

Лабораторная работа №9

Понятие подпрограммы. Отладчик GDB.

Бочаров Андрей

Содержание

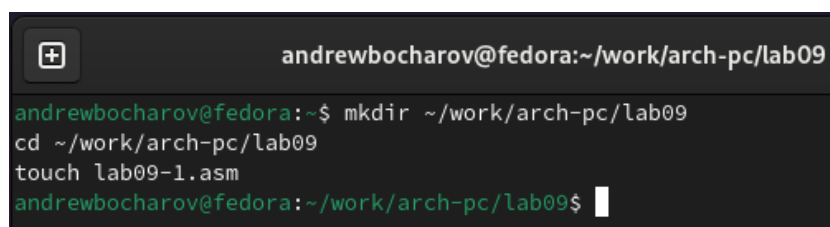
| | | |
|----------|--|-----------|
| 1 | Цель работы | 3 |
| 2 | Выполнение лабораторной работы | 4 |
| 3 | Выполнение самостоятельной работы | 14 |
| 4 | Выводы | 19 |

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

Создал и перешел в директорию для лабораторной работы. Создал файл lab9-1.asm (рис. 2.1).



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

Рис. 2.1: Папка для лабораторной работы

Переписал код с листинга 9.1 (рис. 2.2).



```
1 %include 'in_out.asm'
2 SECTION .data
3     msg: DB 'Введите x: ',0
4     result: DB '2x+7=',0
5 SECTION .bss
6     x: RESB 80
7     res: RESB 80
8 SECTION .text
9 GLOBAL _start
10    _start:
11
12 ;-----
13 ; Основная программа
```

Рис. 2.2: Листинг кода

Листинг 9.1:

```
%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

```

```

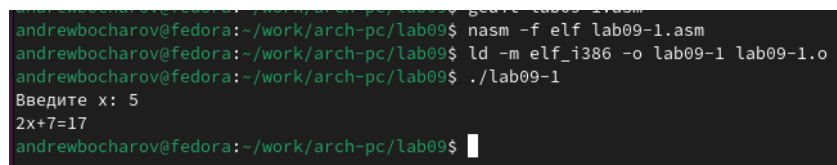
        call quit

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

_calcul:
        mov ebx,2
        mul ebx
        add eax,7
        mov [res],eax
ret ; выход из подпрограммы

```

Создал исполняемый файл и запустил его. (рис. 2.3).



```

andrewbocharov@fedora:~/work/arch-pc/lab09$ gedit lab09-1.asm
andrewbocharov@fedora:~/work/arch-pc/lab09$ nasm -f elf lab09-1.asm
andrewbocharov@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-1 lab09-1.o
andrewbocharov@fedora:~/work/arch-pc/lab09$ ./lab09-1
Введите x: 5
2x+7=17
andrewbocharov@fedora:~/work/arch-pc/lab09$

```

Рис. 2.3: Результат выполнения

Создал файл lab09-2.asm с текстом программы из листинга 9.2 (Программа печати сообщения Hello world!)

Листинг 9.2:

```

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

SECTION .text
        global _start

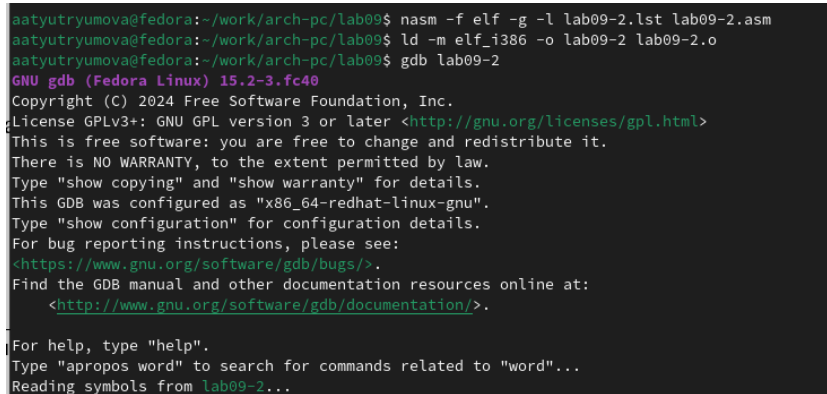
```

```

_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

```

Получил исполняемый файл. Для работы с GDB добавил в исполняемый файл отладочную информацию, трансляцией с ключом '-g'. Загрузил 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

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

```

(gdb) run
Starting program: /home/andrewbocharov/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 47.71 K separate debug info for system-supplied DSO at 0xf7ffc000
Hello, world!
[Inferior 1 (process 7582) exited normally]
(gdb)

```

Рис. 2.5: Результат выполнения

Для более подробного анализа программы установил брейкпоинт на метку `_start`, с которой начинается выполнение любой ассемблерной программы, и запустил её. (рис. 2.6).

```

(gdb) break _start
Breakpoint 1 at 0x08049000: file lab09-2.asm, line 9.
(gdb) run
Starting program: /home/andrewbocharov/work/arch-pc/lab09/lab09-2

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

```

Рис. 2.6: Добавление брейкпоинта

Посмотрел 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 $0x804a000,%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

