Отчет по лабораторной работе №9

Дисциплина: архитектура компьютера

Абуков Ислам Ренатович

Цель работы

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

Задание

1. Реализация подпрограмм в NASM
2. Отладка программ с помощью GDB
3. Самостоятельное выполнение заданий по материалам лабораторной работы

Теоретическое введение

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.

Можно выделить следующие типы ошибок:

• синтаксические ошибки — обнаруживаются во время трансляции исходного кода и вызваны нарушением ожидаемой формы или структуры языка; • семантические ошибки — являются логическими и приводят к тому, что программа запускается, отрабатывает, но не даёт желаемого результата; • ошибки в процессе выполнения — не обнаруживаются при трансляции и вызывают пре- рывание выполнения программы (например, это ошибки, связанные с переполнением или делением на ноль).

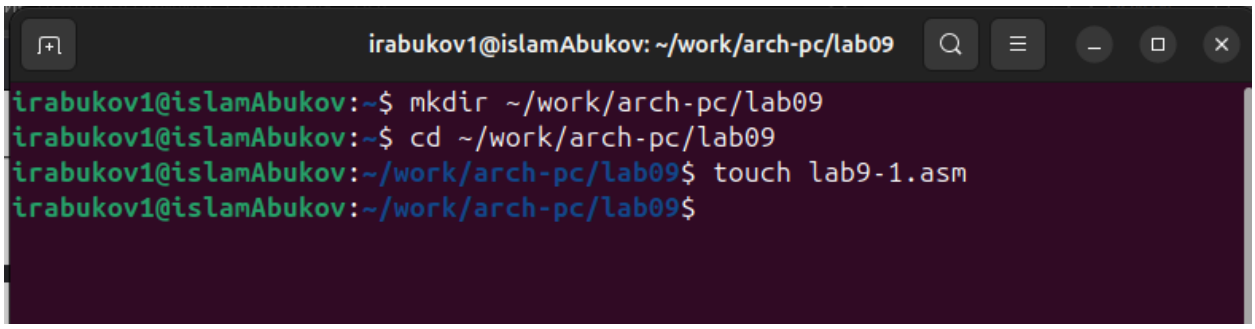
Второй этап — поиск местонахождения ошибки. Некоторые ошибки обнаружить доволь- но трудно. Лучший способ найти место в программе, где находится ошибка, это разбить программу на части и произвести их отладку отдельно друг от друга.

Третий этап — выяснение причины ошибки. После определения местонахождения ошибки обычно проще определить причину неправильной работы программы. Последний этап — исправление ошибки. После этого при повторном запуске программы, может обнаружиться следующая ошибка, и процесс отладки начнётся заново.

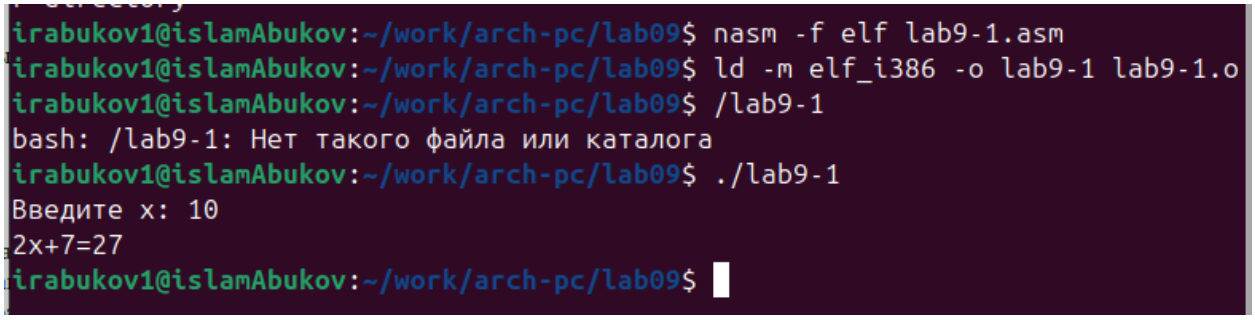
Выполнение лабораторной работы

Релазиация подпрограмм в NASM

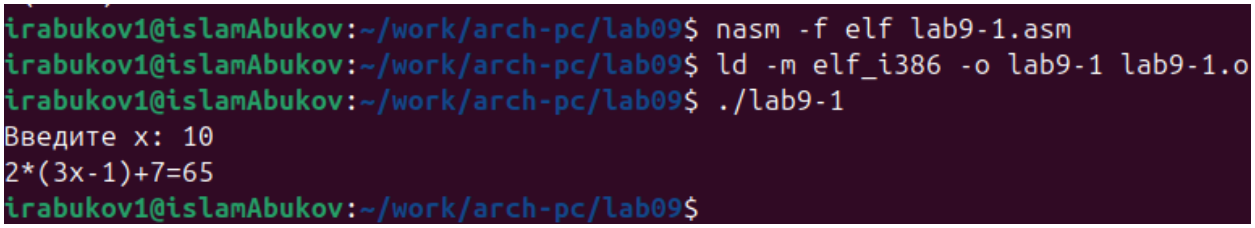
Создаю каталог для выполнения лабораторной работы №9



Копирую в файл код из листинга, компилирую и запускаю его, данная программа выполняет вычисление функции



Изменяю текст программы, добавив в нее подпрограмму, теперь она вычисляет значение функции для выражения f(g(x)).



