

Выполнение лабораторной работы №9

Дисциплина: Архитектура компьютера

Ефремова Полина Александровна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Реализация подпрограмм в NASM	8
4.2	Отладка программ с помощью GDB	15
4.2.1	Добавление точек останова	24
4.2.2	Работа с данными программы в GDB	25
4.3	Задание для самостоятельной работы	30
5	Выводы	38
	Список литературы	39

Список иллюстраций

4.1	Создание файла	8
4.2	Ввод программы	9
4.3	Запуск программы	11
4.4	Ввод программы	12
4.5	Проверка программы	13
4.6	Создание файла с текстом программы	16
4.7	Трансляция программы	18
4.8	Загрузка в gdb	19
4.9	Использование команды run	20
4.10	Установка брекпоинта	21
4.11	Просмотр кода программы	22
4.12	Программы с синтаксисом Intel	23
4.13	Режим псевдографики	24
4.14	установка еще одного breakpoint	24
4.15	Использование программы si	25
4.16	info registers	26
4.17	Содержимое переменных	26
4.18	Изменение содержимого переменных	27
4.19	Вывод значений	27
4.20	Изменение значений	28
4.21	Новый файл	29
4.22	работа с файлом lab09-3	30
4.23	задание 1 ср	31
4.24	Запуск программы с ошибкой	33
4.25	Нахождение ошибки	35
4.26	Проверка	37

Список таблиц

1 Цель работы

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

2 Задание

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

3 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, отработывает, но не даёт желаемого результата; • ошибки в процессе выполнения — не обнаруживаются при трансляции и вызывают прерывание выполнения программы (например, это ошибки, связанные с переполнением или делением на ноль).

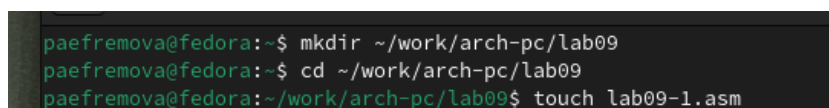
Второй этап — поиск местонахождения ошибки. Некоторые ошибки обнаружить довольно трудно. Лучший способ найти место в программе, где находится ошибка, это разбить программу на части и произвести их отладку отдельно друг от друга.

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.

4 Выполнение лабораторной работы

4.1 Реализация подпрограмм в NASM

1. Создаю файл в папке для программ лабораторной работы №9 (рис. 4.1).



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

Рис. 4.1: Создание файла

2. В созданный файл вставляю программу с листинга 9.1 (рис. 4.2).


```
GNU nano 7.2 /home/paefremova/work/arch-pc/lab09/lab09-1.asm Изменён
#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

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Позиция
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^J Выровнять ^/_ К строке
```

Рис. 4.2: Ввод программы

Листинг 9.1. Пример программы с использованием вызова подпрограммы

```
%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

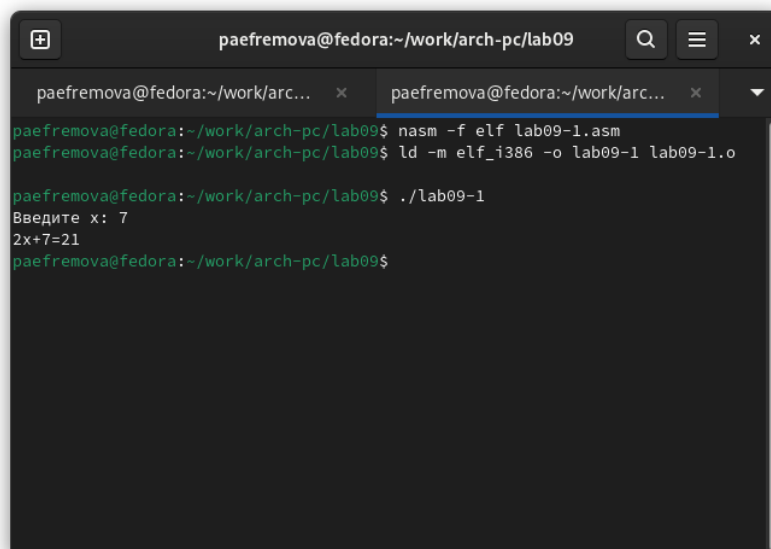
mul ebx

add eax,7

mov [res],eax

ret ; выход из подпрограммы

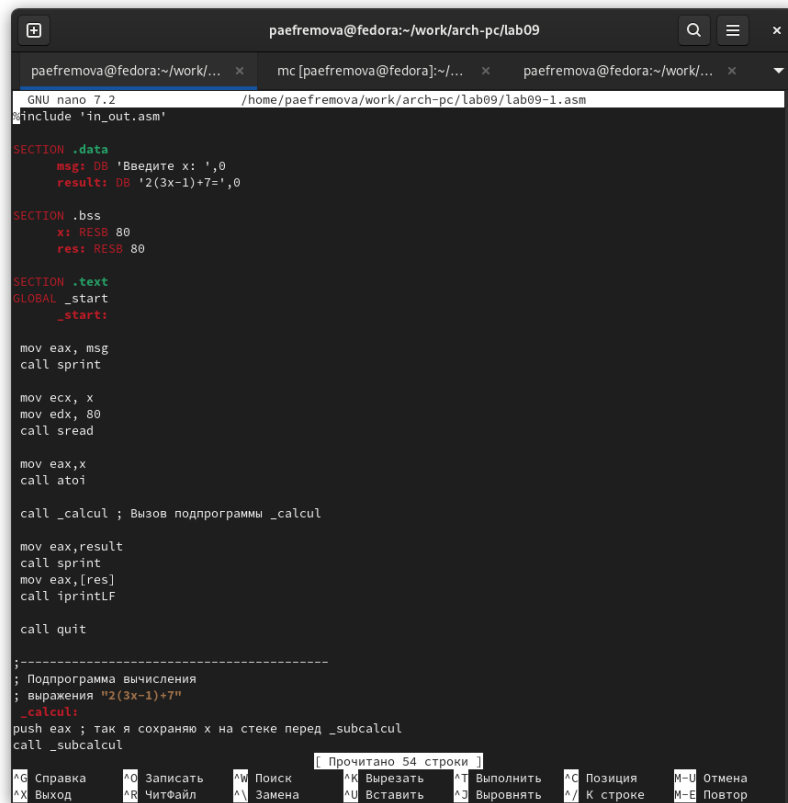
3. Теперь запускаю программу и проверяю ее работу (рис. 4.3).



```
paefremova@fedora:~/work/arch-pc/lab09
paefremova@fedora:~/work/arch-pc/lab09$ nasm -f elf lab09-1.asm
paefremova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-1 lab09-1.o
paefremova@fedora:~/work/arch-pc/lab09$ ./lab09-1
Введите x: 7
2x+7=21
paefremova@fedora:~/work/arch-pc/lab09$
```

Рис. 4.3: Запуск программы

4. Меняю код программы так, чтобы вычислить $f(g(x))$, 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. (рис. 4.4).



```
GNU nano 7.2 /home/paefremova/work/arch-pc/lab09/lab09-1.asm
#include "in_out.asm"

SECTION .data
msg: DB 'Введите x: ',0
result: DB '2(3x-1)+7=',0

SECTION .bss
x: RESB 80
res: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax, msg
call sprint

mov ecx, x
mov edx, 80
call sread

mov eax, x
call atoi

call _calcul ; Вызов подпрограммы _calcul

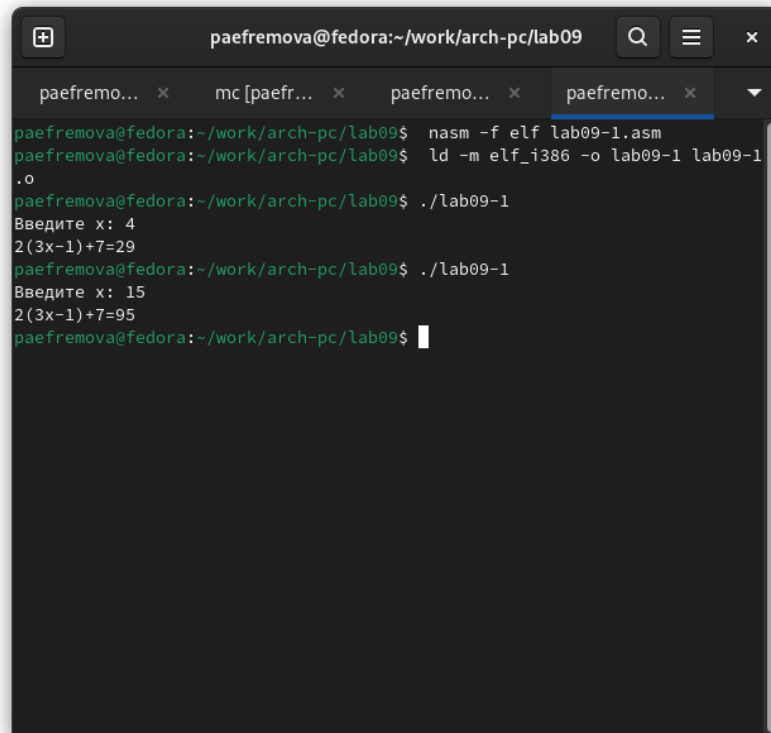
mov eax, result
call sprint
mov eax, [res]
call iprintLF

call quit

;-----
; Подпрограмма вычисления
; выражения "2(3x-1)+7"
_calcul:
push eax ; так я сохраняю x на стеке перед _subcalcul
call _subcalcul
```

Рис. 4.4: Ввод программы

5. Теперь проверю программу на точность работы (рис. 4.5).



The screenshot shows a terminal window with the title bar 'paefremova@fedora:~/work/arch-pc/lab09'. The terminal contains the following commands and output:

```
paefremova@fedora:~/work/arch-pc/lab09$ nasm -f elf lab09-1.asm
paefremova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-1 lab09-1.o
paefremova@fedora:~/work/arch-pc/lab09$ ./lab09-1
Введите x: 4
2(3x-1)+7=29
paefremova@fedora:~/work/arch-pc/lab09$ ./lab09-1
Введите x: 15
2(3x-1)+7=95
paefremova@fedora:~/work/arch-pc/lab09$
```

Рис. 4.5: Проверка программы

Ниже прикрепляю текст измененной программы:

```
%include 'in_out.asm'

SECTION .data
    msg: DB 'Введите x: ',0
    result: DB '2(3x-1)+7=',0

SECTION .bss
    x: RESB 80
    res: RESB 80

SECTION .text
GLOBAL _start
```

```

_start:

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

; выражения "2(3x-1)+7"
_calcul:
push eax ; так я сохраняю x на стеке перед _subcalcul
call _subcalcul

```

```

mov ebx,2
mul ebx      ; 2 * (3x - 1)
add eax,7    ; 2(3x - 1) + 7
mov [res],eax ; Сохраняю результат в res
pop eax ; Получаю результат из _subcalcul
ret

;-----
; Подпрограмма вычисления 3x - 1
_subcalcul:
mov ebx,3
mul ebx      ; 3x
sub eax,1    ; 3x - 1
ret

```

4.2 Отладка программ с помощью GDB

6. Создаю файл lab09-2.asm с текстом программы из Листинга 9.2. (рис. 4.6).

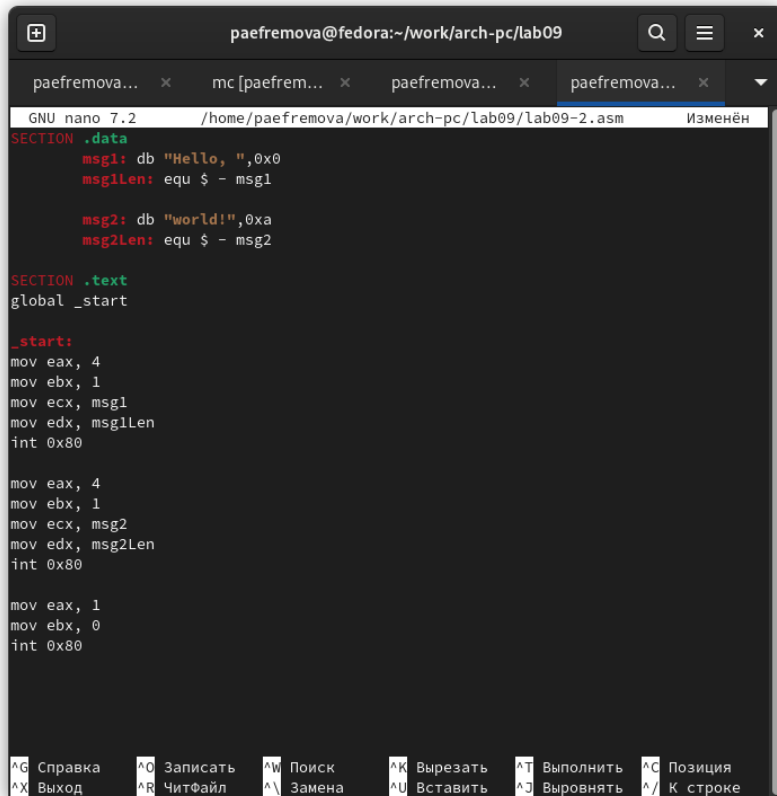


Рис. 4.6: Создание файла с текстом программы

Текст программы:

```
SECTION .data
    msg1: db "Hello, ",0x0
    msg1Len: equ $ - msg1

    msg2: db "world!",0xa
    msg2Len: equ $ - msg2

SECTION .text
global _start

_start:
```

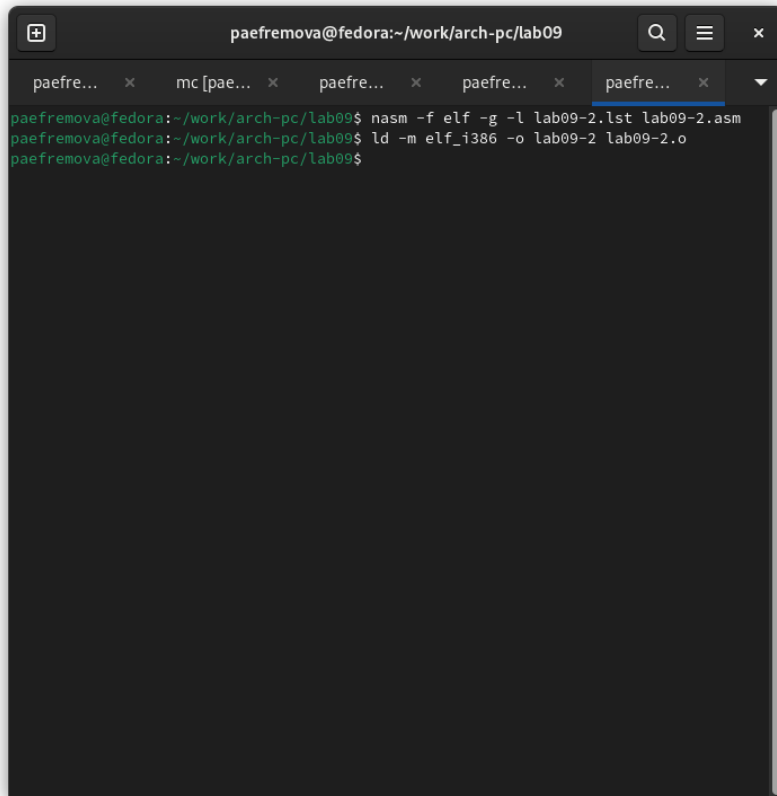


```
mov eax, 4
mov ebx, 1
mov ecx, msg1
mov edx, msg1Len
int 0x80
```

```
mov eax, 4
mov ebx, 1
mov ecx, msg2
mov edx, msg2Len
int 0x80
```

```
mov eax, 1
mov ebx, 0
int 0x80
```

7. Для работы с GDB в исполняемый файл необходимо добавить отладочную информацию, для этого трансляцию программ необходимо проводить с ключом '-g'. (рис. 4.7).

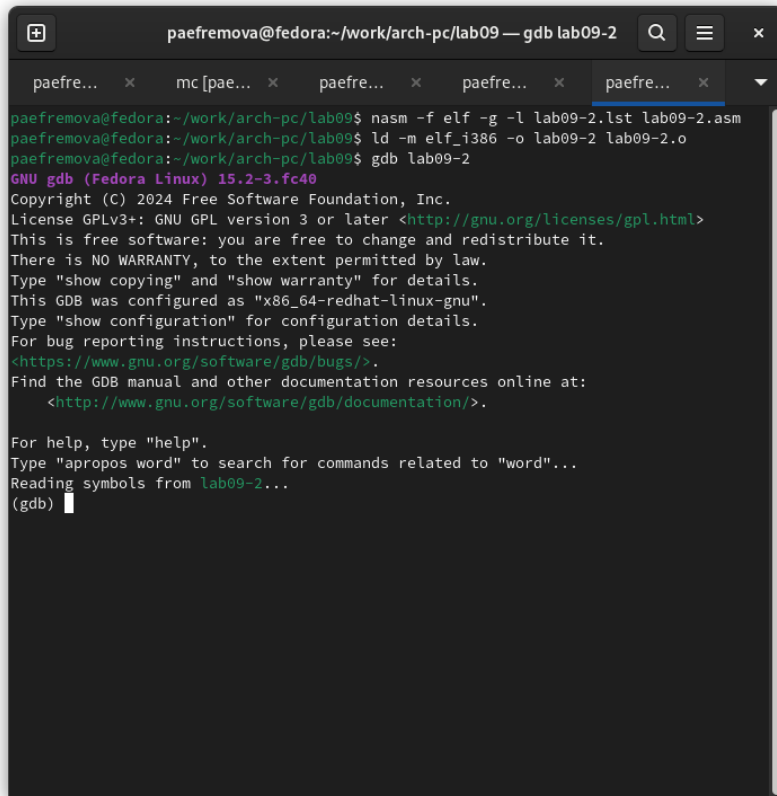


A terminal window titled 'paefremova@fedora:~/work/arch-pc/lab09'. The window contains the following commands and their outputs:

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

Рис. 4.7: Трансляция программы

8. Загружаю исполняемый файл в отладчик gdb. (рис. 4.8).

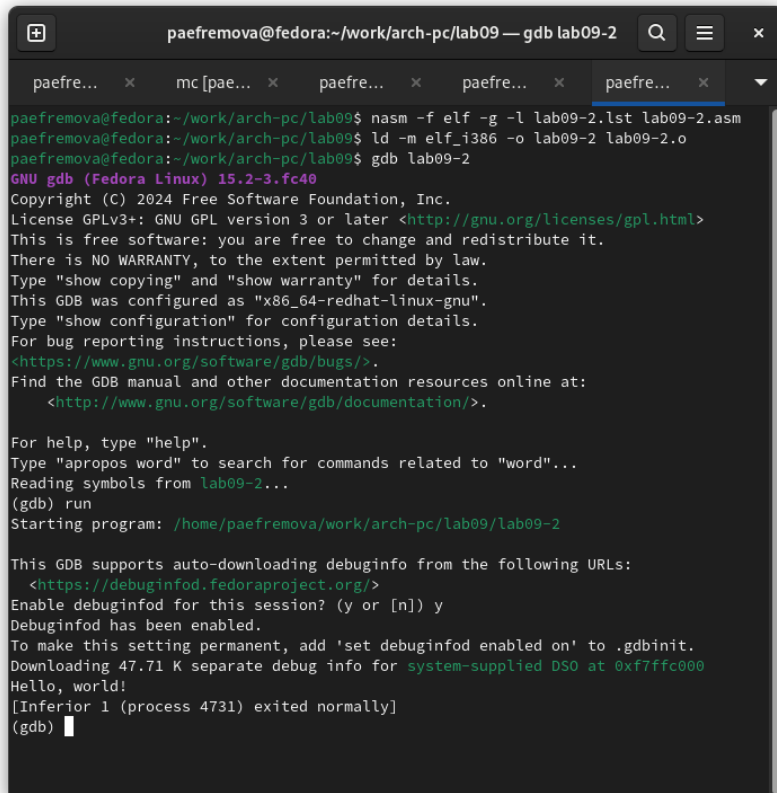


```
paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefre... x mc[pae... x paefre... x paefre... x paefre... x
paefremova@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-2.lst lab09-2.asm
paefremova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-2 lab09-2.o
paefremova@fedora:~/work/arch-pc/lab09$ 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...
(gdb)
```

Рис. 4.8: Загрузка в gdb

9. Проверяю работу программы, запустив ее в оболочке GDB с помощью команды `run` (сокращённо `r`) (рис. 4.9).



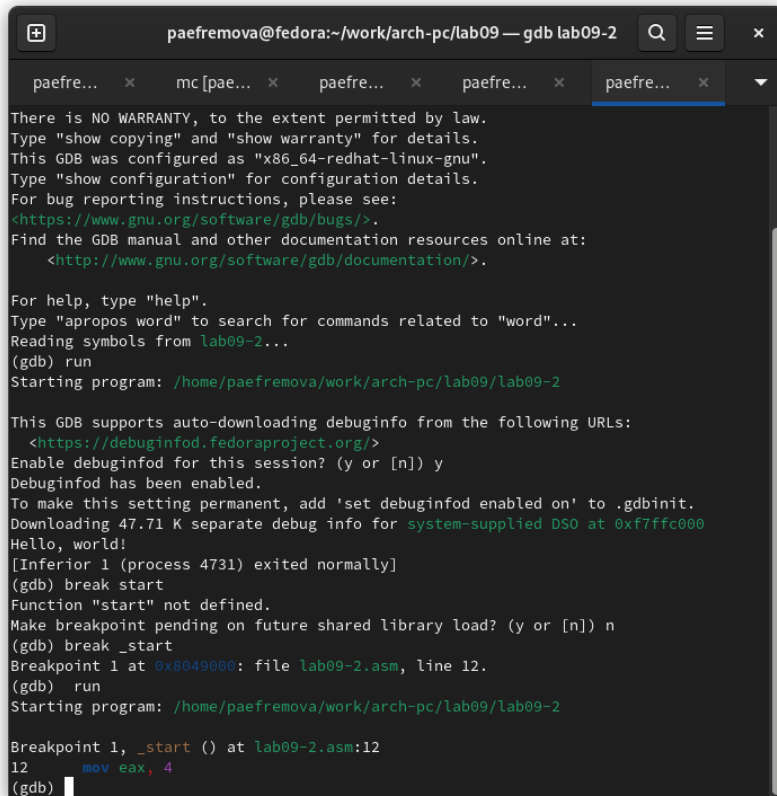
```
paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefre... x mc[pae... x paefre... x paefre... x paefre... x
paefremova@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-2.lst lab09-2.asm
paefremova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-2 lab09-2.o
paefremova@fedora:~/work/arch-pc/lab09$ 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...
(gdb) run
Starting program: /home/paefremova/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 4731) exited normally]
(gdb)
```

Рис. 4.9: Использование команды run

10. Для более подробного анализа программы устанавливаю брейкпоинт на метку `_start`, с которой начинается выполнение любой ассемблерной программы, и запускаю её. (рис. 4.10).



The screenshot shows a terminal window titled "paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2". The window contains the following text:

```
paefre... x mc[pae... x paefre... x paefre... x paefre... x
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/>.

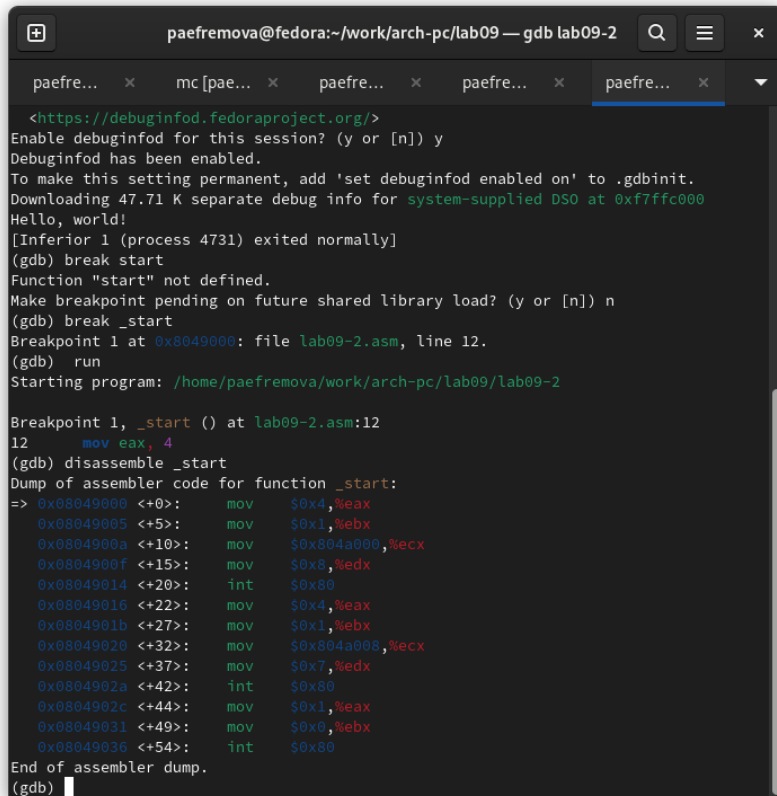
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab09-2...
(gdb) run
Starting program: /home/paefremova/work/arch-pc/lab09/lab09-2

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 4731) exited normally]
(gdb) break start
Function "start" not defined.
Make breakpoint pending on future shared library load? (y or [n]) n
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab09-2.asm, line 12.
(gdb) run
Starting program: /home/paefremova/work/arch-pc/lab09/lab09-2