Переключился 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

Разница в режимах состоит в порядке вывода регистра и его значения. Включил режим псевдографики для более удобного анализа программы. (рис. 2.9).

```
andrewbocharov@fedora:~/work/arch-pc/lab09 — gdb lab09-2
—Register group: general—
eax      0x0      0
ecx      0x0      0
edx      0x0      0
ebx      0x0      0
esp      0xffffcfe0 0xffffcfe0
ebp      0x0      0x0

B->0x08049000 <_start>  mov     eax,0x4
      0x08049005 <_start+5>  mov     ebx,0x1
      0x0804900a <_start+10> mov     ecx,0x804a000
      0x0804900f <_start+15> mov     edx,0x8
      0x08049014 <_start+20> int     0x80
      0x08049016 <_start+22> mov     eax,0x4

native process 7706 (asm) In: _start L9 PC: 0x08049000
(gdb) layout asm
(gdb) layout regs
(gdb)
```

Рис. 2.9: Результат работы

Определил адрес предпоследней инструкции (mov ebx,0x0) (рис. 2.10).

```
0x08049025 <_start+37> mov     edx,0x7
0x0804902a <_start+42> int     0x80
0x0804902c <_start+44> mov     eax,0x1
0x08049031 <_start+49> mov     ebx,0x0
0x08049036 <_start+54> int     0x80
0x08049038          add     BYTE PTR [eax],al
```

Рис. 2.10: Адрес предпоследней инструкции

Установил еще точку остановки по адресу инструкции. (рис. 2.11)

```

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

```

Рис. 2.11: Установка брейкпоинта

Посмотрел информацию о всех установленных брейкпоинтах. (рис. 2.12).

```

(gdb) i b
Num    Type           Disp Enb Address      What
1      breakpoint    keep y 0x8049000 lab09-2.asm:9
       breakpoint already hit 1 time
2      breakpoint    keep y 0x8049031 lab09-2.asm:20
(gdb)

```

Рис. 2.12: Информация о брейкпоинтах

Выполнил 5 инструкций с помощью команды si и проследил за изменением значений регистров. Изменяются значения регистров eax и ebx (рис. 2.13)

The screenshot shows the GDB interface with the 'Register group: general' window. The registers and their values are:

| Register | Type | Value |
|----------|------------|------------|
| eax | 0x4 | 4 |
| ecx | 0x804a000 | 134520832 |
| edx | 0x8 | 8 |
| ebx | 0x1 | 1 |
| esp | 0xffffcfe0 | 0xffffcfe0 |
| ebp | 0x0 | 0x0 |

Below the register window, the assembly instructions are listed:

```

0x804900f <_start+15> mov    edx,0x8
0x8049014 <_start+20> int     0x80
0x8049016 <_start+22> mov    eax,0x4
>0x804901b <_start+27> mov    ebx,0x1
0x8049020 <_start+32> mov    ecx,0x804a008
0x8049025 <_start+37> mov    edx,0x7

```

The bottom of the window shows the command prompt with the following commands entered:

```

(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) s
(gdb) si
(gdb)

```

Рис. 2.13: Значения регистров

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

```

(gdb) x/1sb &msg1
0x804a000 <msg1>:      "Hello, "
(gdb) x/1sb 0x804a800
0x804a800:      ""
(gdb) x/1sb 0x804a008
0x804a008 <msg2>:      "world!\n\034"
(gdb)

```

Рис. 2.14: Значение переменных msg1 и msg2

Изменил первый символ переменной msg1 (рис. 2.15).

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

Рис. 2.15: Переменная msg1

Изменил первый символ переменной msg2 (рис. 2.16).

```
(gdb) set *(char*)&msg2='t'
(gdb) x/1sb &msg2
0x804a008 <msg2>: "torld!\n\034"
```

Рис. 2.16: Переменная msg2

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

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

Рис. 2.17: Значение регистра ebx

Завершил выполнение программы с помощью команды continue и вышел из GDB с помощью команды quit (рис. 2.18).

```
native process 7706 (asm) In: _start

Breakpoint 2, _start () at lab09-2.asm:20
(gdb) quit
A debugging session is active.

      Inferior 1 [process 7706] will be killed.

Quit anyway? (y or n)
```

Рис. 2.18: Завершение работы

Скопировал 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. Создал исполняемый файл и загрузил в gdb программу с аргументами и ключом `-args`. (рис. 2.19).

```
andrewbocharov@fedora:~/work/arch-pc/lab09$ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab09-3.asm
andrewbocharov@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-3.lst lab09-3.asm
ld -m elf_i386 -o lab09-3 lab09-3.o
andrewbocharov@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

Установил точку останова перед первой инструкцией в программе и запустил ее (рис. 2.20).

```
(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab09-3.asm, line 10.
(gdb) run
Starting program: /home/andrewbocharov/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)
```

Рис. 2.20: Информация о брейпоинтах

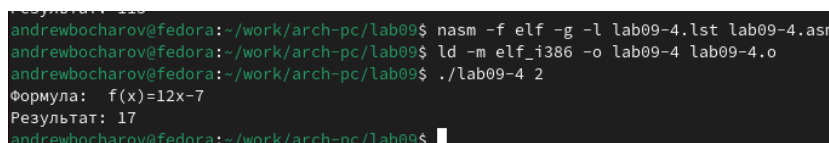
Посмотрел позиции стека, в которых располагаются аргументы программы (рис. 2.21).

