

Лабораторная работа №9

Понятие подпрограммы. Отладчик GDB.

Тютрюмова Анжелина

Содержание

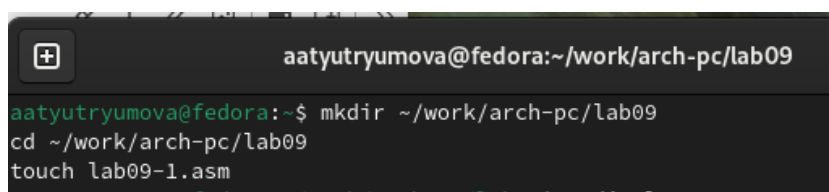
1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Выполнение самостоятельной работы	15
4	Выводы	20

1 Цель работы

Целью работы является приобретение навыков написания программ с использованием подпрограмм. Знакомство с методами отладки при помощи GDB и его основными возможностями.

2 Выполнение лабораторной работы

Создала и перешла в директорию для лабораторной работы создала файл lab9-1.asm (рис. 2.1).

A terminal window with a dark background. The title bar shows a window icon and the text 'aatyutryumova@fedora:~/work/arch-pc/lab09'. The terminal content shows three commands being executed: 'mkdir ~/work/arch-pc/lab09', 'cd ~/work/arch-pc/lab09', and 'touch lab09-1.asm'. The prompt is 'aatyutryumova@fedora:~\$' for each line.

```
aatyutryumova@fedora:~$ mkdir ~/work/arch-pc/lab09
aatyutryumova@fedora:~$ cd ~/work/arch-pc/lab09
aatyutryumova@fedora:~/work/arch-pc/lab09$ touch lab09-1.asm
```

Рис. 2.1: Папка для лабораторной работы

Переписала код с лабараторной работы(рис. 2.2).

```
GNU nano 7.2 lab09-1.asm
%include 'in_out.asm'
SECTION .data
    msg: DB 'Введите x: ',0
    result: DB '2x+7=',0
SECTION .bss
    x: RESB 80
    res: RESB 80
SECTION .text
GLOBAL _start
_start:

;-----
; Основная программа
;-----

    mov eax, msg
    call sprint

    mov ecx, x
    mov edx, 80

[ Прочитано 43 строки ]
^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^J Выводить
```

Рис. 2.2: Листинг кода

Листинг кода:

```
%include 'in_out.asm'
SECTION .data
    msg: DB 'Введите x: ',0
    result: DB '2x+7=',0
SECTION .bss
    x: RESB 80
    res: RESB 80
SECTION .text
GLOBAL _start
_start:

;-----
```

```

; Основная программа
;-----

    mov eax, msg
    call sprint

    mov ecx, x
    mov edx, 80
    call sread

    mov eax, x
    call atoi

    call _calcul      ; Вызов подпрограммы _calcul

    mov eax, result
    call sprint
    mov eax, [res]
    call iprintLF

    call quit

;-----
; Подпрограмма вычисления
; выражения "2x+7"

_calcul:
    mov ebx, 2
    mul ebx

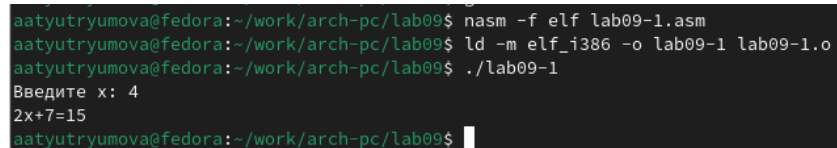
```

```

    add eax,7
    mov [res],eax
ret ; выход из подпрограммы

```

Создала исполняемый файл и запустила его. (рис. 2.3).



```

aatyutryumova@fedora:~/work/arch-pc/lab09$ nasm -f elf lab09-1.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-1 lab09-1.o
aatyutryumova@fedora:~/work/arch-pc/lab09$ ./lab09-1
Введите x: 4
2x+7=15
aatyutryumova@fedora:~/work/arch-pc/lab09$

```

Рис. 2.3: Результат выполнения

Создала файл lab09-2.asm с кодом из лабораторной работы. (Программа печати сообщения Hello world!)