Breakpoint 1, _start () at lab09-2.asm:12
12      mov eax, 4
(gdb) |
```

Рис. 4.10: Установка брекпоинта

11. Посмотрю дисассимилированный код программы с помощью команды `disassemble`, начиная с метки `_start` (рис. 4.11).

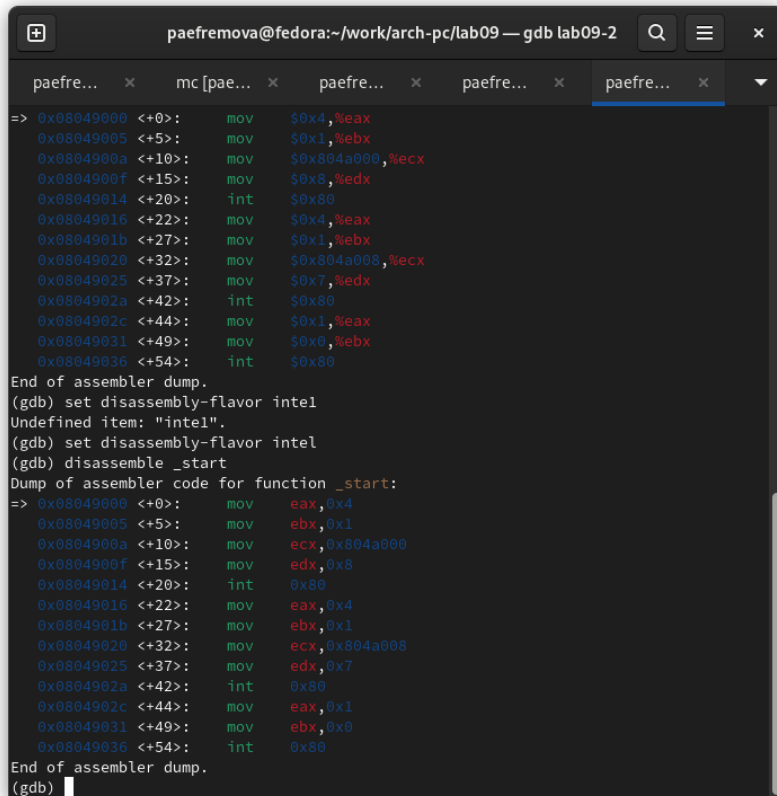


```
paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefre... x mc[pae... x paefre... x paefre... x paefre... x
<https://debuginfod.fedoraproject.org/>
Enable debuginfod for this session? (y or [n]) y
Debuginfod has been enabled.
To make this setting permanent, add 'set debuginfod enabled on' to .gdbinit.
Downloading 47.71 K separate debug info for system-supplied DSO at 0xf7ffc000
Hello, world!
[Inferior 1 (process 4731) exited normally]
(gdb) break start
Function "start" not defined.
Make breakpoint pending on future shared library load? (y or [n]) n
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab09-2.asm, line 12.
(gdb) run
Starting program: /home/paefremova/work/arch-pc/lab09/lab09-2

Breakpoint 1, _start () at lab09-2.asm:12
12      mov eax, 4
(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. 4.11: Просмотр кода программы

12. Переключаюсь на отображение команд с Intel'овским синтаксисом, вводя команду `set disassembly-flavor intel` (рис. 4.12).



```
paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefre... x mc [pae... x paefre... x paefre... x paefre... x
=> 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) set disassembly-flavor intel
Undefined item: "intel".
(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. 4.12: Программы с синтаксисом Intel

Теперь посмотрим на различия отображения синтаксиса машинных команд в режимах АТТ и Intel:

Порядок операндов: Intel: назначение, источник; АТТ: источник, назначение.

Регистры: Intel: eax; АТТ: %eax.

Размер операндов:(ТТ - размер операндов указывается явно с помощью суффиксов, Intel - Размер операндов неявно определяется контекстом.

Константы: Intel: mov eax, 10; АТТ: mov \$10, %eax.

Главное отличие — порядок операндов.

13. Включу режим псевдографики для более удобного анализа программы (рис. 4.13).

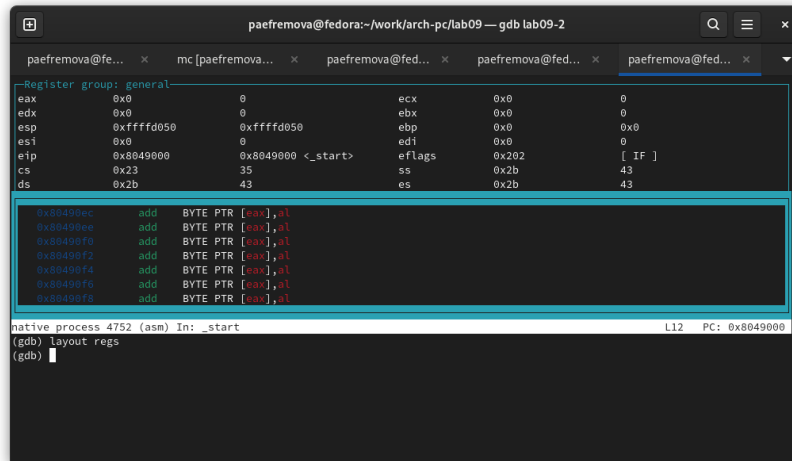


Рис. 4.13: Режим псевдографики

4.2.1 Добавление точек останова

С помощью `i b` (info breakpoints) убеждаюсь в том, что создала первую точку `break`.

14. Теперь устанавливаю еще одну точку останова по адресу инструкции и ввожу `i b`, которая показывает, что обе точки останова созданы. (рис. 4.14).

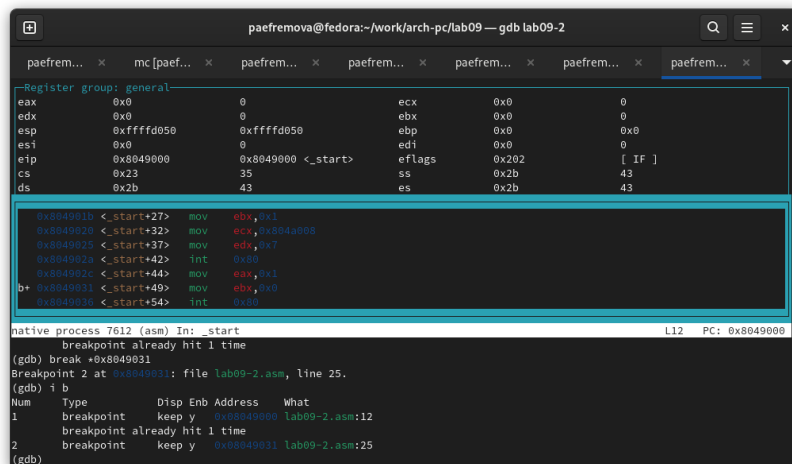
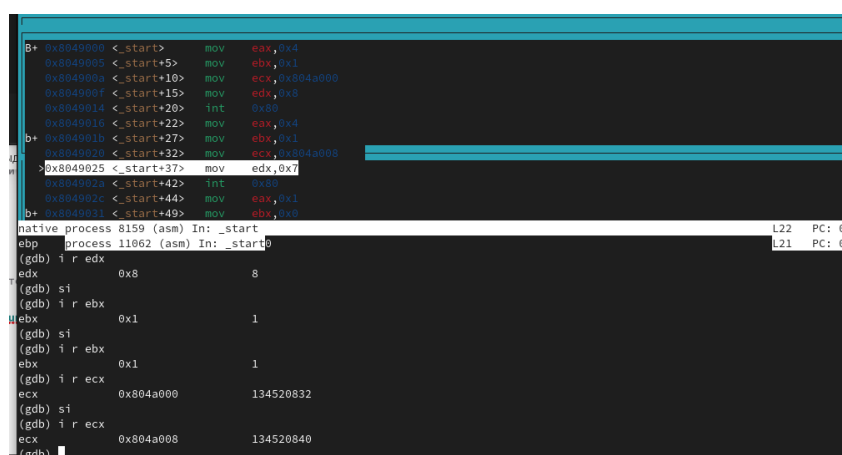


Рис. 4.14: установка еще одного breakpoint

4.2.2 Работа с данными программы в GDB

15. Выполнила 5 инструкций с помощью команды `stepi` (или `si`) и проследила за изменением значений регистров. Поменялось значение регистра `ecx,0x804a008`. Я поняла изменения так, каждый раз выводится значение последнего обработанного регистра (крайнего). Если ввести `i` для регистра, который находится после текущего положения и (иногда) в текущем положении, то значение регистра будет засчитано как 0. Ниже прикрепила скрин того, как выполняла это. (рис. 4.15).