```
(gdb) x/x $esp
0xffffcfa0: 0x00000005
(gdb) x/s *(void**)(esp + 4)
0xffffd162: "/home/andrewbocharov/work/arch-pc/lab09/lab09-3"
(gdb) x/s *(void**)(esp + 8)
0xffffd192: "аргумент1"
(gdb) x/s *(void**)(esp + 12)
0xffffd1a4: "аргумент"
(gdb) x/s *(void**)(esp + 16)
0xffffd1b5: "2"
(gdb) x/s *(void**)(esp + 20)
0xffffd1b7: "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

Преобразовал программу из лабораторной работы №8 (Задание №1 для самостоятельной работы), реализовав вычисление значения функции $f(x)=12x-7$ как подпрограмму. (рис. 3.1).



```
andrewbocharov@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-4.lst lab09-4.asm
andrewbocharov@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-4 lab09-4.o
andrewbocharov@fedora:~/work/arch-pc/lab09$ ./lab09-4 2
Формула: f(x)=12x-7
Результат: 17
andrewbocharov@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)=12x-7",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)
    mov esi, 0       ; Используем `esi` для хранения
                     ; промежуточных сумм

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 esi,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, esi       ; записываем сумму в регистр `eax`
    call iprintLF     ; печать результата
    call quit         ; завершение программы

```

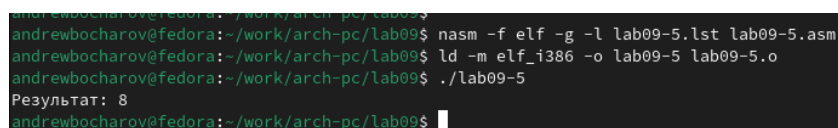
```

;-----
; Подпрограмма вычисления
; функции "f(x)=12x-7"

_calcul:
    mov ebx, 12      ; ebx = 12
    mul ebx          ; Умножаем на 12
    sub eax, 7       ; вычитаем 7
    ret

```

Проверил, что программа из листинга 9.3 при запуске дает неверный результат (рис. 3.2)



```

andrewbocharov@fedora: ~/work/arch-pc/lab09$
andrewbocharov@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-5.lst lab09-5.asm
andrewbocharov@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-5 lab09-5.o
andrewbocharov@fedora:~/work/arch-pc/lab09$ ./lab09-5
Результат: 8
andrewbocharov@fedora:~/work/arch-pc/lab09$

```

Рис. 3.2: Выполнение программы

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


```

+
andrewbocharov@fedora:~/w

Register group: general
eax      0x8      8
edx      0x0      0
esp      0xffffcfe0 0xffffcfe0
esi      0x0      0
eip      0x80490fb 0x80490fb <_start+1
cs       0x23     35
ds       0x2b     43
fs       0x0      0

B+ 0x80490e8 <_start>    mov    $0x3,%ebx
0x80490ed <_start+5>    mov    $0x2,%eax
0x80490f2 <_start+10>   add    %eax,%ebx
0x80490f4 <_start+12>   mov    $0x4,%ecx
0x80490f9 <_start+17>   mul    %ecx
>0x80490fb <_start+19>  add    $0x5,%ebx
0x80490fe <_start+22>   mov    %eax,%edi
0x8049100 <_start+24>   mov    $0x804a000,%eax
0x8049105 <_start+29>   call   0x804900f <sprint>
0x804910a <_start+34>   mov    %edi,%eax
0x804910c <_start+36>   call   0x8049086 <iprintLF>

native process 10262 (asm) In: _start
<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 enab

Breakpoint 1, _start () at lab09-5.asm:11
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb)

```

Рис. 3.3: Отладчика GDB

Листинг кода(исправленный):

```

#include 'in_out.asm'

SECTION .data
div: DB 'Результат: ',0

```

```

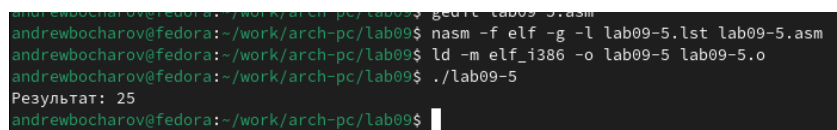
SECTION .text
GLOBAL _start
_start:

; ---- 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

```

Проверил корректность исполнения программы (рис. 3.4)



```

andrewbocharov@fedora:~/work/arch-pc/lab09$ gcc -c lab09-5.asm
andrewbocharov@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-5.lst lab09-5.asm
andrewbocharov@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-5 lab09-5.o
andrewbocharov@fedora:~/work/arch-pc/lab09$ ./lab09-5
Результат: 25
andrewbocharov@fedora:~/work/arch-pc/lab09$

```

Рис. 3.4: Выполнение программы

4 Выводы

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