Листинг кода:

```

SECTION .data
    msg1: db "Hello, ",0x0
    msg1Len: equ $ - msg1
    msg2: db "world!",0xa
    msg2Len: equ $ - msg2

SECTION .text
    global _start

_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, msg1
    mov edx, msg1Len
    int 0x80
    mov eax, 4
    mov ebx, 1
    mov ecx, msg2

```

```

mov edx, msg2Len
int 0x80
mov eax, 1
mov ebx, 0
int 0x80

```

Получила исполняемый файл. Для работы с GDB добавила в исполняемый файл отладочную информацию, трансляцией с ключом '-g'. Загрузила исполняемый файл в отладчике gdb. (рис. 2.4).

```

aatyutryumova@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-2.lst lab09-2.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-2 lab09-2.o
aatyutryumova@fedora:~/work/arch-pc/lab09$ gdb lab09-2
GNU gdb (Fedora Linux) 15.2-3.fc40
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-redhat-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 lab09-2...

```

Рис. 2.4: Отладчике gdb

Проверила работу программы, запустив ее в оболочке GDB с помощью команды run. (рис. 2.5).

```

(gdb) run

Starting program: /home/aatyutryumova/work/arch-pc/lab09/lab09-2

This GDB supports auto-downloading debuginfo from the following URLs:
<https://debuginfod.fedoraproject.org/>
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 5697) exited normally]

```

Рис. 2.5: Результат выполнения

Для более подробного анализа программы установила брейкпоинт на метку _start, с которой начинается выполнение любой ассемблерной программы, и

запустила её. (рис. 2.6).

```
(gdb) break _start

Breakpoint 1 at 0x8049000: file lab09-2.asm, line 9.
(gdb) run

Starting program: /home/aatyutryumova/work/arch-pc/lab09/lab09-2

Breakpoint 1, _start () at lab09-2.asm:9
9      mov eax, 4
(gdb)
```

Рис. 2.6: Добавление брейкпоинта

Посмотрела дисассимилированный код программы с помощью команды `disassemble` начиная с метки `_start` (рис. 2.7).

```
(gdb) disassemble _start

Dump of assembler code for function _start:
=> 0x08049000 <+0>:      mov     $0x4,%eax
0x08049005 <+5>:      mov     $0x1,%ebx
0x0804900a <+10>:     mov     $0x804a000,%ecx
0x0804900f <+15>:     mov     $0x8,%edx
0x08049014 <+20>:     int     $0x80
0x08049016 <+22>:     mov     $0x4,%eax
0x0804901b <+27>:     mov     $0x1,%ebx
0x08049020 <+32>:     mov     $0x804a008,%ecx
0x08049025 <+37>:     mov     $0x7,%edx
0x0804902a <+42>:     int     $0x80
0x0804902c <+44>:     mov     $0x1,%eax
0x08049031 <+49>:     mov     $0x0,%ebx
0x08049036 <+54>:     int     $0x80
End of assembler dump.
(gdb)
```

Рис. 2.7: Дисассимилированный код

Переключитесь на отображение команд с Intel'овским синтаксисом, введя команду `set disassembly-flavor intel` рис. 2.8).

```

(gdb) set disassembly-flavor intel

(gdb) disassemble _start

Dump of assembler code for function _start:
=> 0x08049000 <+0>:      mov     eax,0x4
    0x08049005 <+5>:      mov     ebx,0x1
    0x0804900a <+10>:     mov     ecx,0x804a000
    0x0804900f <+15>:     mov     edx,0x8
    0x08049014 <+20>:     int     0x80
    0x08049016 <+22>:     mov     eax,0x4
    0x0804901b <+27>:     mov     ebx,0x1
    0x08049020 <+32>:     mov     ecx,0x804a008
    0x08049025 <+37>:     mov     edx,0x7
    0x0804902a <+42>:     int     0x80
    0x0804902c <+44>:     mov     eax,0x1
    0x08049031 <+49>:     mov     ebx,0x0
    0x08049036 <+54>:     int     0x80
End of assembler dump.
(gdb) █