```
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
0x804901b <.start+27>   mov     ebx,0x1
0x804901e <.start+32>   mov     ecx,0x804a008
>0x8049025 <.start+37>   mov     edx,0x7
0x804902a <.start+42>   int     0x80
0x804902c <.start+44>   mov     eax,0x1
b+ 0x8049031 <.start+49>   mov     ebx,0x8

native process 8159 (asm) In: _start      L22  PC: 6
ebp    process 11062 (asm) In: _start0    L21  PC: 6
(gdb) i r edx                            8
edx    0x8
(gdb) si
(gdb) i r ebx                            1
ebx    0x1
(gdb) si
(gdb) i r ebx                            1
ebx    0x1
(gdb) i r ecx                            134520832
ecx    0x804a000
(gdb) si
(gdb) i r ecx                            134520840
ecx    0x804a008
(gdb)
```

Рис. 4.15: Использование программы `si`

16. Просмотр содержимого регистров. (рис. 4.16).

```

paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefremov... x mc[paefre... x paefremov... x paefremov... x paefremov... x paefremov... x paefremov... x
eax      0x4      4      ecx      0x804a008      134520840
ebx      0x0      0      ebx      0x0      0
esi      0x804902a      0x804902a <_start+42>      edi      0x002      0 [ IF ]
eip      0x804902a      0x804902a <_start+42>      eflags      0x202      [ IF ]
cs       0x23      35      ss       0x2b      43
ds       0x2b      43      es       0x2b      43
fs       0x0      0      gs       0x0      0

0x8049000 <_start> mov     edi,edx
0x8049000 <_start> mov     eax,0x4
native process 11512 (asm) In: _start
0x804900a <_start+10> mov     ecx,0x804a008
0x804900f <_start+15> mov     edx,0x0
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
0x804902a <_start+42> int     0x80
0x804902c <_start+44> mov     ecx,0x1
0x8049031 <_start+49> mov     ebx,0x0
(gdb) process 11062 (asm) In: _start
(gdb) process 11512 (asm) In: _start
L22 PC: 0x804902a
L12 PC: 0x8049000
--Type <RET> for more, q to quit, c to continue without paging--

```

Рис. 4.16: info registers

17. Смотрю содержимое переменных по имени и по адресу (рис. 4.17).

```

paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefre... x mc[pa... x paefre... x paefre... x paefre... x paefre... x paefre... x
Register group: general
eax      0x0      0      ecx      0x0      0
edx      0x0      0      ebx      0x0      0
esp      0xffffd050      0xffffd050      ebp      0x0      0x0
esi      0x0      0      edi      0x0      0
eip      0x8049000      0x8049000 <_start>      eflags      0x202      [ IF ]
cs       0x23      35      ss       0x2b      43
ds       0x2b      43      es       0x2b      43
fs       0x0      0      gs       0x0      0

0x804900f <_start+15> mov     edx,0x0
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
0x804902a <_start+42> int     0x80
0x804902c <_start+44> mov     ecx,0x1
native process 11512 (asm) In: _start
L12 PC: 0x8049000
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
(gdb) x/lb &msg1
0x804a000 <msg1>: "Hello, "
(gdb) x/lb 0x804a008
0x804a008 <msg2>: "world!\n@34"
(gdb) 

```

Рис. 4.17: Содержимое переменных

18. Меняю содержимое переменных по имени и по адресу. (рис. 4.18).

```

paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefre... x mc[pa... x paefre... x paefre... x paefre... x paefre... x paefre... x
Register group: general
eax 0x0 0 ecx 0x0 0
edx 0x0 0 ebx 0x0 0
esp 0xffffd050 0xffffd050 ebp 0x0 0
esi 0x0 0 edi 0x0 0
eip 0x8049000 0x8049000 <_start> eflags 0x202 [ IF ]
cs 0x23 35 ss 0x2b 43
ds 0x2b 43 es 0x2b 43
fs 0x0 0 gs 0x0 0

0x804900f <_start+15> mov     edi,0x0
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
0x804902a <_start+42> int     0x80
0x804902c <_start+44> mov     eax,0x1

native process 11512 (asm) In: _start
L12 PC: 0x8049000
(gdb) set {char}&msg1= 'h'
No symbol "msg1" in current context.
(gdb) set {char}&msg1= 'h'
(gdb) x/1sb &msg1
0x804a000 <msg1>:  "hello, "
(gdb) set {char}&msg2= 'x'
'msg2' has unknown type; cast it to its declared type
(gdb) set {char}&msg2= 'x'
(gdb) x/1sb &msg2
0x804a008 <msg2>:  "xorld!\n\034"
(gdb)

```

Рис. 4.18: Изменение содержимого переменных

19. Вывожу в различных форматах значение регистра edx. (рис. 4.19).

```

paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2
paefre... x mc[pa... x paefre... x paefre... x paefre... x paefre... x paefre... x
Register group: general
eax 0x4 4 ecx 0x804a000 134520832
edx 0x8 8 ebx 0x1 1
esp 0xffffd050 0xffffd050 ebp 0x0 0
esi 0x0 0 edi 0x0 0
eip 0x8049014 0x8049014 <_start+2> eflags 0x202 [ IF ]
cs 0x23 35 ss 0x2b 43
ds 0x2b 43 es 0x2b 43
fs 0x0 0 gs 0x0 0

B+ 0x8049000 <_start> mov     eax,0x4
0x8049005 <_start+5> mov     ebx,0x1
0x804900a <_start+10> mov     ecx,0x804a008
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

native process 12142 (asm) In: _start
L16 PC: 0x8049014
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) p/t $edx
$3 = 1000
(gdb) p/s $edx
$4 = 8
(gdb) p/x $edx
$5 = 0x8
(gdb)

```

Рис. 4.19: Вывод значений

20. С помощью команды set изменяю значение регистра ebx (рис. 4.20).

The screenshot shows a GDB terminal window with the title bar 'paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-2'. The window contains several tabs, with the active one showing the 'Register group: general' section. This section displays a table of registers and their values:

Register	Value	Register	Value	Register	Value
eax	0x4	ecx	0x804a000	134520832	
edx	0x8	ebx	0x2	2	
esp	0xffffd050	ebp	0x0	0x0	
esi	0x0	edi	0x0	0x0	
eip	0x8049014	<_start+2>	eflags	0x202	[IF]
cs	0x23	35	ss	0x2b	43
ds	0x2b	43	es	0x2b	43
fs	0x0	0	gs	0x0	0

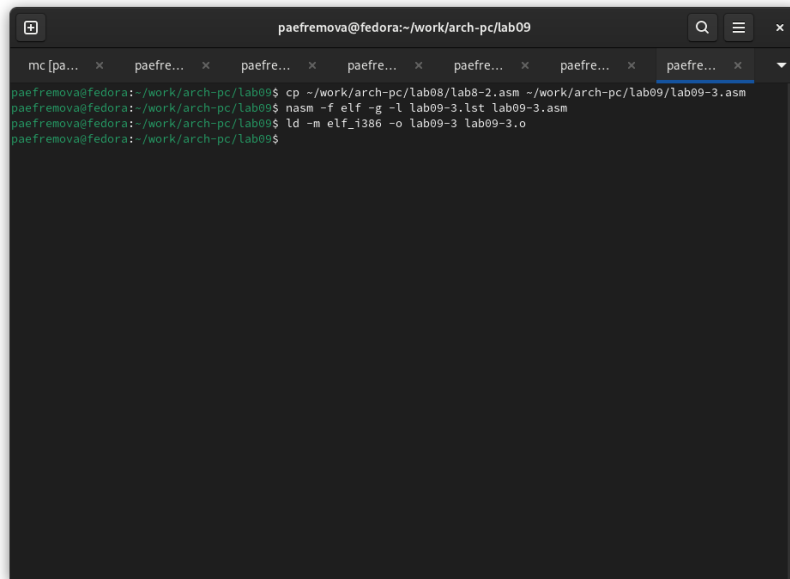
Below the register list, the assembly code is displayed, starting with 'B+ 0x8049008 <_start>'. The code includes several 'mov' instructions and an 'int' instruction. The current instruction being executed is 'int 0x80' at address '0x8049014'. The bottom of the window shows the GDB prompt '(gdb)' and the command 'p/s \$ebx' being entered, with the output '54 = 8'.

Рис. 4.20: Изменение значений

Команда `p/s $ebx` не имеет единственного “правильного” вывода. Результат полностью зависит от того, что содержится в регистре `ebx`. Только если `ebx` указывает на действительный нуль-терминированный массив символов в памяти, вы увидите осмысленный вывод.

Завершаю выполнение программы с помощью команды `continue` (сокращенно `c`) и выхожу из GDB с помощью команды `quit` (сокращенно `q`)

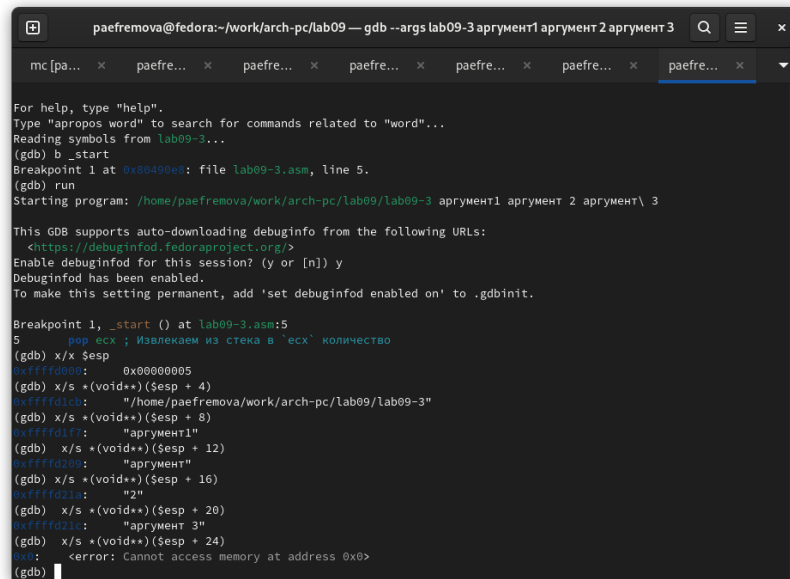
21. Копирую файл `lab8-2.asm`, созданный при выполнении лабораторной работы №8, с программой выводящей на экран аргументы командной строки (Листинг 8.2) в файл с именем `lab09-3.asm`. А также создаю исполняемый файл. (рис. 4.21).

A screenshot of a terminal window titled 'paefremova@fedora:~/work/arch-pc/lab09'. The window has several tabs at the top, with the active one labeled 'paefre...'. The terminal shows the following commands and their outputs:

```
paefremova@fedora:~/work/arch-pc/lab09$ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab09-3.asm
paefremova@fedora:~/work/arch-pc/lab09$ nasm -f elf -g -l lab09-3.lst lab09-3.asm
paefremova@fedora:~/work/arch-pc/lab09$ ld -m elf_1386 -o lab09-3 lab09-3.o
paefremova@fedora:~/work/arch-pc/lab09$
```

Рис. 4.21: Новый файл

22. Загрузите исполняемый файл в отладчик, указав аргументы. Затем установим точку останова перед первой инструкцией в программе и запустим. Адрес вершины стека храниться в регистре `esp` и по этому адресу располагается число равное количеству аргументов командной строки (включая имя программы) ее. Посмотрю остальные позиции стека – по адресу `[esp+4]` располагается адрес в памяти где находится имя программы, по адресу `[esp+8]` храниться адрес первого аргумента, по адресу `[esp+12]` – второго и т.д (рис. 4.22).



```
paefremova@fedora:~/work/arch-pc/lab09 — gdb --args lab09-3 аргумент1 аргумент2 аргумент3
mc [pa... x paefre... x paefre... x paefre... x paefre... x paefre... x paefre... x
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab09-3...
(gdb) b start
Breakpoint 1 at 0x80490e8: file lab09-3.asm, line 5.
(gdb) run
Starting program: /home/paefremova/work/arch-pc/lab09/lab09-3 аргумент1 аргумент2 аргумент3