Отладка программ с помощью GDB

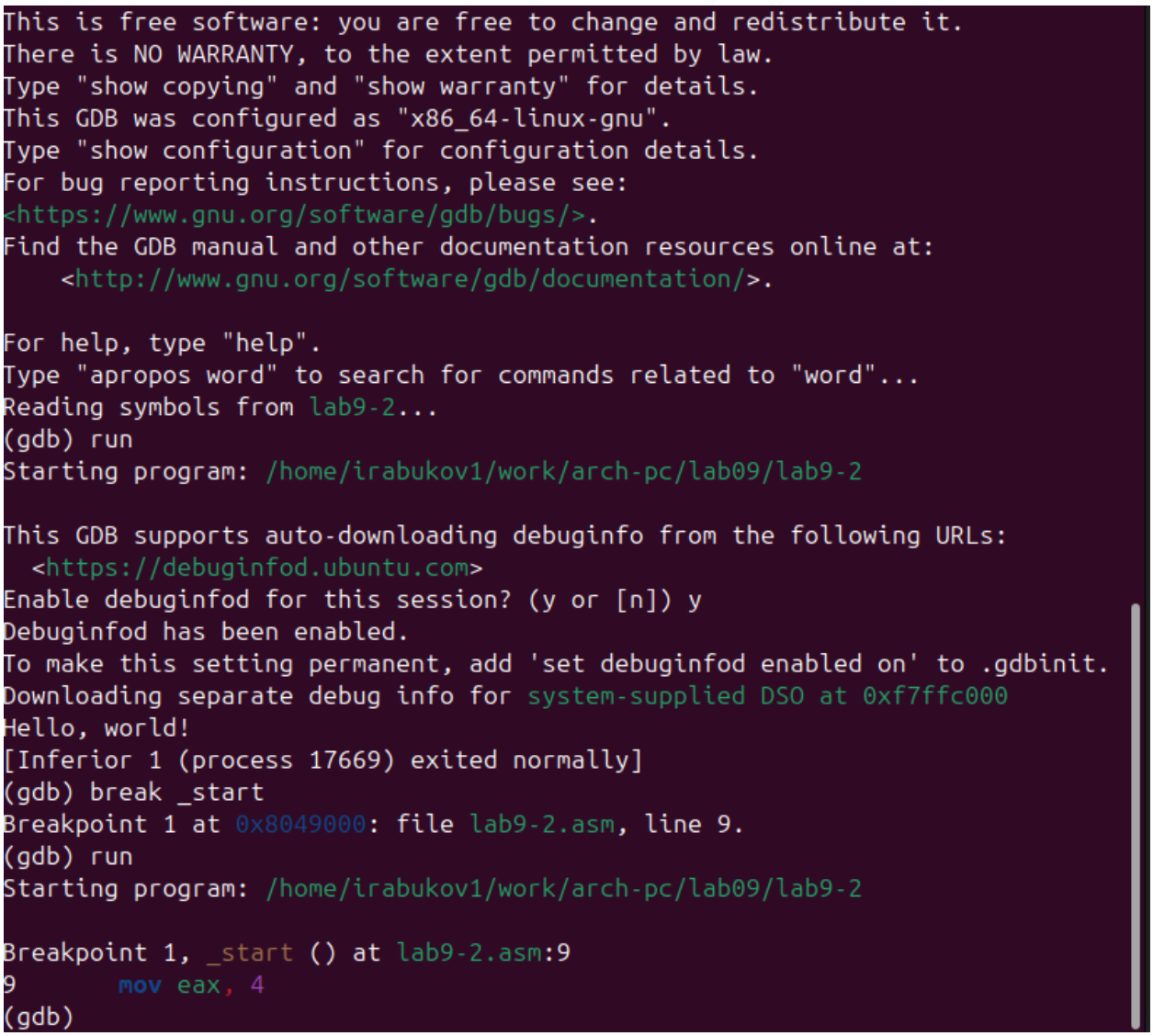
В созданный файл копирую программу второго листинга, транслирую с созданием файла листинга и отладки, компоную и запускаю в отладчике

```
irabukov1@islamAbukov:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9
-2.asm
irabukov1@islamAbukov:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
irabukov1@islamAbukov:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-2...
(gdb)
```

Запустив программу командой run, я убедился в том, что она работает исправно

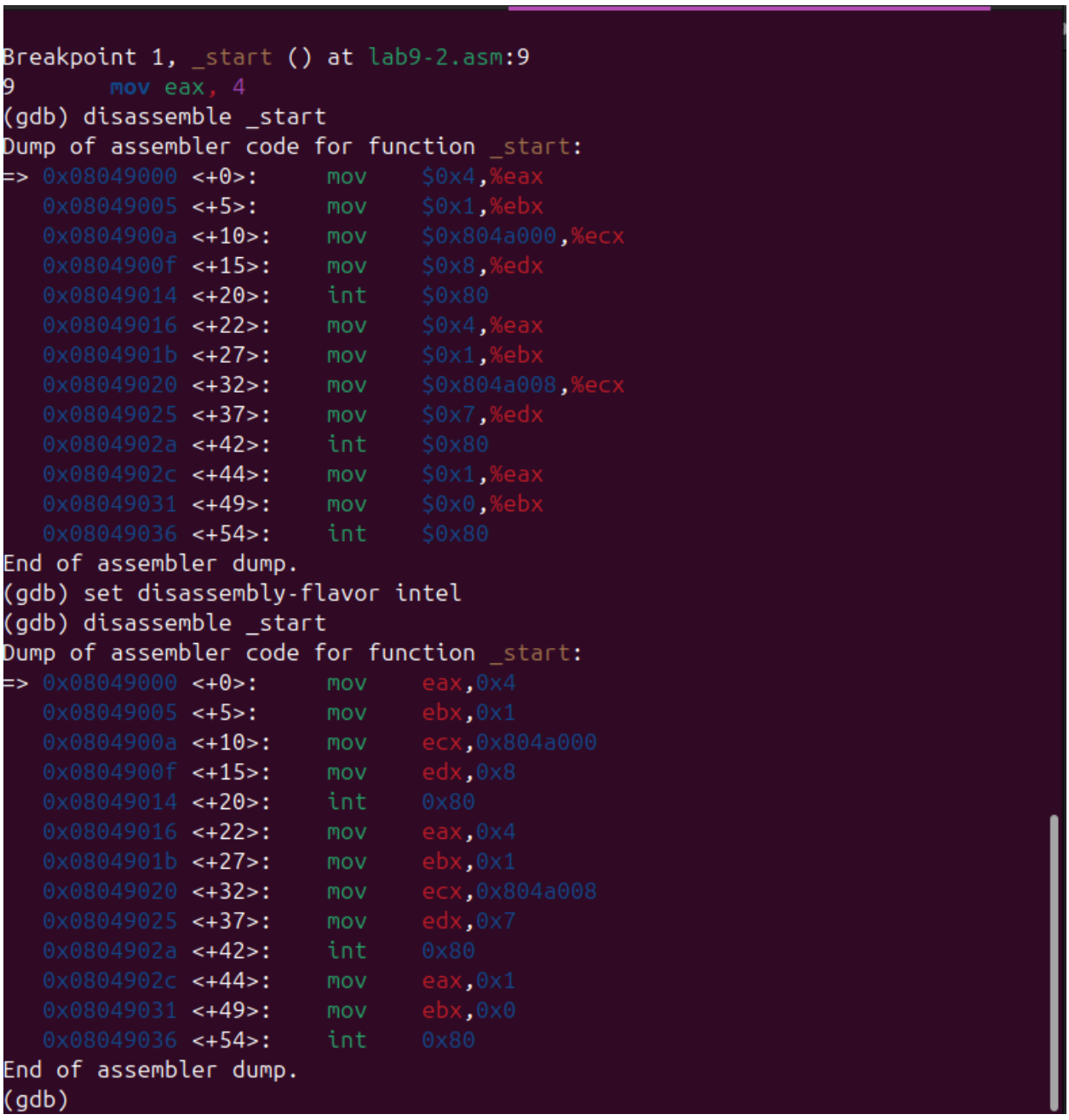
```
irabukov1@islamAbukov:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9
-2.asm
irabukov1@islamAbukov:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
irabukov1@islamAbukov:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-2...
(gdb) run
Starting program: /home/irabukov1/work/arch-pc/lab09/lab9-2

This GDB supports auto-downloading debuginfo from the following URLs:
  <https://debuginfod.ubuntu.com>
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Downloading separate debug info for system-supplied DSO at 0xf7ffc000
Hello, world!
[Inferior 1 (process 17669) exited normally]
(gdb)
```

Для более подробного анализа программы добавляю брейкпоинт на метку _start и снова запускаю отладку

