Отчет по лабораторной работе №9

По теме: Понятие подпрограммы. Отладчик GDB

Выполнил: Чубаев Кирилл Евгеньевич, НММбд-04-24

**Содержание**

[Цель работы 1](#_Toc183640190)

[Ход выполнения лабораторной работы 1](#_Toc183640191)

[Выполнение самостоятельной работы 12](#_Toc183640192)

[Вывод 18](#_Toc183640193)

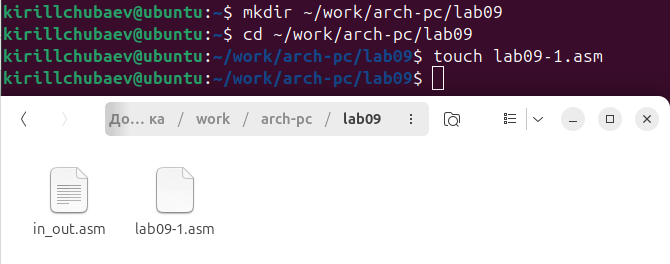
[Список литературы 18](#_Toc183640194)

# Цель работы

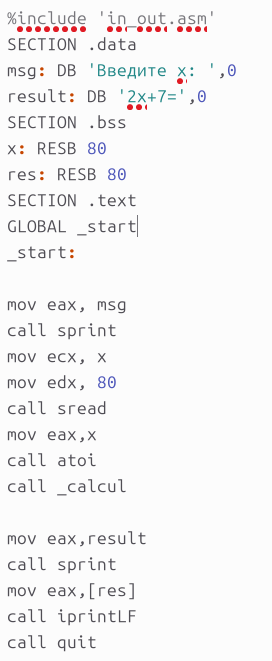
Целью данной лабораторной работы является приобретение навыков написания программ с использованием подпрограмм, а также знакомство с методами отладки при помощи GDB и его основными возможностями

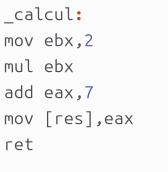
# Ход выполнения лабораторной работы

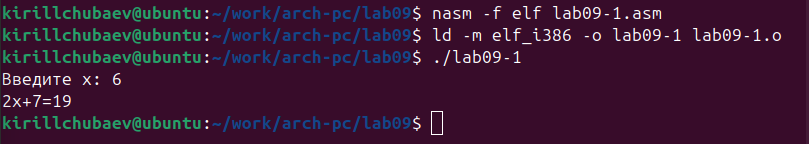
1. Сначала я создал каталог lab09 и создал файл lab09-1.asm:



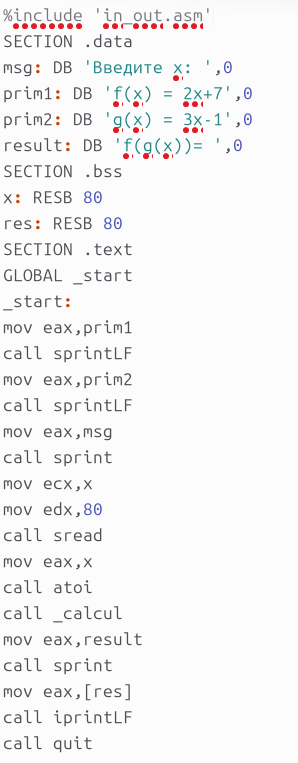
1. Я ввел код программы из листинга 9.1 в созданный файл и запустил программу:

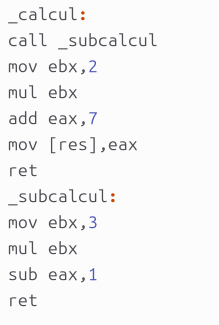


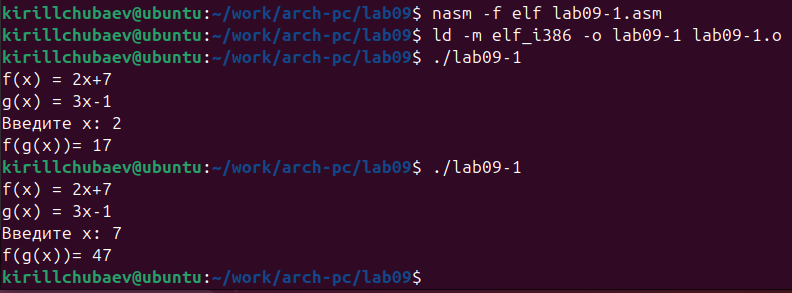




1. Далее я изменил код программы так, чтобы она решала выражение f(g(x)). Программа работает корректно:





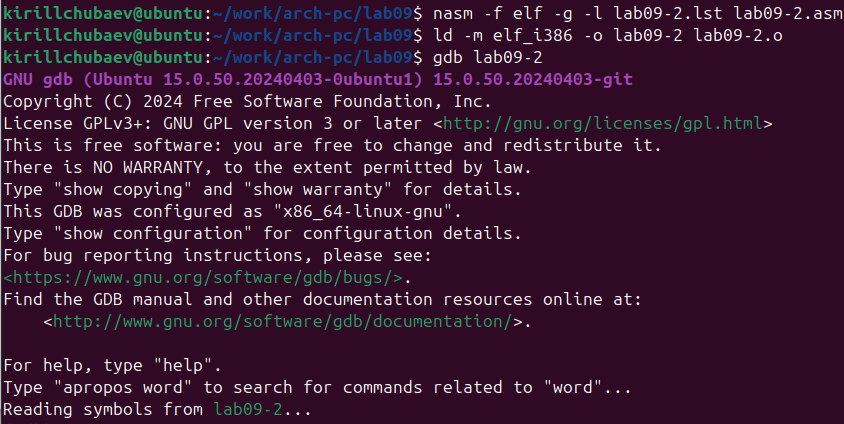


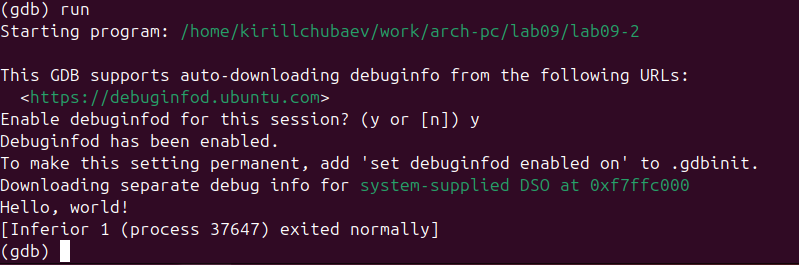
1. Потом я создал файл lab09-2.asm и вписал туда код программы с помощью листинга 9.2:

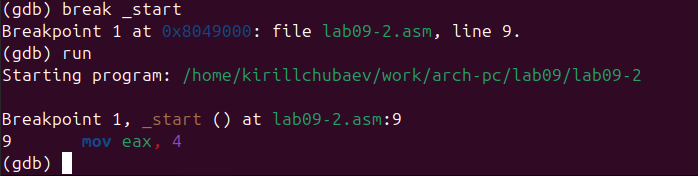




1. Затем загрузил и запустил файл программы в отладчик gdb и поставил break-отметку на метку \_start:







1. Я внимательно изучил дисассимплированный код программы, начиная с метки:

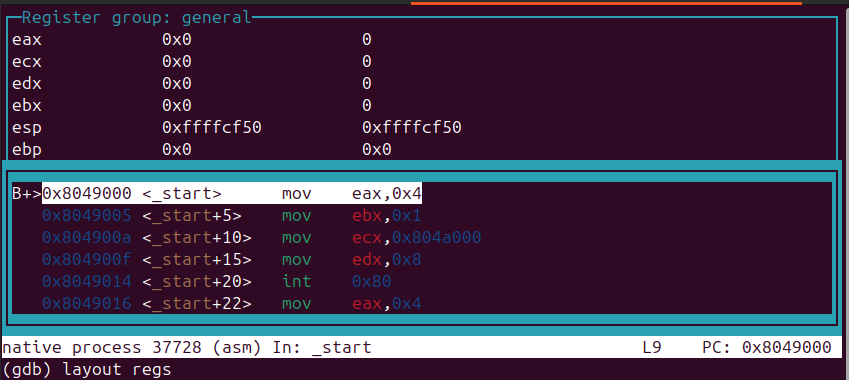


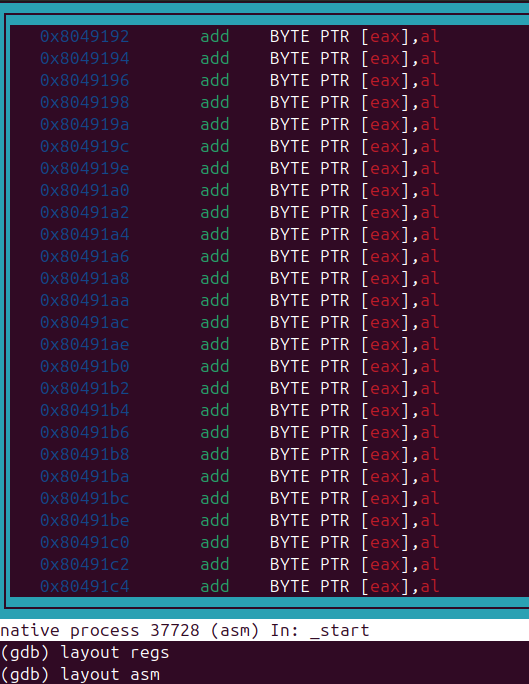
1. Далее с помощью специальных команд я переключился на intel’овское отображение синтаксиса:



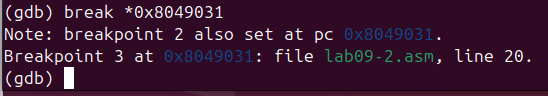
Отличие заключается в том, что в диссамилированном отображении в командах используют символы “%” и “$”, а в Intel’овском отображении эти символы не используются.

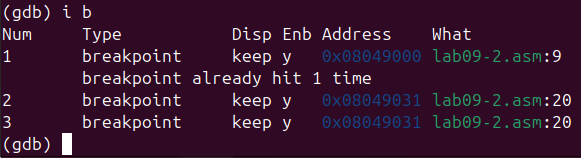
1. Потом я включил режим псевдографики для удобного анализа программы:



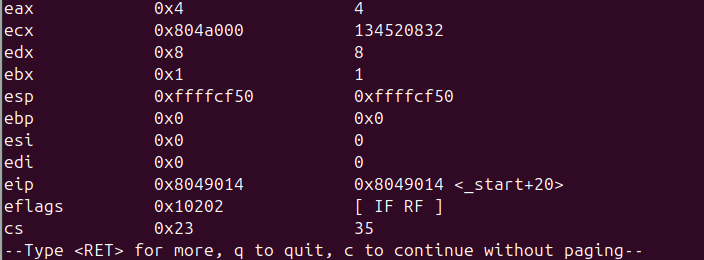


1. Я посмотрел наличие меток с помощью специальной команды и установил еще одну метку по адресу инструкции:





1. С помощью команды info registers я посмотрел содержимое регистров:



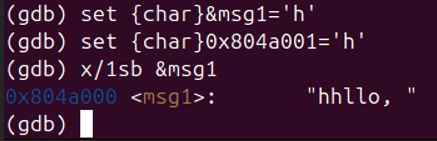
1. Далее посмотрел значение переменной msg1 по специальному имени:



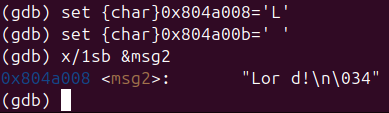
1. Потом посмотрел значение второй переменной msg2 по адресу:



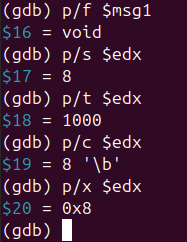
1. С помощью команды set я изменил значение переменной msg1:



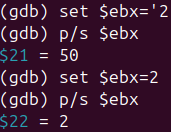
1. По аналогичному принципу я изменил переменную msg2:



1. Затем я вывел значение регистра edx в двоичном, символьном и 16-ичном виде:

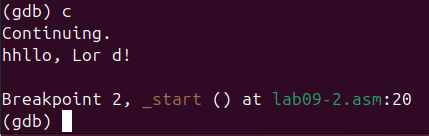


1. Я изменил значение регистра ebx следующим способом:



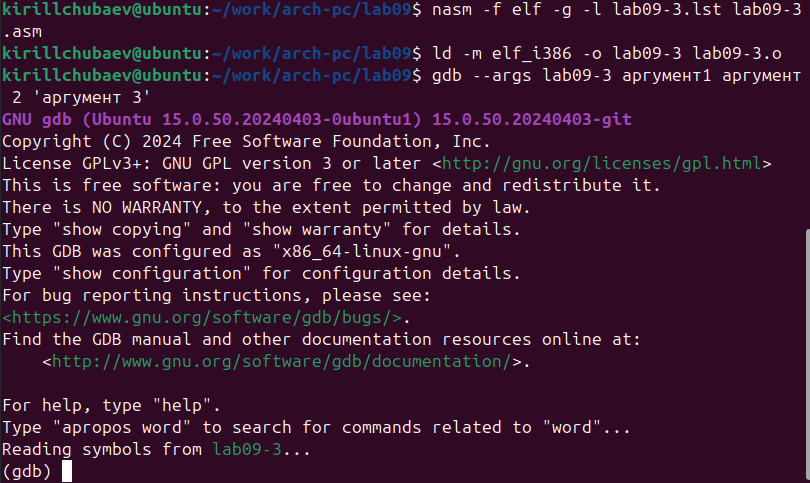
Команда выводит два разных значения, так как в первый раз вносится значение 2, а во втором случае регистр равен двум.

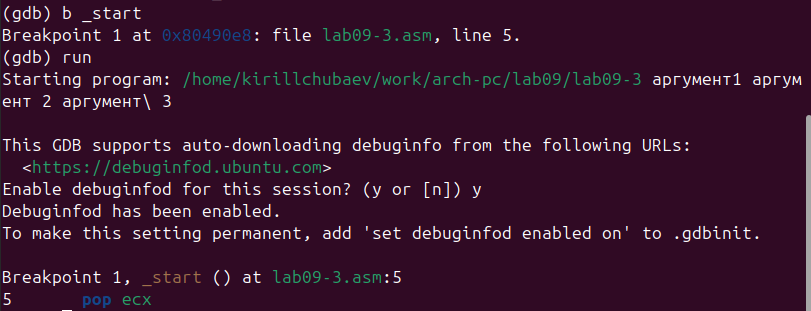
1. Я завершил работу с файлом в отладчике с помощью команд “c”, “si”, и “quit”:



1. Далее я скопировал файл lab8-2.asm из лабораторной работы №8 и переименовал его. Запустил файл в отладчике, указал аргументы и запустил файл, поставив метку на \_start:



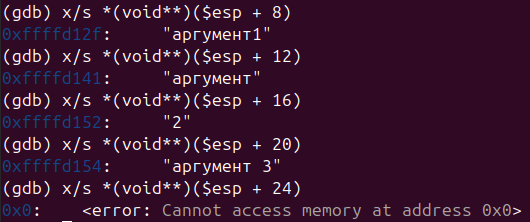




1. Я проверил адрес вершины стека и убедился, что там хранится 5 элементов:



1. Затем я посмотрел все позиции стека:



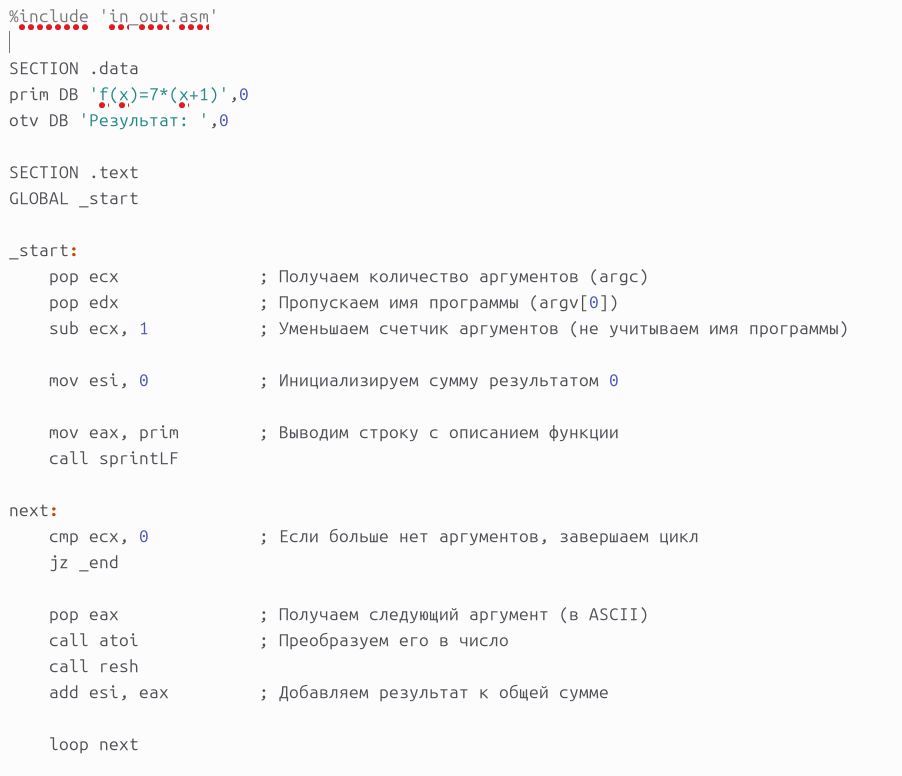
По первому адресу хранится сам адрес, в остальных адресах хранятся элементы. Элементы расположены с интервалом в 4 единицы, так как стек может хранить только до 4 байт. Для того чтобы данные сохранялись нормально и без помех, компьютер использует новый стек для новой информации.

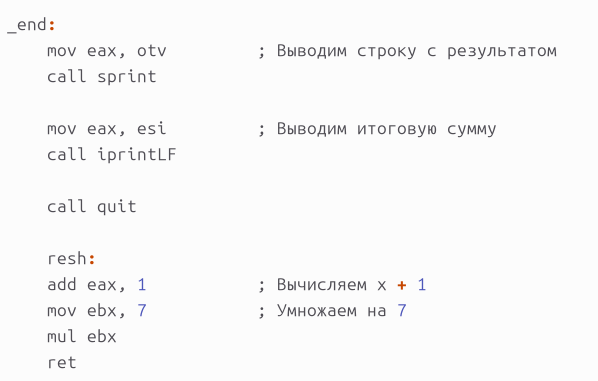
# Выполнение самостоятельной работы

1. Сначала я создал файл для выполнения первого задания самостоятельной работы под названием lab09-test1.asm:

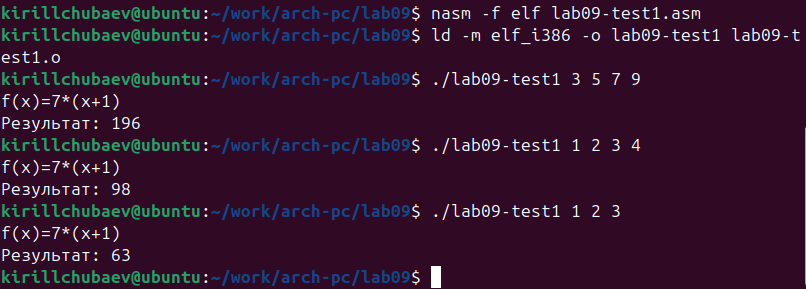


1. Затем преобразовал свой код программы из лабораторной работы №8 и реализовал вычисления как подпрограмму:





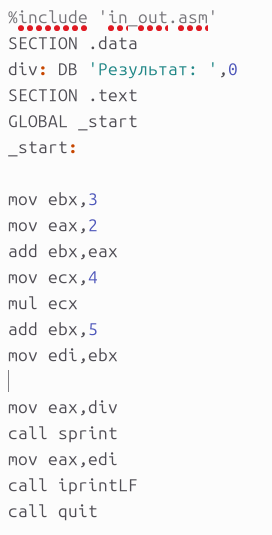
1. Создал исполняемый файл и запустил его, чтобы проверить правильность выполнения программы. Программа работает исправно:

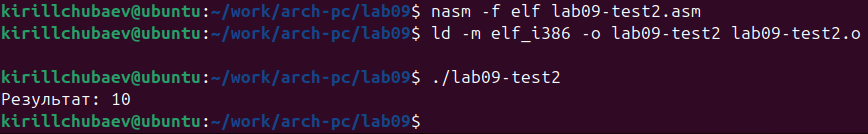


1. Для выполнения второго задания я создал файл lab09-test2.asm:

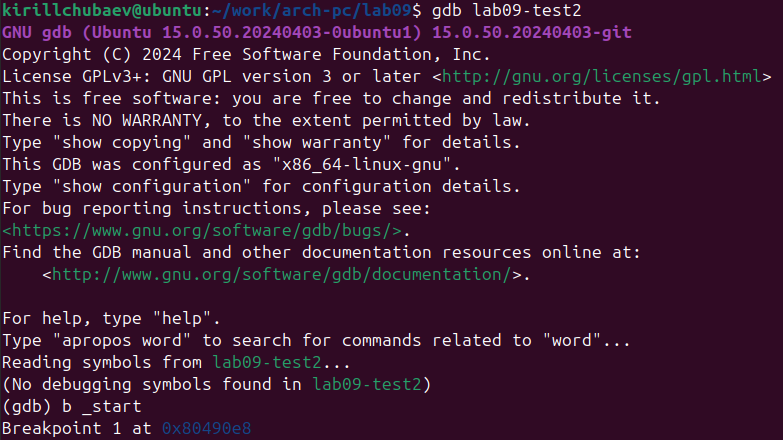


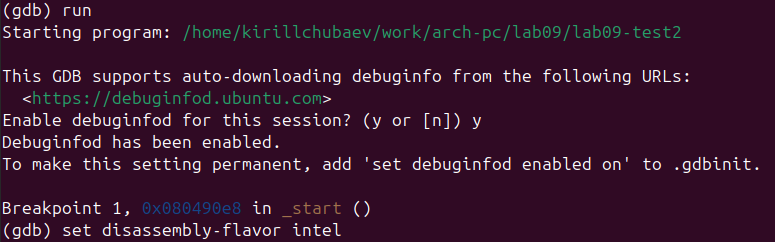
1. С помощью листинга 9.3 я написал код необходимой программы. Потом создал исполняемый файл и запустил его:

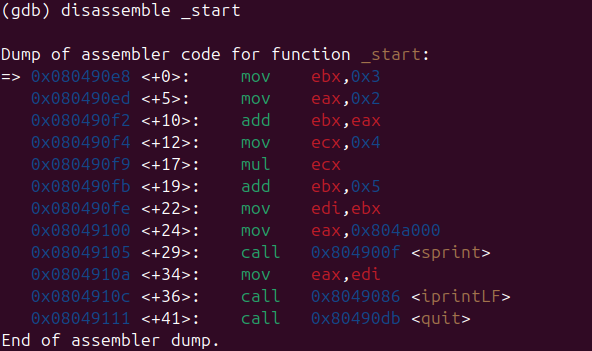




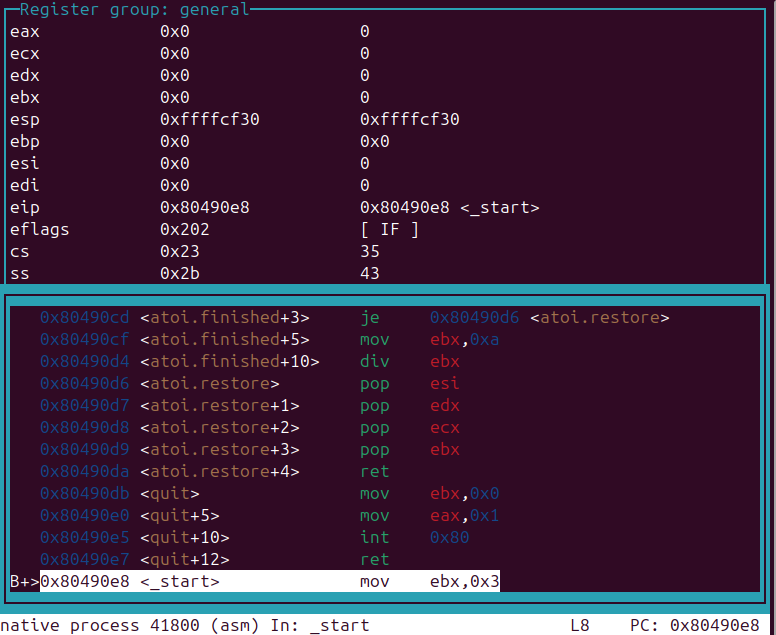
1. После выявления ошибки, которая связана с неправильным вычислением программы, я запустил её в отладчике:



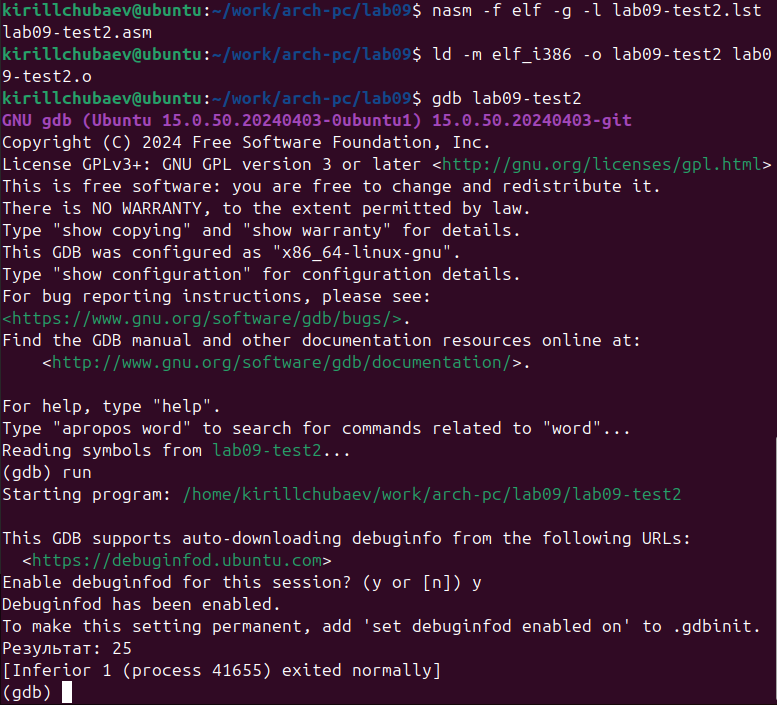




1. Затем я открыл регистры и внимательно проанализировал их. Я увидел, что некоторые регистры стоят не на своих местах и исправил это:



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



# Вывод

В ходе выполнения данной лабораторной работы я приобрел полезные навыки написания программ с использованием подпрограмм. Помимо этого, я познакомился с методами отладки при помощи GDB и с 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).