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:5
5      pop ecx ; Извлекаем из стека в 'ecx' количество
(gdb) x/x $esp
0xffffd000: 0x00000005
(gdb) x/s *(void**)(esp + 4)
0xffffd1cb: "/home/paefremova/work/arch-pc/lab09/lab09-3"
(gdb) x/s *(void**)(esp + 8)
0xffffd1f7: "аргумент1"
(gdb) x/s *(void**)(esp + 12)
0xffffd209: "аргумент2"
(gdb) x/s *(void**)(esp + 16)
0xffffd21a: "аргумент3"
(gdb) x/s *(void**)(esp + 20)
0xffffd22b: "2"
(gdb) x/s *(void**)(esp + 24)
0xffffd23c: "3"
(gdb) x/s *(void**)(esp + 28)
0xffffd24d: "4"
(gdb) x/s *(void**)(esp + 32)
0xffffd25e: "5"
(gdb) x/s *(void**)(esp + 36)
0xffffd26f: "6"
(gdb) x/s *(void**)(esp + 40)
0xffffd280: "7"
(gdb) x/s *(void**)(esp + 44)
0xffffd291: "8"
(gdb) x/s *(void**)(esp + 48)
0xffffd2a2: "9"
(gdb) x/s *(void**)(esp + 52)
0xffffd2b3: "10"
(gdb) x/s *(void**)(esp + 56)
0xffffd2c4: "11"
(gdb) x/s *(void**)(esp + 60)
0xffffd2d5: "12"
(gdb) x/s *(void**)(esp + 64)
0xffffd2e6: "13"
(gdb) x/s *(void**)(esp + 68)
0xffffd2f7: "14"
(gdb) x/s *(void**)(esp + 72)
0xffffd308: "15"
(gdb) x/s *(void**)(esp + 76)
0xffffd319: "16"
(gdb) x/s *(void**)(esp + 80)
0xffffd32a: "17"
(gdb) x/s *(void**)(esp + 84)
0xffffd33b: "18"
(gdb) x/s *(void**)(esp + 88)
0xffffd34c: "19"
(gdb) x/s *(void**)(esp + 92)
0xffffd35d: "20"
(gdb) x/s *(void**)(esp + 96)
0xffffd36e: "21"
(gdb) x/s *(void**)(esp + 100)
0xffffd37f: "22"
(gdb) x/s *(void**)(esp + 104)
0xffffd390: "23"
(gdb) x/s *(void**)(esp + 108)
0xffffd3a1: "24"
(gdb) x/s *(void**)(esp + 112)
0xffffd3b2: "25"
(gdb) x/s *(void**)(esp + 116)
0xffffd3c3: "26"
(gdb) x/s *(void**)(esp + 120)
0xffffd3d4: "27"
(gdb) x/s *(void**)(esp + 124)
0xffffd3e5: "28"
(gdb) x/s *(void**)(esp + 128)
0xffffd3f6: "29"
(gdb) x/s *(void**)(esp + 132)
0xffffd407: "30"
(gdb) x/s *(void**)(esp + 136)
0xffffd418: "31"
(gdb) x/s *(void**)(esp + 140)
0xffffd429: "32"
(gdb) x/s *(void**)(esp + 144)
0xffffd43a: "33"
(gdb) x/s *(void**)(esp + 148)
0xffffd44b: "34"
(gdb) x/s *(void**)(esp + 152)
0xffffd45c: "35"
(gdb) x/s *(void**)(esp + 156)
0xffffd46d: "36"
(gdb) x/s *(void**)(esp + 160)
0xffffd47e: "37"
(gdb) x/s *(void**)(esp + 164)
0xffffd48f: "38"
(gdb) x/s *(void**)(esp + 168)
0xffffd4a0: "39"
(gdb) x/s *(void**)(esp + 172)
0xffffd4b1: "40"
(gdb) x/s *(void**)(esp + 176)
0xffffd4c2: "41"
(gdb) x/s *(void**)(esp + 180)
0xffffd4d3: "42"
(gdb) x/s *(void**)(esp + 184)
0xffffd4e4: "43"
(gdb) x/s *(void**)(esp + 188)
0xffffd4f5: "44"
(gdb) x/s *(void**)(esp + 192)
0xffffd506: "45"
(gdb) x/s *(void**)(esp + 196)
0xffffd517: "46"
(gdb) x/s *(void**)(esp + 200)
0xffffd528: "47"
(gdb) x/s *(void**)(esp + 204)
0xffffd539: "48"
(gdb) x/s *(void**)(esp + 208)
0xffffd54a: "49"
(gdb) x/s *(void**)(esp + 212)
0xffffd55b: "50"
(gdb) x/s *(void**)(esp + 216)
0xffffd56c: "51"
(gdb) x/s *(void**)(esp + 220)
0xffffd57d: "52"
(gdb) x/s *(void**)(esp + 224)
0xffffd58e: "53"
(gdb) x/s *(void**)(esp + 228)
0xffffd59f: "54"
(gdb) x/s *(void**)(esp + 232)
0xffffd5b0: "55"
(gdb) x/s *(void**)(esp + 236)
0xffffd5c1: "56"
(gdb) x/s *(void**)(esp + 240)
0xffffd5d2: "57"
(gdb) x/s *(void**)(esp + 244)
0xffffd5e3: "58"
(gdb) x/s *(void**)(esp + 248)
0xffffd5f4: "59"
(gdb) x/s *(void**)(esp + 252)
0xffffd605: "60"
(gdb) x/s *(void**)(esp + 256)
0xffffd616: "61"
(gdb) x/s *(void**)(esp + 260)
0xffffd627: "62"
(gdb) x/s *(void**)(esp + 264)
0xffffd638: "63"
(gdb) x/s *(void**)(esp + 268)
0xffffd649: "64"
(gdb) x/s *(void**)(esp + 272)
0xffffd65a: "65"
(gdb) x/s *(void**)(esp + 276)
0xffffd66b: "66"
(gdb) x/s *(void**)(esp + 280)
0xffffd67c: "67"
(gdb) x/s *(void**)(esp + 284)
0xffffd68d: "68"
(gdb) x/s *(void**)(esp + 288)
0xffffd69e: "69"
(gdb) x/s *(void**)(esp + 292)
0xffffd6af: "70"
(gdb) x/s *(void**)(esp + 296)
0xffffd6b0: "71"
(gdb) x/s *(void**)(esp + 300)
0xffffd6c1: "72"
(gdb) x/s *(void**)(esp + 304)
0xffffd6d2: "73"
(gdb) x/s *(void**)(esp + 308)
0xffffd6e3: "74"
(gdb) x/s *(void**)(esp + 312)
0xffffd6f4: "75"
(gdb) x/s *(void**)(esp + 316)
0xffffd705: "76"
(gdb) x/s *(void**)(esp + 320)
0xffffd716: "77"
(gdb) x/s *(void**)(esp + 324)
0xffffd727: "78"
(gdb) x/s *(void**)(esp + 328)
0xffffd738: "79"
(gdb) x/s *(void**)(esp + 332)
0xffffd749: "80"
(gdb) x/s *(void**)(esp + 336)
0xffffd75a: "81"
(gdb) x/s *(void**)(esp + 340)
0xffffd76b: "82"
(gdb) x/s *(void**)(esp + 344)
0xffffd77c: "83"
(gdb) x/s *(void**)(esp + 348)
0xffffd78d: "84"
(gdb) x/s *(void**)(esp + 352)
0xffffd79e: "85"
(gdb) x/s *(void**)(esp + 356)
0xffffd7af: "86"
(gdb) x/s *(void**)(esp + 360)
0xffffd7b0: "87"
(gdb) x/s *(void**)(esp + 364)
0xffffd7c1: "88"
(gdb) x/s *(void**)(esp + 368)
0xffffd7d2: "89"
(gdb) x/s *(void**)(esp + 372)
0xffffd7e3: "90"
(gdb) x/s *(void**)(esp + 376)
0xffffd7f4: "91"
(gdb) x/s *(void**)(esp + 380)
0xffffd805: "92"
(gdb) x/s *(void**)(esp + 384)
0xffffd816: "93"
(gdb) x/s *(void**)(esp + 388)
0xffffd827: "94"
(gdb) x/s *(void**)(esp + 392)
0xffffd838: "95"
(gdb) x/s *(void**)(esp + 396)
0xffffd849: "96"
(gdb) x/s *(void**)(esp + 400)
0xffffd85a: "97"
(gdb) x/s *(void**)(esp + 404)
0xffffd86b: "98"
(gdb) x/s *(void**)(esp + 408)
0xffffd87c: "99"
(gdb) x/s *(void**)(esp + 412)
0xffffd88d: "100"
(gdb) x/s *(void**)(esp + 416)
0xffffd89e: "101"
(gdb) x/s *(void**)(esp + 420)
0xffffd8af: "102"
(gdb) x/s *(void**)(esp + 424)
0xffffd8b0: "103"
(gdb) x/s *(void**)(esp + 428)
0xffffd8c1: "104"
(gdb) x/s *(void**)(esp + 432)
0xffffd8d2: "105"
(gdb) x/s *(void**)(esp + 436)
0xffffd8e3: "106"
(gdb) x/s *(void**)(esp + 440)
0xffffd8f4: "107"
(gdb) x/s *(void**)(esp + 444)
0xffffd905: "108"
(gdb) x/s *(void**)(esp + 448)
0xffffd916: "109"
(gdb) x/s *(void**)(esp + 452)
0xffffd927: "110"
(gdb) x/s *(void**)(esp + 456)
0xffffd938: "111"
(gdb) x/s *(void**)(esp + 460)
0xffffd949: "112"
(gdb) x/s *(void**)(esp + 464)
0xffffd95a: "113"
(gdb) x/s *(void**)(esp + 468)
0xffffd96b: "114"
(gdb) x/s *(void**)(esp + 472)
0xffffd97c: "115"
(gdb) x/s *(void**)(esp + 476)
0xffffd98d: "116"
(gdb) x/s *(void**)(esp + 480)
0xffffd99e: "117"
(gdb) x/s *(void**)(esp + 484)
0xffffd9af: "118"
(gdb) x/s *(void**)(esp + 488)
0xffffd9b0: "119"
(gdb) x/s *(void**)(esp + 492)
0xffffd9c1: "120"
(gdb) x/s *(void**)(esp + 496)
0xffffd9d2: "121"
(gdb) x/s *(void**)(esp + 500)
0xffffd9e3: "122"
(gdb) x/s *(void**)(esp + 504)
0xffffd9f4: "123"
(gdb) x/s *(void**)(esp + 508)
0xffffd005: "124"
(gdb) x/s *(void**)(esp + 512)
0xffffd016: "125"
(gdb) x/s *(void**)(esp + 516)
0xffffd027: "126"
(gdb) x/s *(void**)(esp + 520)
0xffffd038: "127"
(gdb) x/s *(void**)(esp + 524)
0xffffd049: "128"
(gdb) x/s *(void**)(esp + 528)
0xffffd05a: "129"
(gdb) x/s *(void**)(esp + 532)
0xffffd06b: "130"
(gdb) x/s *(void**)(esp + 536)
0xffffd07c: "131"