```

Рис. 2.8: Отображение команд с Intel'овским синтаксисом

Различие отображения синтаксиса машинных команд в режимах АТТ и Intel состоит в том, что меняется положение название регистров и его значения. Включила режим псевдографики для более удобного анализа программы (рис. 2.9).

```

aatyutryumova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
Register group: general
eax      0x0      0
ecx      0x0      0
edx      0x0      0
ebx      0x0      0
esp      0xffffd020 0xffffd020
ebp      0x0      0x0

B->0x8049000 <_start>    mov     eax,0x4
0x8049005 <_start+5>    mov     ebx,0x1
0x804900a <_start+10>   mov     ecx,0x804a000
0x804900f <_start+15>   mov     edx,0x8
0x8049014 <_start+20>   int     0x80
0x8049016 <_start+22>   mov     eax,0x4

native process 5759 (asm) In: _start
(gdb) layout regs
(gdb)

```

Рис. 2.9: Результат работы

Установила еще одну точку остановки по адресу инструкции. Для этого определила адрес предпоследней инструкции (mov ebx,0x0) и установила точку остановки. Посмотрела информацию о всех установленных брейкпоинтах. (рис. 2.10, рис. 2.11, рис. 2.12).

```

0x804902a <_start+42>   int     0x80
0x804902c <_start+44>   mov     eax,0x1
0x8049031 <_start+49>   mov     ebx,0x0
0x8049036 <_start+54>   int     0x80

```

Рис. 2.10: Адрес предпоследней инструкции

```

(gdb) break *0x8049031
Breakpoint 3 at 0x8049031: file lab09-2.asm, line 20.
(gdb)

```

Рис. 2.11: Установка брейкпоинта

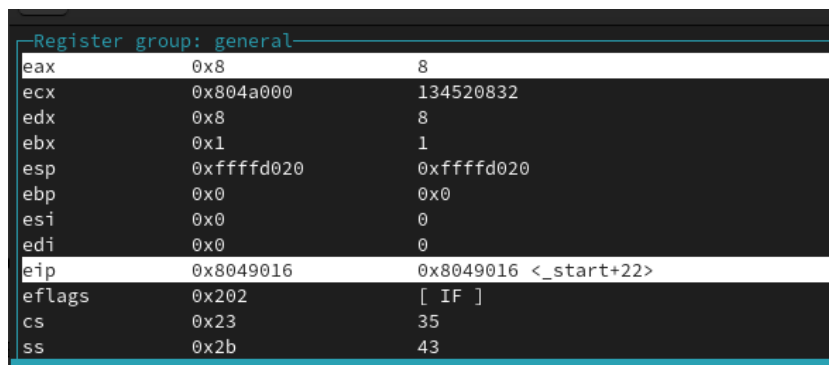
```

(gdb) break *0x8049031
Breakpoint 3 at 0x8049031: file lab09-2.asm, line 20.
(gdb) i b
Num    Type           Disp Enb Address      What
1      breakpoint      keep y   0x08049000 lab09-2.asm:9
       breakpoint already hit 1 time
2      breakpoint      keep y   <PENDING> 0x8049031
3      breakpoint      keep y   0x08049031 lab09-2.asm:20
(gdb)

```

Рис. 2.12: Информация о брейкпоинтах

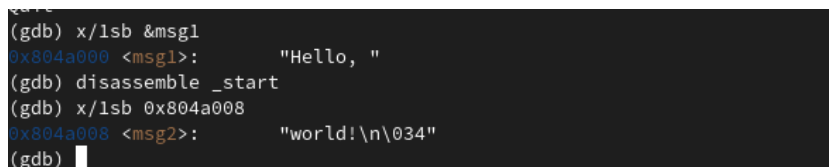
Выполнила 5 инструкций с помощью команды `si` и проследите за изменением значений регистров. Значения регистров `eax` и `eip` (рис. 2.13).



