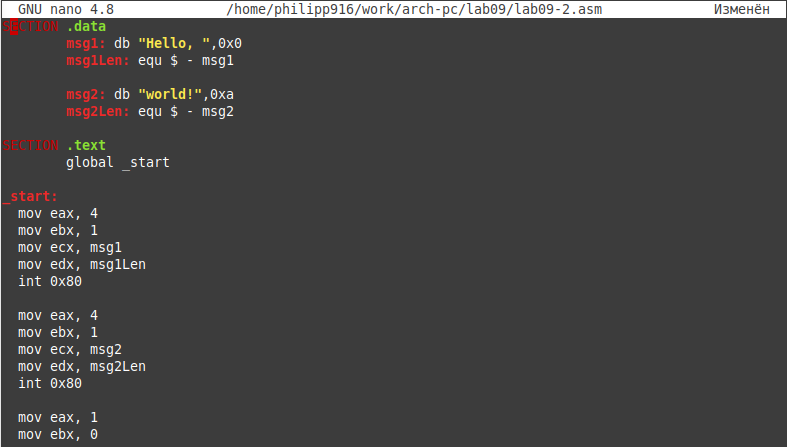


1. Я изменил текст программы, добавив подпрограмму \_subcalcul в подпрограмму \_calcul, для вычисления выражения 𝑓(𝑔(𝑥)), где 𝑥 вводится с клавиатуры, 𝑓(𝑥) = 2𝑥 + 7, 𝑔(𝑥) = 3𝑥 − 1. Т.е. 𝑥 передается в подпрограмму \_calcul из нее в подпрограмму \_subcalcul, где вычисляется выражение 𝑔(𝑥), результат возвращается в \_calcul и вычисляется выражение 𝑓(𝑔(𝑥)). Результат возвращается в основную программу для вывода результата на экран:

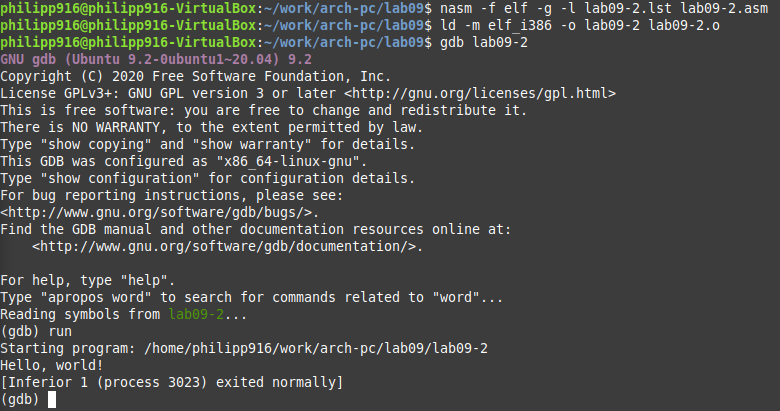
https://lh7-rt.googleusercontent.com/docsz/AD_4nXdnStvf8MSNQ8wzK1n548Wb5fJqE9sq7UzlXKVDe1KmN_qTvxs8v4SLMgoiDACeq_iPmA96R7n2PaH5geHMRBRiUP8sZG9NaNlEs95xnaKhZ4DvgKC6BZGGTChOriCZKlVyYgk?key=W7mmvdQShFn4usySfJ5O_tqV

9.4.2. Отладка программ с помощью GDB

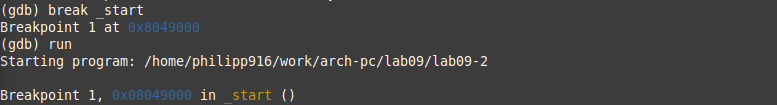
1. Я Создал файл lab09-2.asm с текстом программы из Листинга 9.2.:

https://lh7-rt.googleusercontent.com/docsz/AD_4nXfp0nK2sxw3V71cxTrDA9AuHaN2zz2u_myKklF_KHOzoSMw5GforkRvo70ovljS_3-aFQNPIc8pHFaaDYfHhBF8w7Sj0xB-iWEffybmJ0U1BnYcZ-5T3X29zp1q7SPZrbYbdKVc?key=W7mmvdQShFn4usySfJ5O_tqVhttps://lh7-rt.googleusercontent.com/docsz/AD_4nXfazfrePWENNYd1I4ak2aAb6xvCHh6p-bGA-KuCtooLV9H9QDFInkCZvC534OvQ7Cnt777juAKMWmlQawQO6u4AHN66sEQ2DbxV3YZ9_1SzHUeXqdn4d9dhmoZrrsy37rCKDKo?key=W7mmvdQShFn4usySfJ5O_tqV

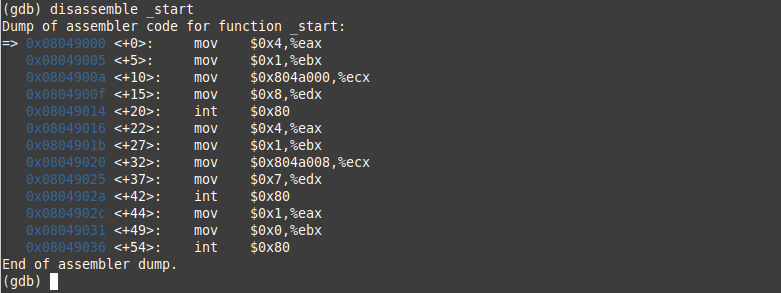
1. Я получил исполняемый файл. Для работы с GDB в исполняемый файл необходимо добавить отладочную информацию, для этого трансляцию программ необходимо проводить с ключом ‘-g’. Я загрузил исполняемый файл в отладчик gdb и проверил работу программы, запустив ее в оболочке GDB с помощью команды run (сокращённо r):



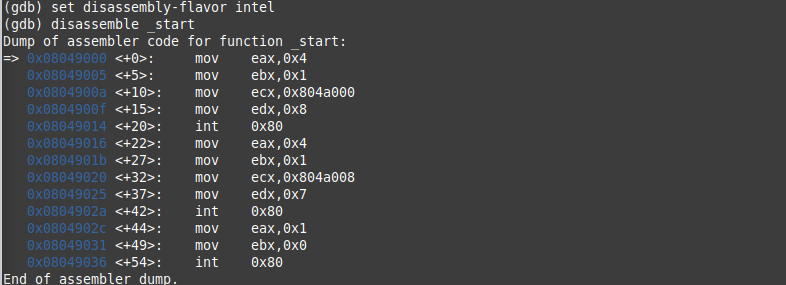
1. Для более подробного анализа программы установил брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запустил её:



1. Далее я внимательно просмотрел дисассимилированный код программы с помощью команды disassemble начиная с метки \_start:

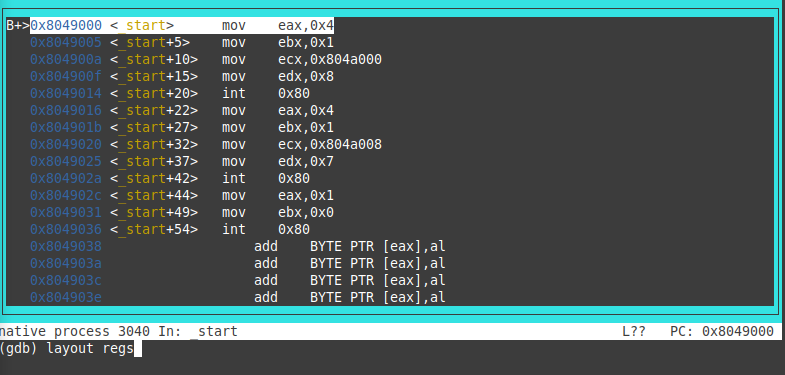


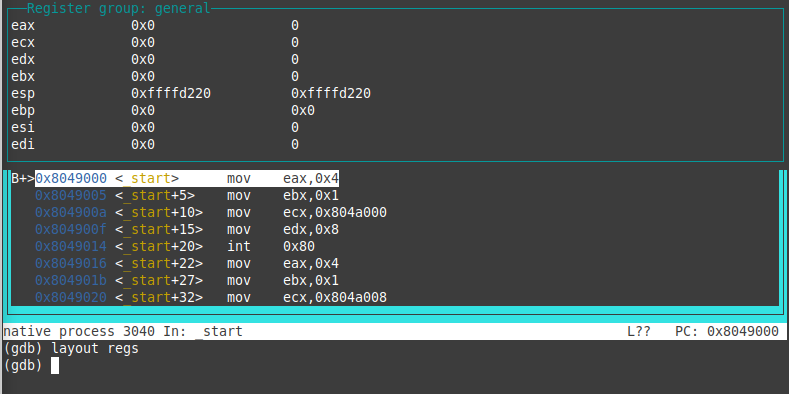
1. Далее с помощью специальных команд я переключился на intel’овское отображение синтаксиса:



Отличие заключается в том, что в диссамилированном отображении в командах используют символы ‘%’ и ‘$’, а в Intel’овском отображении эти символы отсутствуют.

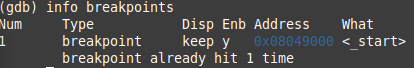
1. Следующим шагом я включил режим псевдографики для более удобного анализа программы:

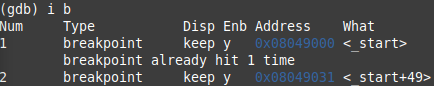




9.4.2.1. Добавление точек останова(метки)

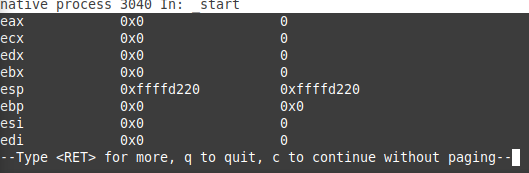
1. Я посмотрел наличие метки с помощью команды info breakpoints, а также установил еще одну метку по адресу инструкции:



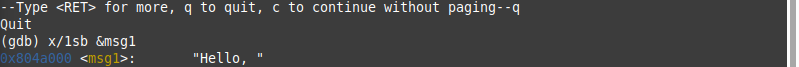


9.4.2.2. Работа с данными программы в GDB

1. С помощью команды info registers (i r) я посмотрел значение регистров:



1. Далее я посмотрел значение переменной msg1 по имени:



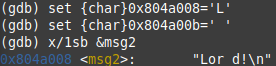
1. Посмотрел значение второй переменной msg2 по адресу:

https://lh7-rt.googleusercontent.com/docsz/AD_4nXe-D0RxMnG9fN_72VMLfYhwWiLA3xJbOOaqbWxEjurPmohEAt6Uz1XQkeBkYo-GHbyB3rak-VeNPcjbT0Kpt0qkVCG4yjbd78oDMqmscggc9wX4u9eydmadekJZ3xznlHh5Zore?key=W7mmvdQShFn4usySfJ5O_tqV

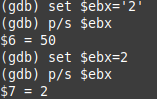
1. С помощью команды set я изменил первый символ переменной msg1:



1. Аналогично изменил переменную msg2:

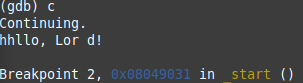


1. Я изменил значение регистра ebx с помощью команды set:



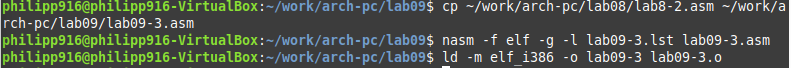
Команда выводит различные значения т.к. в первый раз мы вносим в регистр значение 2, а во втором случае присваиваем регистру значение 2

1. Я завершил работу с файлом в отладчике с помощью комманд ‘c’, ‘si’ и ‘quit’:

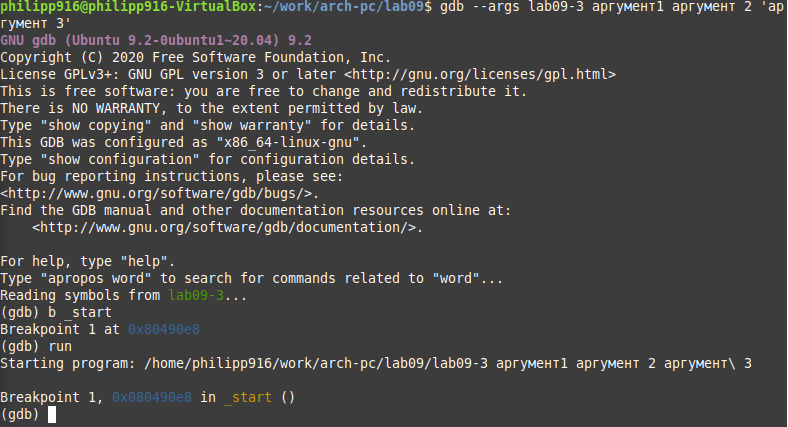


9.4.2.3. Обработка аргументов командной строки в GDB

1. Я скопировал файл lab8-2.asm, созданный при выполнении лабораторной работы №8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем lab09-3.asm, а также создал исполняемый файл:



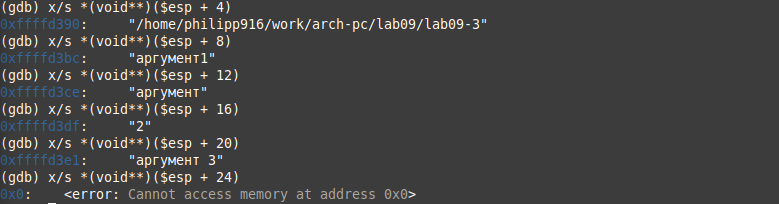
1. Далее я загрузил исполняемый файл в отладчик, указав аргументы: gdb --args lab09-3 аргумент1 аргумент 2 'аргумент 3' . В отладчике поставил метку на \_start и запустил программу:



1. Я проверил адрес вершины стека и убедился, что там хранится 5 элементов:

https://lh7-rt.googleusercontent.com/docsz/AD_4nXd-7dMsxpGAwmafmioUPLr5L7RzWKbija64ZhlsWsk3fiMwrK7XIXSp177kWrOpOhTfXB1Q_KUSC4FALXXbokPHO4J6ur445rCYpGf_84yngTHlYKRk56tcLszbuQOnIOhVOQM?key=W7mmvdQShFn4usySfJ5O_tqV

1. Затем я посмотрел все позиции стека:



По первому адресу хранится сам адрес, в остальных адресах, в свою очередь, хранятся элементы. Элементы расположены с интервалом в 4 единицы, так как стек может хранить до 4 байт. Для того чтобы данные сохранялись правильно, компьютер использует новый стек для новой информации.

# 9.5. Выполнение самостоятельной работы

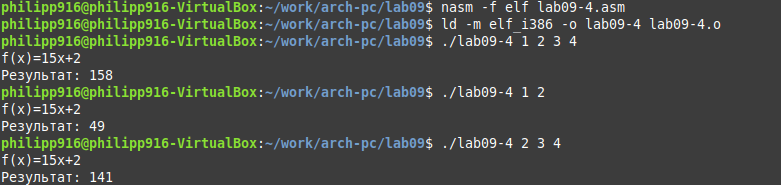
1. Сначала я создал файл для выполнения первого задания самостоятельной работы под названием lab09-4.asm:

https://lh7-rt.googleusercontent.com/docsz/AD_4nXfqddKtKv1FIRhgRoKODzMiQq8Q899bzjGkQ0KdMTfQ37PldWH0OVbpyK1v2TFiFTXI3rzDHF36mHs5-inrT0NB-6fCwCOxiJBZkOGphc8L-nFt3tK3J4_4Tdpv-HJNvbSqPys?key=W7mmvdQShFn4usySfJ5O_tqV