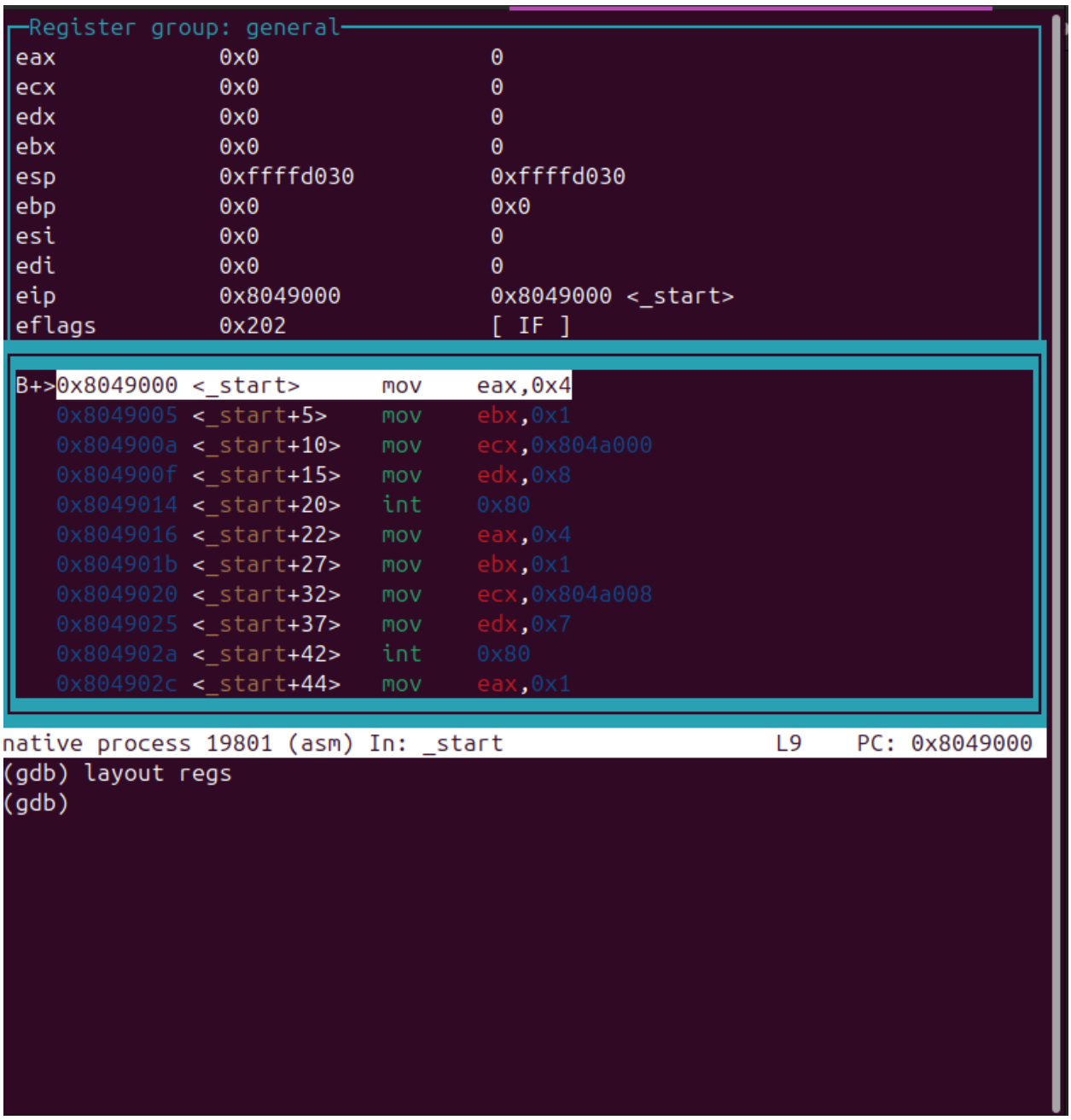


Далее смотрю дисассимилированный код программы, перевожу на команд с синтаксисом Intel *амд топчик.*

Различия между синтаксисом ATT и Intel заключаются в порядке операндов (ATT - Операнд источника указан первым. Intel - Операнд назначения указан первым), их размере (ATT - размер операндов указывается явно с помощью суффиксов, непосредственные операнды предваряются символом $; Intel - Размер операндов неявно определяется контекстом, как ax, eax, непосредственные операнды пишутся напрямую), именах регистров(ATT - имена регистров предваряются символом %, Intel - имена регистров пишутся без префиксов).

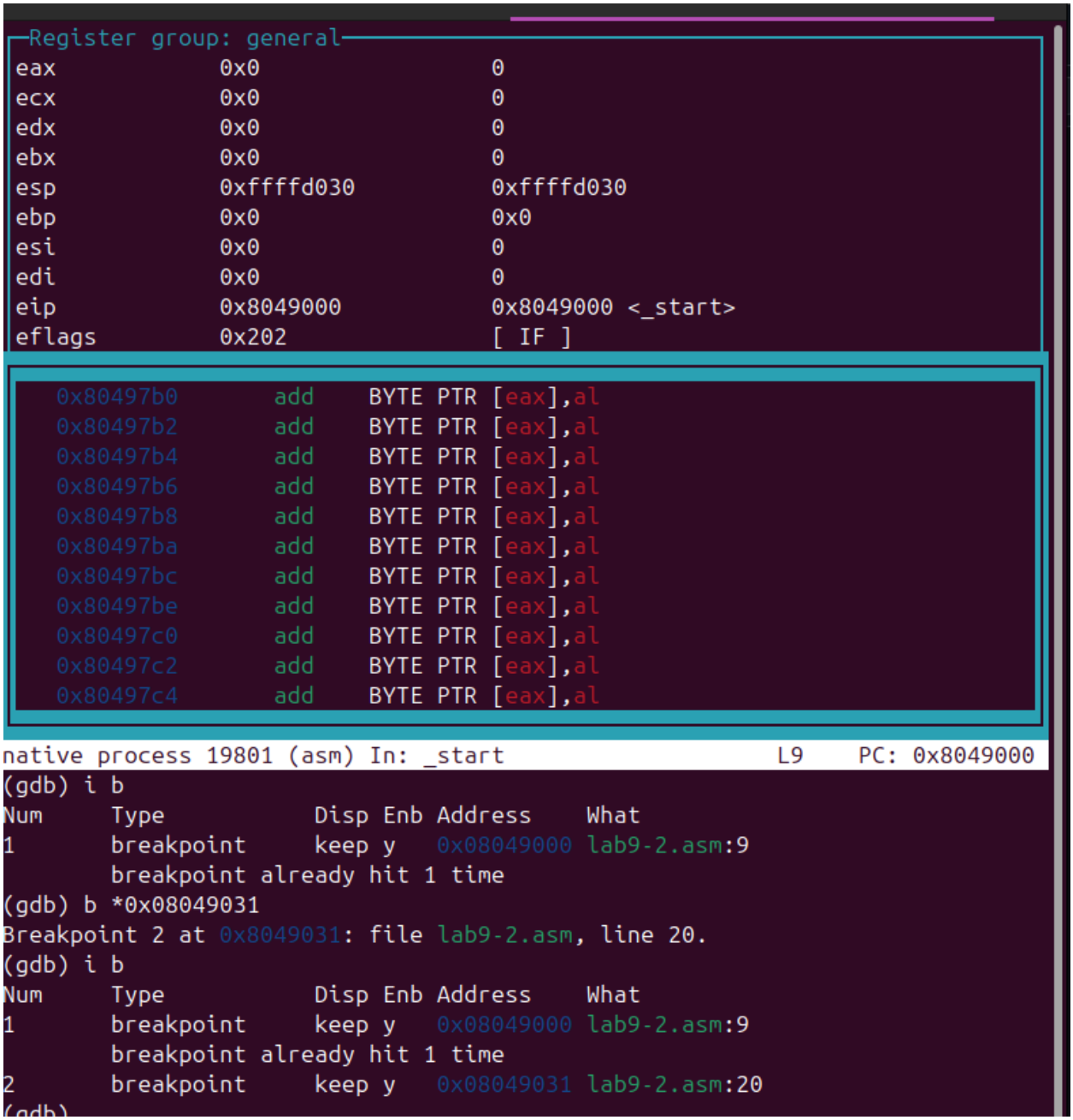


Включаю режим псевдографики для более удобного анализа программы.