Register group: general		
eax	0x8	8
ecx	0x804a000	134520832
edx	0x8	8
ebx	0x1	1
esp	0xffffd020	0xffffd020
ebp	0x0	0x0
esi	0x0	0
edi	0x0	0
eip	0x8049016	0x8049016 <_start+22>
eflags	0x202	[IF]
cs	0x23	35
ss	0x2b	43

Рис. 2.13: Информация о брейкпоинтах

Посмотрела значения переменных `msg1` и `msg2` по имени (рис. 2.14).



```
(gdb) x/1sb &msg1
0x804a000 <msg1>: "Hello, "
(gdb) disassemble _start
(gdb) x/1sb 0x804a008
0x804a008 <msg2>: "world!\n\034"
(gdb)
```

Рис. 2.14: Значение переменных `msg1` и `msg2`

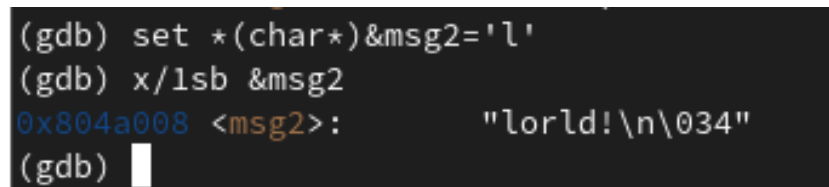
Изменила первый символ переменной `msg1` (рис. 2.15).



```
(gdb) set *(char*)&msg1='h'
(gdb) x/1sb &msg1
0x804a000 <msg1>: "hello, "
(gdb)
```

Рис. 2.15: Переменная `msg1`

Изменила первый символ переменной `msg2` (рис. 2.16).



```
(gdb) set *(char*)&msg2='l'
(gdb) x/1sb &msg2
0x804a008 <msg2>: "lorld!\n\034"
(gdb)
```

Рис. 2.16: Переменная `msg2`

С помощью команды `set` изменила значение регистра `ebx` (рис. 2.17).

```
(gdb) set $ebx=2
(gdb) p/s $ebx
$2 = 2
```

Рис. 2.17: Значение регистра `ebx`

Завершила выполнение программы с помощью команды `continue` и вышла из GDB с помощью команды `quit` (рис. 2.18).

```
(gdb) continue
Continuing.
world!

Breakpoint 3, _start () at lab09-2.asm:20
(gdb) quit
```

Рис. 2.18: Завершение работы

Скопировала файл `lab8-2.asm`, созданный при выполнении лабораторной работы №8, с программой выводящей на экран аргументы командной строки в файл с именем `lab09-3.asm`. Создала исполняемый файл и загрузила в `gdb` программу с аргументами и ключом `-args`. (рис. 2.19).

```
aatyutryumova@fedora:~/work/arch-pc/lab09$ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab09-3.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-3.lst lab09-3.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-3 lab09-3.o
aatyutryumova@fedora:~/work/arch-pc/lab09$ gdb --args lab09-3 аргумент1 аргумент 2 'аргумент 3'
GNU gdb (Fedora Linux) 15.2-3.fc40
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-redhat-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 lab09-3...
(gdb)
```

Рис. 2.19: `lab09-3.asm`

Установила точку останова перед первой инструкцией в программе и запустила ее (рис. 2.20).

```
Breakpoint 1 at 0x00000000: file lab09-3.asm, line 10.
(gdb) run
Starting program: /home/aatyutryumova/work/arch-pc/lab09/lab09-3 аргумент1 аргумент 2 аргумент\ 3

This GDB supports auto-downloading debuginfo from the following URLs:
<https://debuginfod.fedoraproject.org/>
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.

Breakpoint 1, _start () at lab09-3.asm:10
10      pop ecx          ; Извлекаем из стека в 'ecx' количество
(gdb) x/x $esp
0xffffcf00: 0x00000005
(gdb)
```