1. Затем преобразовал свой код программы из лабораторной работы №8 и реализовал вычисления как подпрограмму:



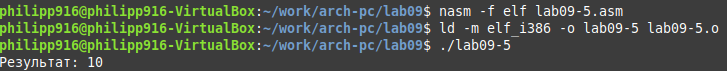
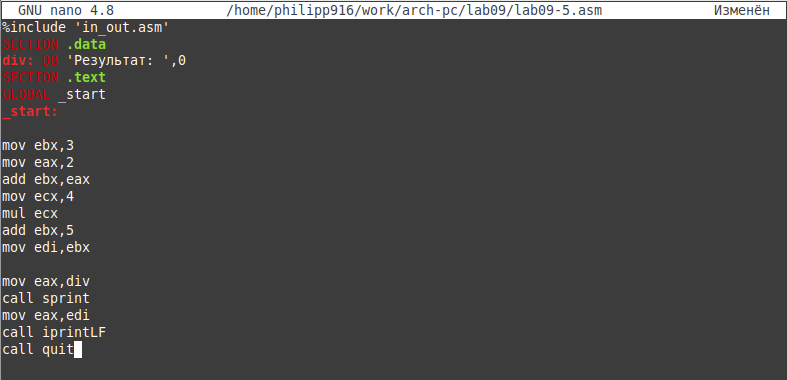
1. Создал исполняемый файл и запустил его, чтобы проверить правильность выполнения программы. Программа работает исправно:



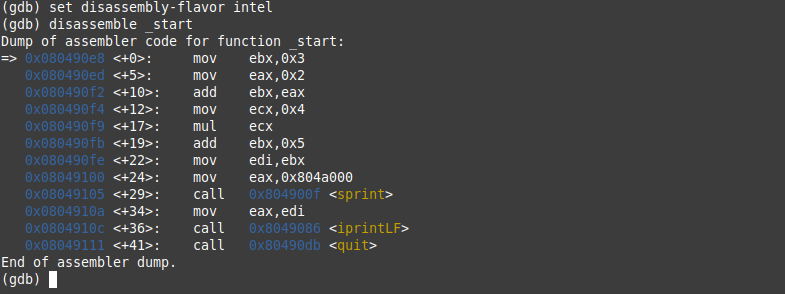
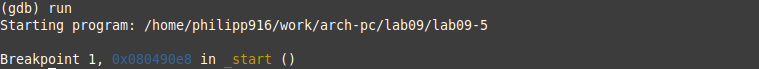
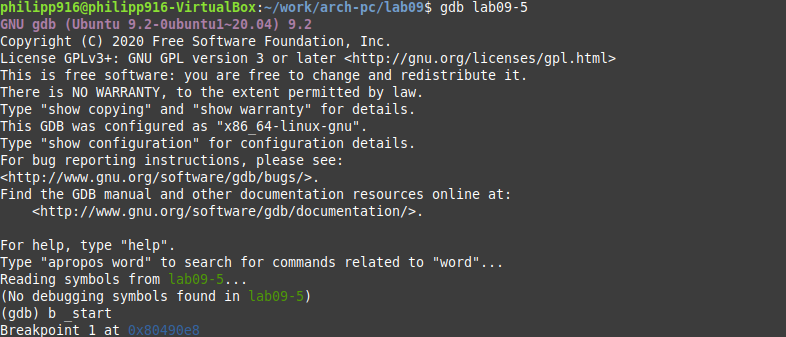
1. Для выполнения второго задания я создал файл lab09-5.asm:

https://lh7-rt.googleusercontent.com/docsz/AD_4nXfaIiXd1K5qxT2NnaqyumEUQvVBTy1KcWT-j4Rj_ZT1evyJ4UVw-p6QZs8le0X-tN20oZiUJpSunSOebBGNF3dM7ec205_zzrmk6ohM3znWk8W_1DGBIU-wnWLsMRjMROsLGokm?key=W7mmvdQShFn4usySfJ5O_tqV