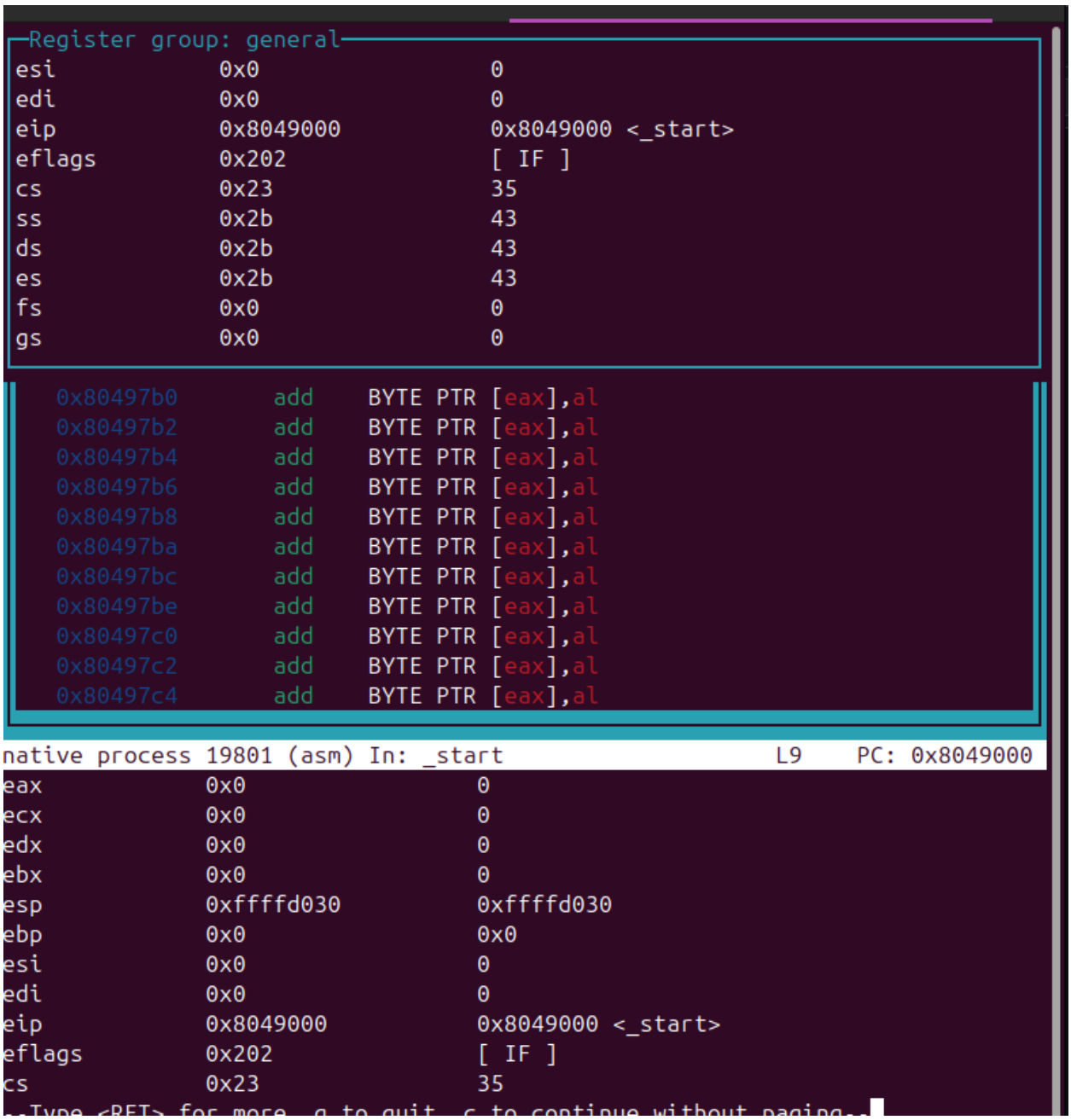
Добавление точек останова

Проверяю в режиме псевдографики, что брейкпоинт сохранился

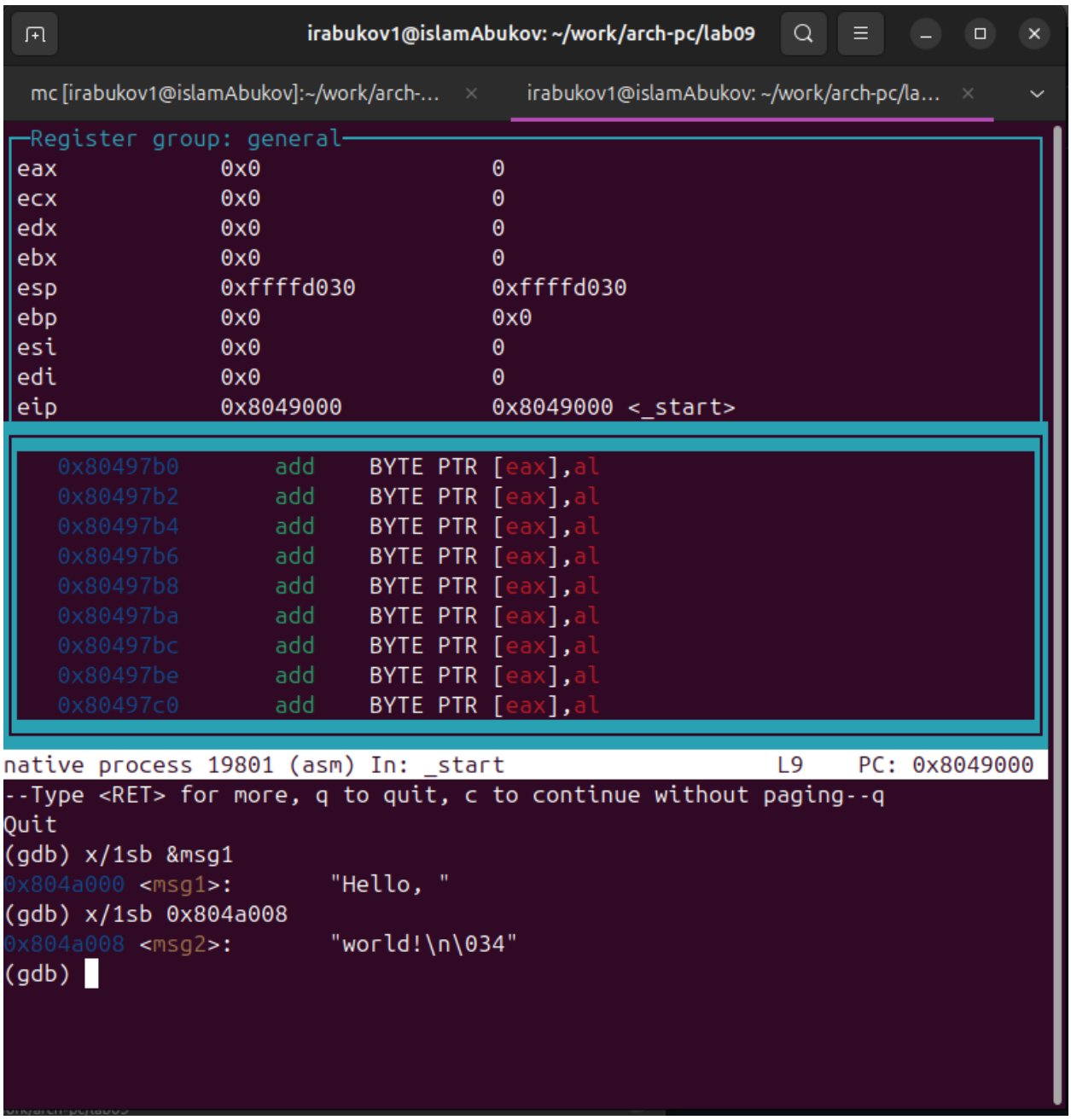


Работа с данными программы в GDB

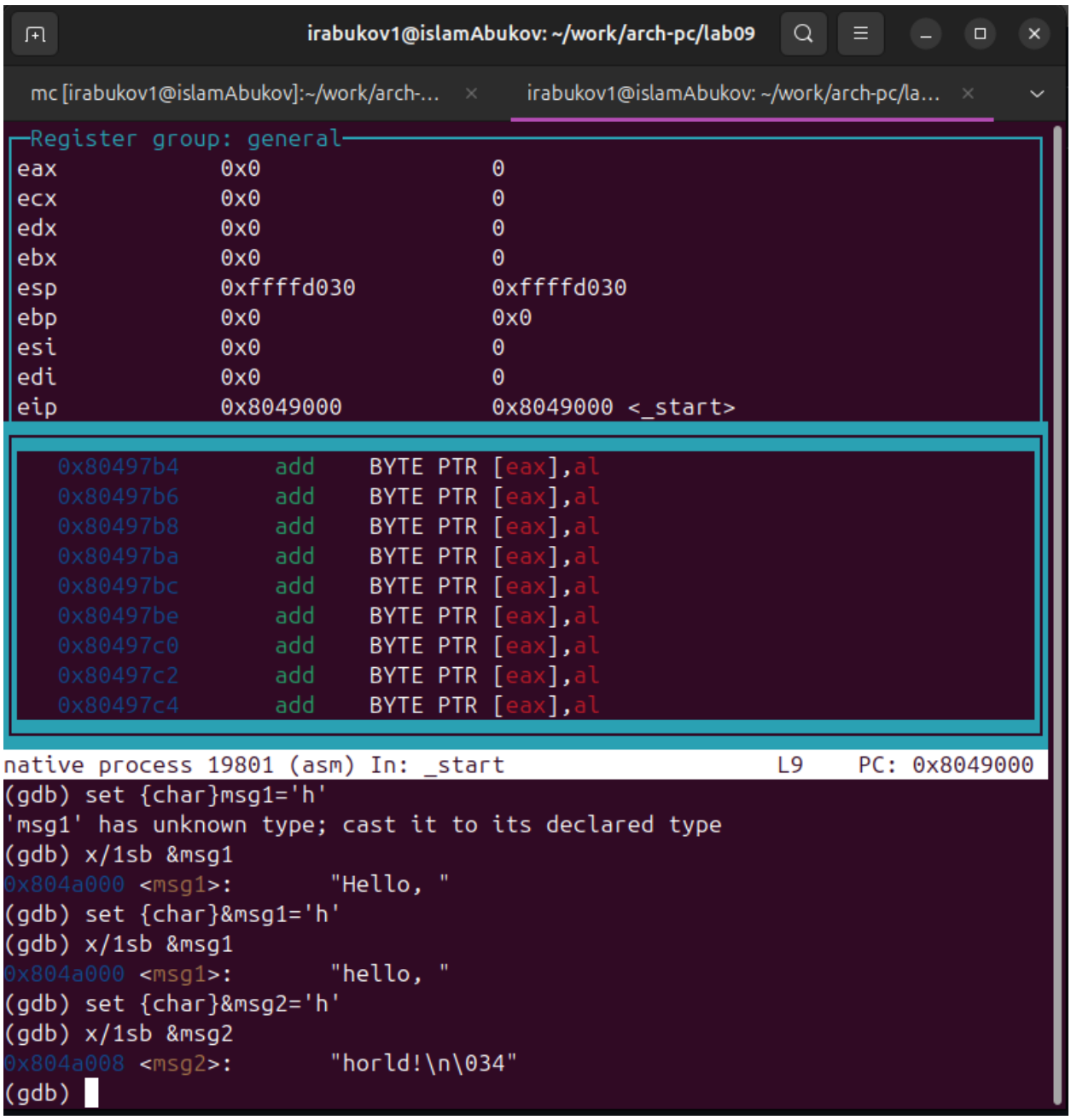