Рис. 2.20: Информация о брейкпоинтах

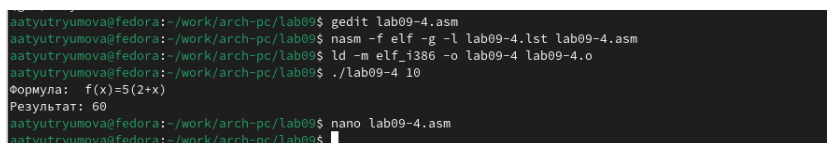
Посмотрела позиции стека, в которых располагаются аргументы программы (рис. 2.21).

```
(gdb) x/x $esp
0xffffcf00: 0x00000005
(gdb) x/s *(void**)(esp + 4)
0xffffd1a0: "/home/aatyutryumova/work/arch-pc/lab09/lab09-3"
(gdb) ^[[200~x/s *(void**)(esp + 8)~
Undefined command: "~". Try "help".
(gdb) x/s *(void**)(esp + 4)
0xffffd1a0: "/home/aatyutryumova/work/arch-pc/lab09/lab09-3"
(gdb) x/s *(void**)(esp + 8)
0xffffd1d0: "аргумент1"
(gdb) x/s *(void**)(esp + 12)
0xffffd1e7: "аргумент"
(gdb) x/s *(void**)(esp + 16)
0xffffd1f0: "2"
(gdb) x/s *(void**)(esp + 20)
0xffffd1f0: "аргумент 3"
(gdb) x/s *(void**)(esp + 24)
0x0: <error: Cannot access memory at address 0x0>
(gdb)
```

Рис. 2.21: Позиции стека и аргументы

3 Выполнение самостоятельной работы

Преобразовала программу из лабораторной работы №8 (Задание №1 для самостоятельной работы), реализовав вычисление значения функции $f(x)=5(2+x)$ как подпрограмму. (рис. 3.1).



```
aatyutryumova@fedora:~/work/arch-pc/lab09$ gedit lab09-4.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-4.lst lab09-4.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-4 lab09-4.o
aatyutryumova@fedora:~/work/arch-pc/lab09$ ./lab09-4 10
Формула: f(x)=5(2+x)
Результат: 60
aatyutryumova@fedora:~/work/arch-pc/lab09$ nano lab09-4.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$
```

Рис. 3.1: Выполнение программы

Листинг кода:

```
%include 'in_out.asm'

SECTION .data
    msg db "Результат: ",0
    formula db "Формула: f(x)=5(2+x)",0

SECTION .bss
    res: RESB 80

SECTION .text
global _start
```

```

_start:
    pop ecx          ; Извлекаем из стека в `ecx` количество
                     ; аргументов (первое значение в стеке)
    pop edx          ; Извлекаем из стека в `edx` имя программы
                     ; (второе значение в стеке)
    sub ecx,1        ; Уменьшаем `ecx` на 1 (количество
                     ; аргументов без названия программы)

next:
    cmp ecx,0h      ; проверяем, есть ли еще аргументы
    jz _end         ; если аргументов нет выходим из цикла
                     ; (переход на метку `_end`)
    pop eax          ; иначе извлекаем следующий аргумент из стека
    call atoi        ; преобразуем символ в число

    call _calcul
    add [res],eax     ; добавляем к промежуточной сумме
                     ; след. аргумент `esi=esi+eax`
    loop next        ; переход к обработке следующего аргумента

_end:
    mov eax, formula; вывод сообщения "Формула: "
    call sprintfLF
    mov eax, msg      ; вывод сообщения "Результат: "
    call sprintf
    mov eax, [res]     ; записываем сумму в регистр `eax`
    call iprintLF      ; печать результата

    call quit          ; завершение программы

```



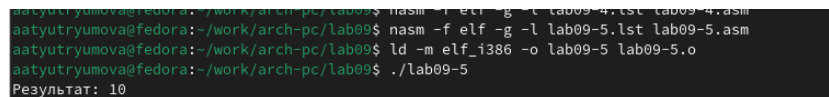
```

;-----
; Подпрограмма вычисления
; функции "f(x)=5(2+x)"

_calcul:
    add eax, 2      ; Прибавляем 2
    mov ebx, 5      ; ebx = 5
    mul ebx         ; Умножаем на 5
    ret