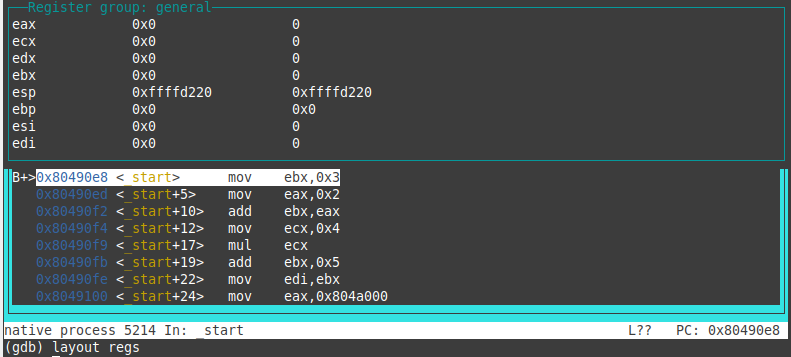
1. С помощью листинга 9.3 я написал код программы. Далее создал исполняемый файл и запустил его:



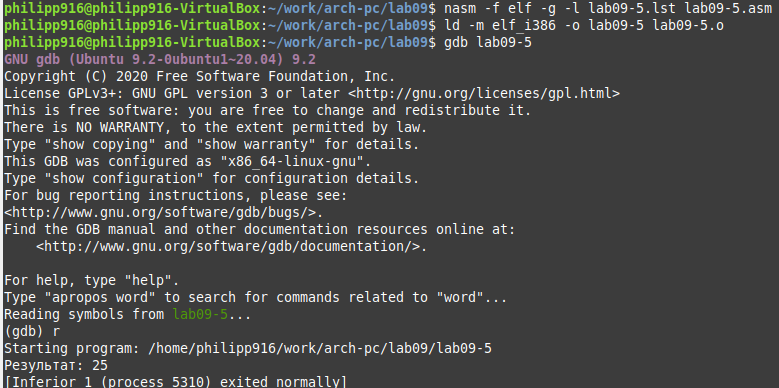
1. Далее я выявил ошибку, связанную с неправильным вычислением программы, запустил программу в отладчике:



1. Я открыл регистры и стал внимательно их изучать. Заметил, что не все регистры стоят на своих местах:



1. После изменения регистров я запустил программу. Программа стала работать корректно и вывела в терминале ответ 25:



# Вывод

Вывод: я приобрел навыки написания программ с использованием подпрограмм. Познакомился с методами отладки при помощи GDB и его основными возможностями.

# Список литературы

1. GDB: The GNU Project Debugger. — URL: https://www.gnu.org/software/gdb/.

2. GNU Bash Manual. — 2016. — URL: https://www.gnu.org/software/bash/manual/.

3. Midnight Commander Development Center. — 2021. — URL: https://midnight-commander.org/.

4. NASM Assembly Language Tutorials. — 2021. — URL: https://asmtutor.com/.

5. *Newham C.* Learning the bash Shell: Unix Shell Programming. — O’Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658.

6. *Robbins A.* Bash Pocket Reference. — O’Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.

7. The NASM documentation. — 2021. — URL: https://www.nasm.us/docs.php.

8. *Zarrelli G.* Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.

9. *Колдаев В. Д.*, *Лупин С. А.* Архитектура ЭВМ. — М. : Форум, 2018.

10. *Куляс О. Л.*, *Никитин К. А.* Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.

11. *Новожилов О. П.* Архитектура ЭВМ и систем. — М. : Юрайт, 2016.

12. Расширенный ассемблер: NASM. — 2021. — URL: https://www.opennet.ru/docs/RUS/nasm/.

13. *Робачевский А.*, *Немнюгин С.*, *Стесик О.* Операционная система UNIX. — 2-е изд. — БХВ-Петербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.

14. *Столяров А.* Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm\_unix.

15. *Таненбаум Э.* Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).

16. *Таненбаум Э.*, *Бос Х.* Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).