Просматриваю содержимое регистров командой info registers Просматриваю содержимое регистров командой info registers



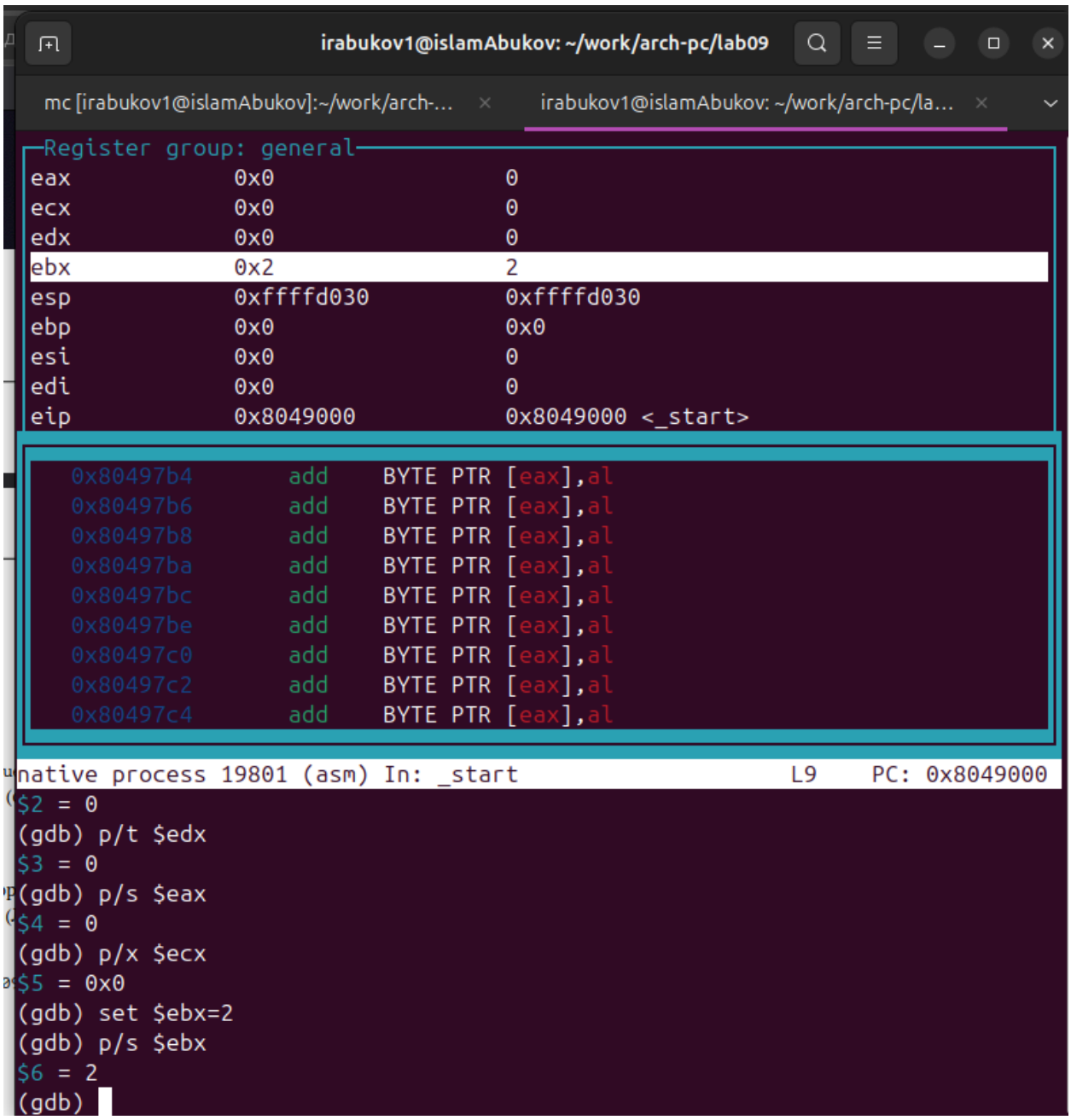
Смотрю содержимое переменных по имени и по адресу