```

Проверила, что программа при запуске дает неверный результат (рис. 3.2)



```

aatyutryumova@fedora: ~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-5.lst lab09-5.asm
aatyutryumova@fedora: ~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-5 lab09-5.o
aatyutryumova@fedora: ~/work/arch-pc/lab09$ ./lab09-5
Результат: 10

```

Рис. 3.2: Выполнение программы

С помощью отладчика GDB, анализируя изменения значений регистров, определила ошибку и исправила ее (рис. 3.3)

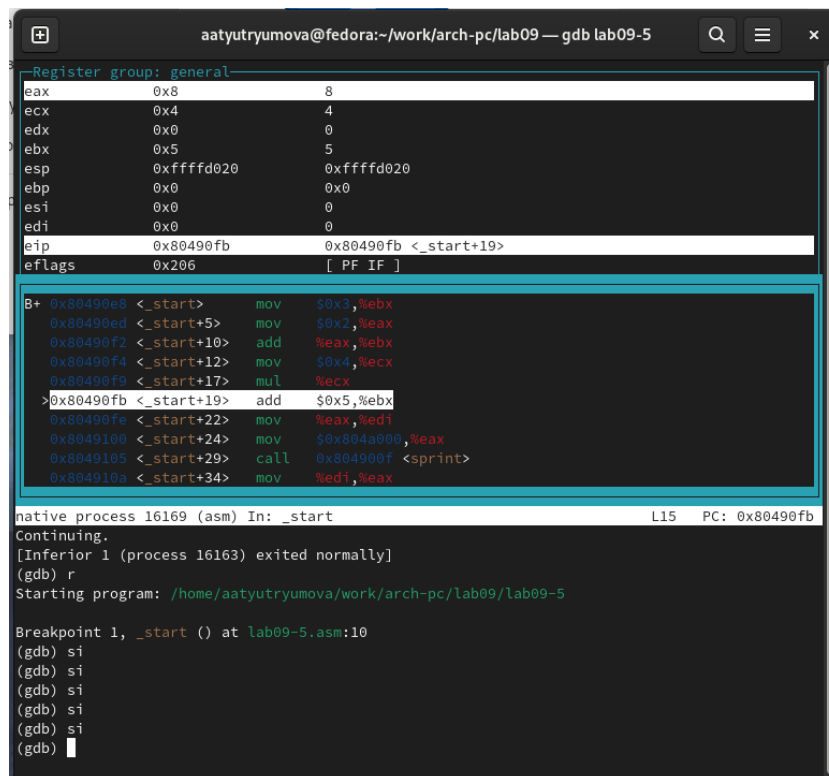


Рис. 3.3: Отладчика GDB

Листинг кода(исправленный):

```
%include 'in_out.asm'

SECTION .data
div: DB 'Результат: ',0

SECTION .text
GLOBAL _start
_start:

; ---- Вычисление выражения (3+2)*4+5
mov ebx,3
mov eax,2
add eax,ebx
```

```

mov ecx,4
mul ecx
add eax,5
mov edi,eax

; ---- Вывод результата на экран
mov eax,div
call sprint
mov eax,edi
call iprintLF
call quit

```

Проверила корректность исполнения программы (рис. 3.4)



```

(gdb) layout regs
aatyutryumova@fedora:~/work/arch-pc/lab09$ gedit lab09-5.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-5.lst lab09-5.asm
aatyutryumova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-5 lab09-5.o
aatyutryumova@fedora:~/work/arch-pc/lab09$ ./lab09-5
Результат: 25

```

Рис. 3.4: Выполнение программы

4 Выводы

Выполнив данную лабораторную работу, я обрела навыки написания программ с использованием подпрограмм. И ознакомилась с методами отладки при помощи GDB и его основными возможностями.