Меняю содержимое переменных по имени и по адресу

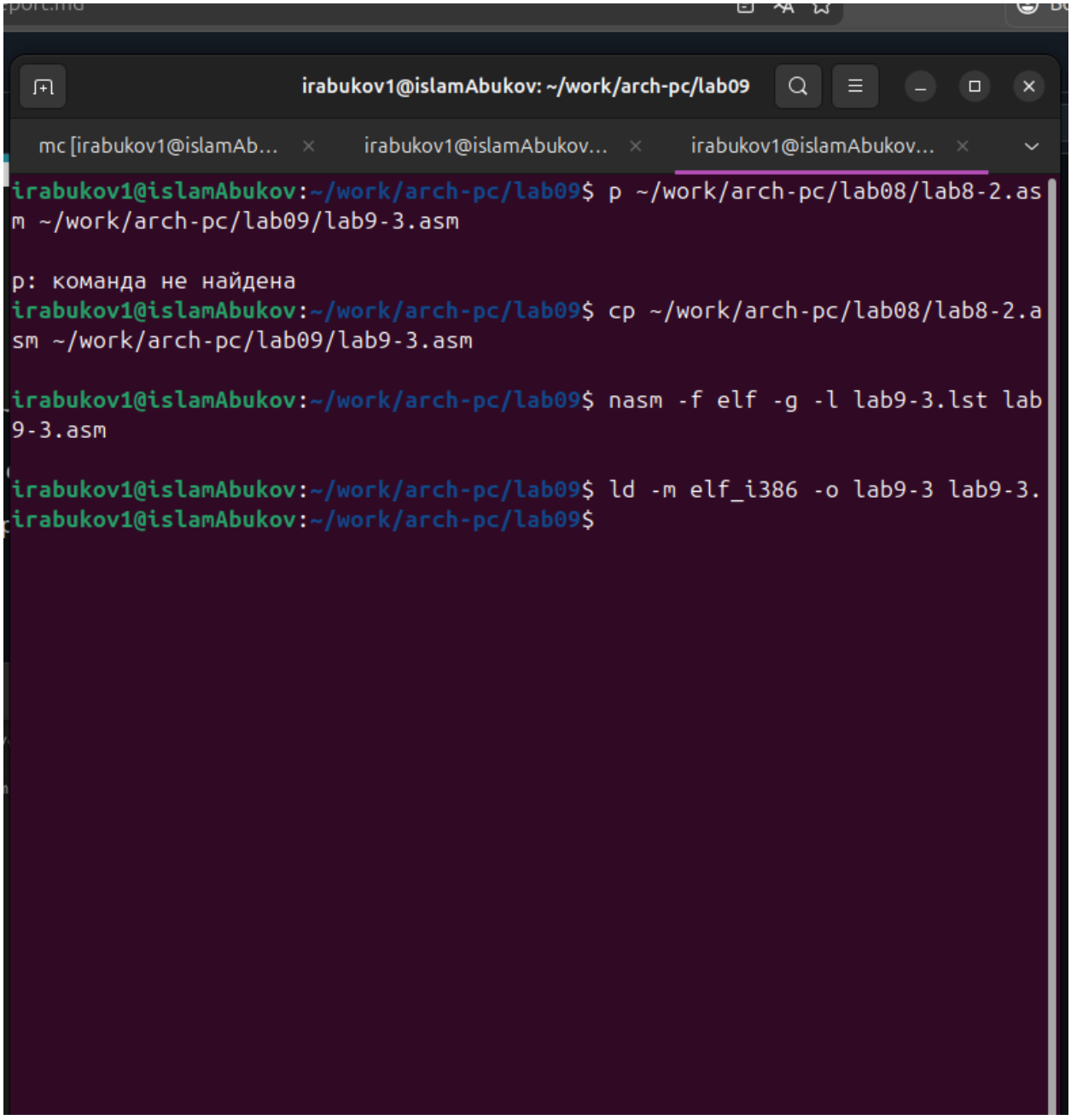


Вывожу в различных форматах значение регистра edx

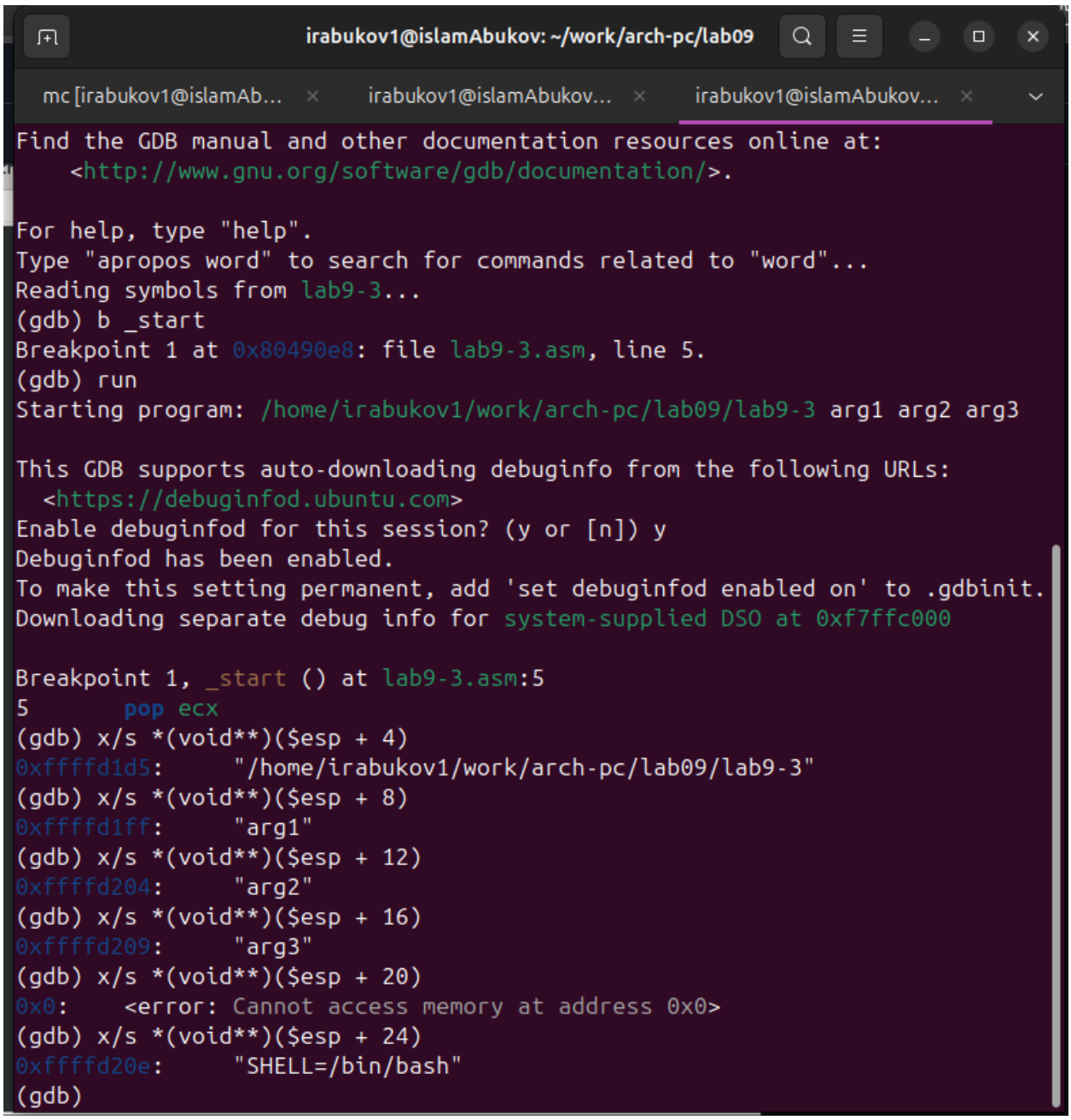


Обработка аргументов командной строки в GDB

Копирую программу из предыдущей лабораторной работы в текущий каталог и и создаю исполняемый файл с файлом листинга и отладки

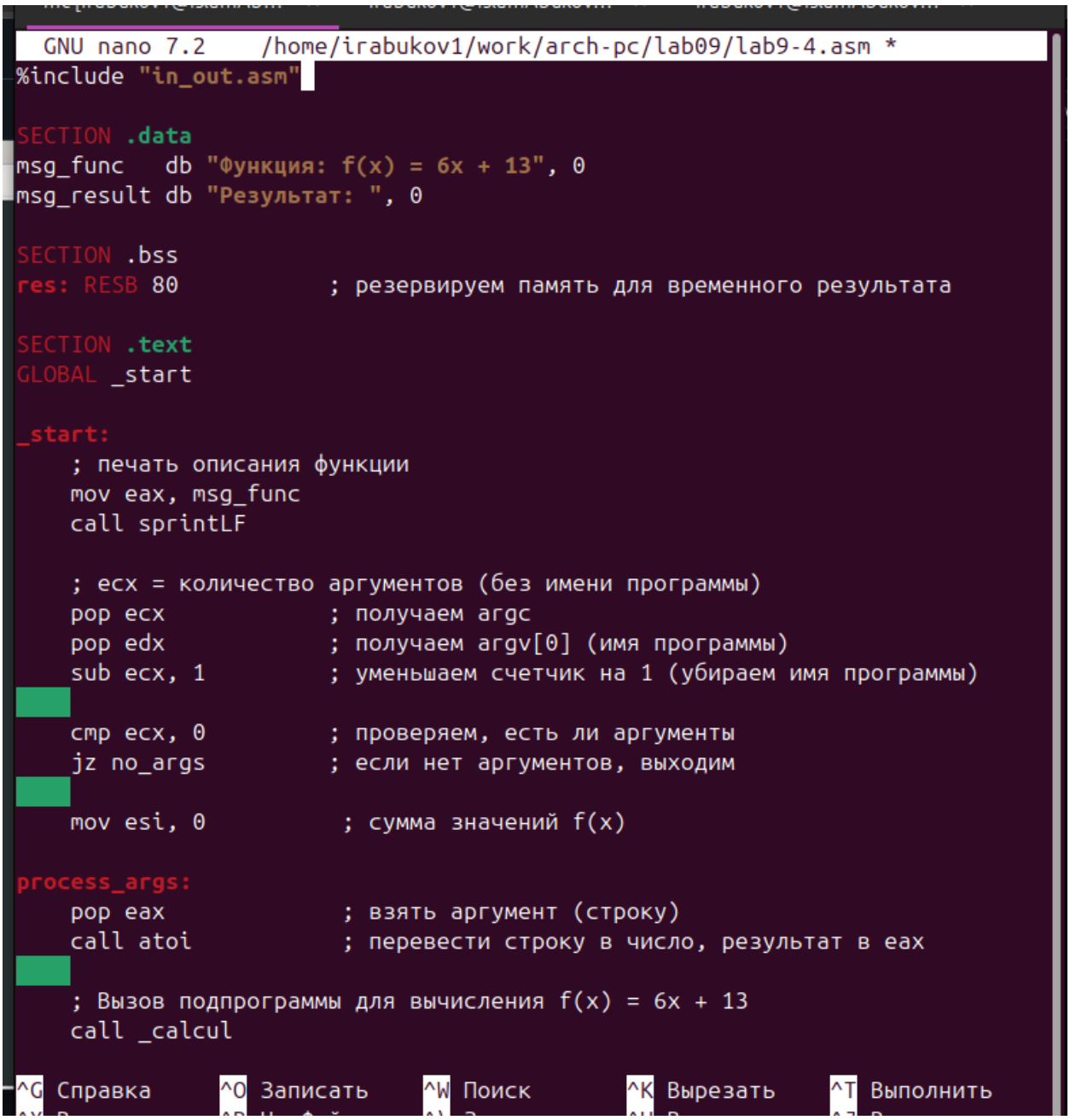


Запускаю программу с режиме отладки с указанием аргументов, указываю брейкпопнт и запускаю отладку. Проверяю работу стека, изменяя аргумент команды просмотра регистра esp на +4, число обусловлено разрядностью системы, а указатель void занимает как раз 4 байта, ошибка при аргументе +24 означает, что аргументы на вход программы закончились.

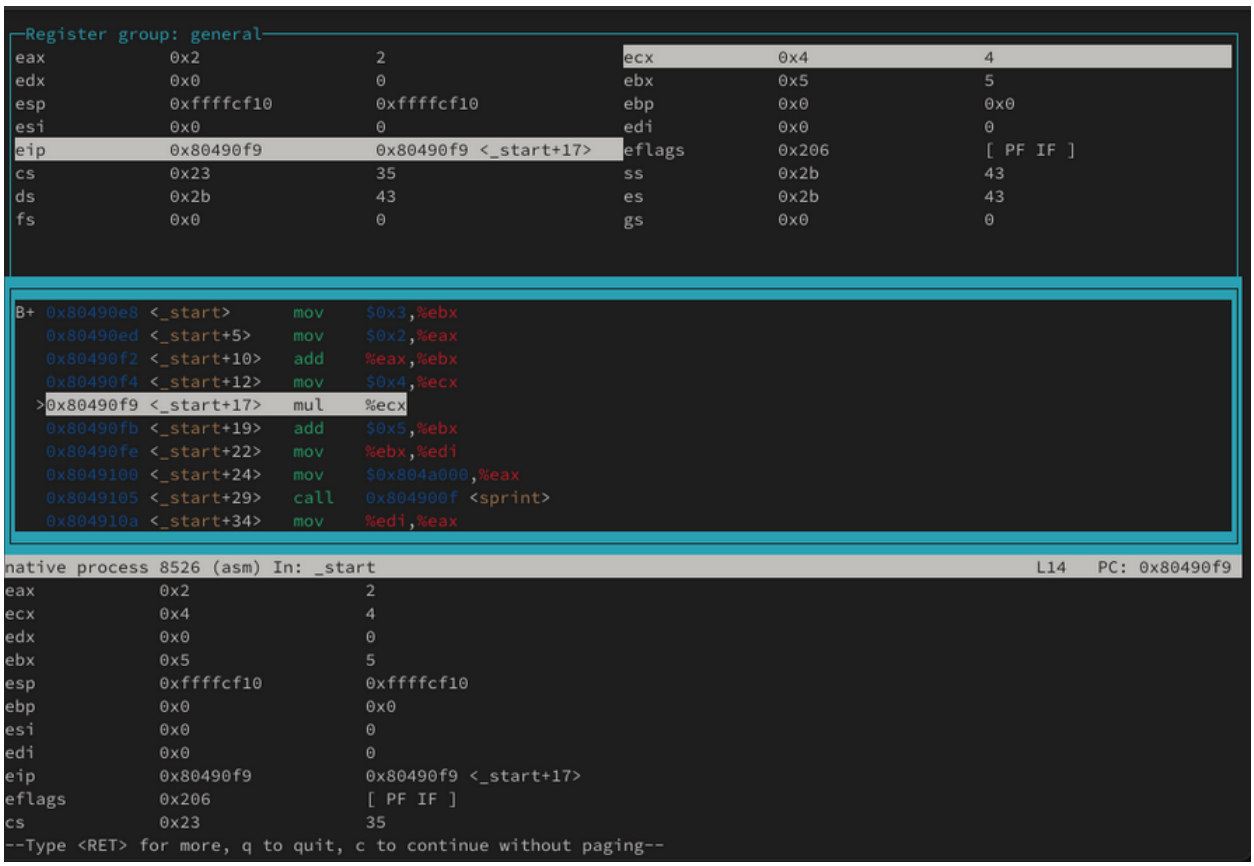


Задание для самостоятельной работы

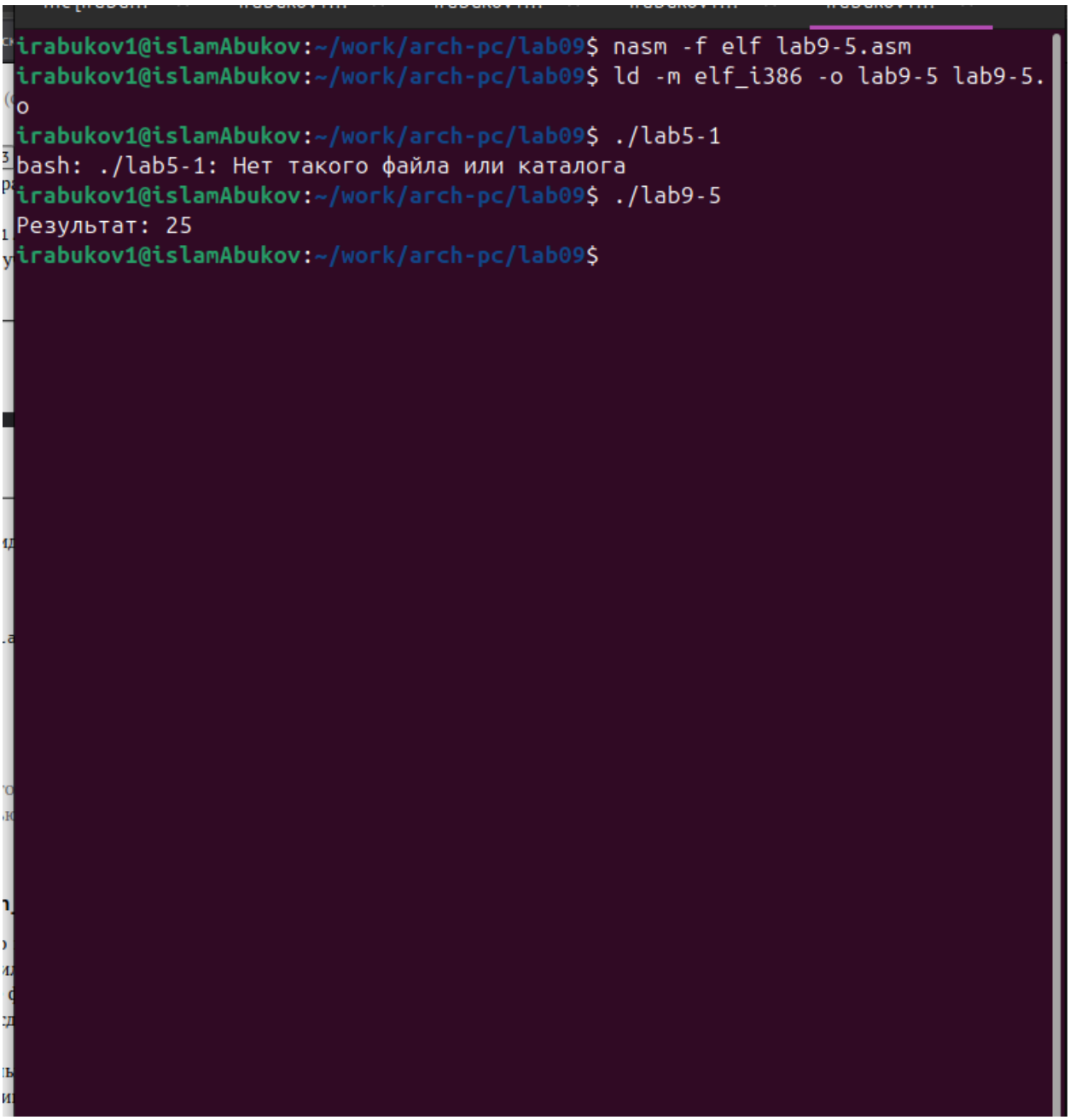
1. Меняю программу самостоятельной части предыдущей лабораторной работы с использованием подпрограммы



2. Запускаю программу в режике отладичка и пошагово через si просматриваю изменение значений регистров через i r. При выполнении инструкции mul ecx можно заметить, что результат умножения записывается в регистр eax, но также меняет и edx. Значение регистра ebx не обновляется напрямую, поэтому результат программа неверно подсчитывает функцию



Исправляю найденную ошибку, теперь программа верно считает значение функции


irabukov1@islamAbukov:~/work/arch-pc/lab09$ nasm -f elf lab9-5.asm
irabukov1@islamAbukov:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-5 lab9-5.
o
irabukov1@islamAbukov:~/work/arch-pc/lab09$ ./lab5-1
bash: ./lab5-1: Нет такого файла или каталога
irabukov1@islamAbukov:~/work/arch-pc/lab09$ ./lab9-5
Результат: 25
irabukov1@islamAbukov:~/work/arch-pc/lab09$