(gdb) x/s *(void**)(esp + 540)
0xffffd08d: "132"
(gdb) x/s *(void**)(esp + 544)
0xffffd09e: "133"
(gdb) x/s *(void**)(esp + 548)
0xffffd0af: "134"
(gdb) x/s *(void**)(esp + 552)
0xffffd0b0: "135"
(gdb) x/s *(void**)(esp + 556)
0xffffd0c1: "136"
(gdb) x/s *(void**)(esp + 560)
0xffffd0d2: "137"
(gdb) x/s *(void**)(esp + 564)
0xffffd0e3: "138"
(gdb) x/s *(void**)(esp + 568)
0xffffd0f4: "139"
(gdb) x/s *(void**)(esp + 572)
0xffffd105: "140"
(gdb) x/s *(void**)(esp + 576)
0xffffd116: "141"
(gdb) x/s *(void**)(esp + 580)
0xffffd127: "142"
(gdb) x/s *(void**)(esp + 584)
0xffffd138: "143"
(gdb) x/s *(void**)(esp + 588)
0xffffd149: "144"
(gdb) x/s *(void**)(esp + 592)
0xffffd15a: "145"
(gdb) x/s *(void**)(esp + 596)
0xffffd16b: "146"
(gdb) x/s *(void**)(esp + 600)
0xffffd17c: "147"
(gdb) x/s *(void**)(esp + 604)
0xffffd18d: "148"
(gdb) x/s *(void**)(esp + 608)
0xffffd19e: "149"
(gdb) x/s *(void**)(esp + 612)
0xffffd1af: "150"
(gdb) x/s *(void**)(esp + 616)
0xffffd1b0: "151"
(gdb) x/s *(void**)(esp + 620)
0xffffd1c1: "152"
(gdb) x/s *(void**)(esp + 624)
0xffffd1d2: "153"
(gdb) x/s *(void**)(esp + 628)
0xffffd1e3: "154"
(gdb) x/s *(void**)(esp + 632)
0xffffd1f4: "155"
(gdb) x/s *(void**)(esp + 636)
0xffffd205: "156"
(gdb) x/s *(void**)(esp + 640)
0xffffd216: "157"
(gdb) x/s *(void**)(esp + 644)
0xffffd227: "158"
(gdb) x/s *(void**)(esp + 648)
0xffffd238: "159"
(gdb) x/s *(void**)(esp + 652)
0xffffd249: "160"
(gdb) x/s *(void**)(esp + 656)
0xffffd25a: "161"
(gdb) x/s *(void**)(esp + 660)
0xffffd26b: "162"
(gdb) x/s *(void**)(esp + 664)
0xffffd27c: "163"
(gdb) x/s *(void**)(esp + 668)
0xffffd28d: "164"
(gdb) x/s *(void**)(esp + 672)
0xffffd29e: "165"
(gdb) x/s *(void**)(esp + 676)
0xffffd2af: "166"
(gdb) x/s *(void**)(esp + 680)
0xffffd2b0: "167"
(gdb) x/s *(void**)(esp + 684)
0xffffd2c1: "168"
(gdb) x/s *(void**)(esp + 688)
0xffffd2d2: "169"
(gdb) x/s *(void**)(esp + 692)
0xffffd2e3: "170"
(gdb) x/s *(void**)(esp + 696)
0xffffd2f4: "171"
(gdb) x/s *(void**)(esp + 700)
0xffffd305: "172"
(gdb) x/s *(void**)(esp + 704)
0xffffd316: "173"
(gdb) x/s *(void**)(esp + 708)
0xffffd327: "174"
(gdb) x/s *(void**)(esp + 712)
0xffffd338: "175"
(gdb) x/s *(void**)(esp + 716)
0xffffd349: "176"
(gdb) x/s *(void**)(esp + 720)
0xffffd35a: "177"
(gdb) x/s *(void**)(esp + 724)
0xffffd36b: "178"
(gdb) x/s *(void**)(esp + 728)
0xffffd37c: "179"
(gdb) x/s *(void**)(esp + 732)
0xffffd38d: "180"
(gdb) x/s *(void**)(esp + 736)
0xffffd39e: "181"
(gdb) x/s *(void**)(esp + 740)
0xffffd3af: "182"
(gdb) x/s *(void**)(esp + 744)
0xffffd3b0: "183"
(gdb) x/s *(void**)(esp + 748)
0xffffd3c1: "184"
(gdb) x/s *(void**)(esp + 752)
0xffffd3d2: "185"
(gdb) x/s *(void**)(esp + 756)
0xffffd3e3: "186"
(gdb) x/s *(void**)(esp + 760)
0xffffd3f4: "187"
(gdb) x/s *(void**)(esp + 764)
0xffffd405: "188"
(gdb) x/s *(void**)(esp + 768)
0xffffd416: "189"
(gdb) x/s *(void**)(esp + 772)
0xffffd427: "190"
(gdb) x/s *(void**)(esp + 776)
0xffffd438: "191"
(gdb) x/s *(void**)(esp + 780)
0xffffd449: "192"
(gdb) x/s *(void**)(esp + 784)
0xffffd45a: "193"
(gdb) x/s *(void**)(esp + 788)
0xffffd46b: "194"
(gdb) x/s *(void**)(esp + 792)
0xffffd47c: "195"
(gdb) x/s *(void**)(esp + 796)
0xffffd48d: "196"
(gdb) x/s *(void**)(esp + 800)
0xffffd49e: "197"
(gdb) x/s *(void**)(esp + 804)
0xffffd4af: "198"
(gdb) x/s *(void**)(esp + 808)
0xffffd4b0: "199"
(gdb) x/s *(void**)(esp + 812)
0xffffd4c1: "200"
(gdb) x/s *(void**)(esp + 816)
0xffffd4d2: "201"
(gdb) x/s *(void**)(esp + 820)
0xffffd4e3: "202"
(gdb) x/s *(void**)(esp + 824)
0xffffd4f4: "203"
(gdb) x/s *(void**)(esp + 828)
0xffffd505: "204"
(gdb) x/s *(void**)(esp + 832)
0xffffd516: "205"
(gdb) x/s *(void**)(esp + 836)
0xffffd527: "206"
(gdb) x/s *(void**)(esp + 840)
0xffffd538: "207"
(gdb) x/s *(void**)(esp + 844)
0xffffd549: "208"
(gdb) x/s *(void**)(esp + 848)
0xffffd55a: "209"
(gdb) x/s *(void**)(esp + 852)
0xffffd56b: "210"
(gdb) x/s *(void**)(esp + 856)
0xffffd57c: "211"
(gdb) x/s *(void**)(esp + 860)
0xffffd58d: "212"
(gdb) x/s *(void**)(esp + 864)
0xffffd59e: "213"
(gdb) x/s *(void**)(esp + 868)
0xffffd5af: "214"
(gdb) x/s *(void**)(esp + 872)
0xffffd5b0: "215"
(gdb) x/s *(void**)(esp + 876)
0xffffd5c1: "216"
(gdb) x/s *(void**)(esp + 880)
0xffffd5d2: "217"
(gdb) x/s *(void**)(esp + 884)
0xffffd5e3: "218"
(gdb) x/s *(void**)(esp + 888)
0xffffd5f4: "219"
(gdb) x/s *(void**)(esp + 892)
0xffffd605: "220"
(gdb) x/s *(void**)(esp + 896)
0xffffd616: "221"
(gdb) x/s *(void**)(esp + 900)
0xffffd627: "222"
(gdb) x/s *(void**)(esp + 904)
0xffffd638: "223"
(gdb) x/s *(void**)(esp + 908)
0xffffd649: "224"
(gdb) x/s *(void**)(esp + 912)
0xffffd65a: "225"
(gdb) x/s *(void**)(esp + 916)
0xffffd66b: "226"
(gdb) x/s *(void**)(esp + 920)
0xffffd67c: "227"
(gdb) x/s *(void**)(esp + 924)
0xffffd68d: "228"
(gdb) x/s *(void**)(esp + 928)
0xffffd69e: "229"
(gdb) x/s *(void**)(esp + 932)
0xffffd6af: "230"
(gdb) x/s *(void**)(esp + 936)
0xffffd6b0: "231"
(gdb) x/s *(void**)(esp + 940)
0xffffd6c1: "232"
(gdb) x/s *(void**)(esp + 944)
0xffffd6d2: "233"
(gdb) x/s *(void**)(esp + 948)
0xffffd6e3: "234"
(gdb) x/s *(void**)(esp + 952)
0xffffd6f4: "235"
(gdb) x/s *(void**)(esp + 956)
0xffffd705: "236"
(gdb) x/s *(void**)(esp + 960)
0xffffd716: "237"
(gdb) x/s *(void**)(esp + 964)
0xffffd727: "238"
(gdb) x/s *(void**)(esp + 968)
0xffffd738: "239"
(gdb) x/s *(void**)(esp + 972)
0xffffd749: "240"
(gdb) x/s *(void**)(esp + 976)
0xffffd75a: "241"
(gdb) x/s *(void**)(esp + 980)
0xffffd76b: "242"
(gdb) x/s *(void**)(esp + 984)
0xffffd77c: "243"
(gdb) x/s *(void**)(esp + 988)
0xffffd78d: "244"
(gdb) x/s *(void**)(esp + 992)
0xffffd79e: "245"
(gdb) x/s *(void**)(esp + 996)
0xffffd7af: "246"
(gdb) x/s *(void**)(esp + 1000)
0xffffd7b0: "247"
(gdb) x/s *(void**)(esp + 1004)
0xffffd7c1: "248"
(gdb) x/s *(void**)(esp + 1008)
0xffffd7d2: "249"
(gdb) x/s *(void**)(esp + 1012)
0xffffd7e3: "250"
(gdb) x/s *(void**)(esp + 1016)
0xffffd7f4: "251"
(gdb) x/s *(void**)(esp + 1020)
0xffffd805: "252"
(gdb) x/s *(void**)(esp + 1024)
0xffffd816: "253"
(gdb) x/s *(void**)(esp + 1028)
0xffffd827: "254"
(gdb) x/s *(void**)(esp + 1032)
0xffffd838: "255"
(gdb) x/s *(void**)(esp + 1036)
0xffffd849: "256"
(gdb) x/s *(void**)(esp + 1040)
0xffffd85a: "257"
(gdb) x/s *(void**)(esp + 1044)
0xffffd86b: "258"
(gdb) x/s *(void**)(esp + 1048)
0xffffd87c: "259"
(gdb) x/s *(void**)(esp + 1052)
0xffffd88d: "260"
(gdb) x/s *(void**)(esp + 1056)
0xffffd89e: "261"
(gdb) x/s *(void**)(esp + 1060)
0xffffd8af: "262"
(gdb) x/s *(void**)(esp + 1064)
0xffffd8b0: "263"
(gdb) x/s *(void**)(esp + 1068)
0xffffd8c1: "264"
(gdb) x/s *(void**)(esp + 1072)
0xffffd8d2: "265"
(gdb) x/s *(void**)(esp + 1076)
0xffffd8e3: "266"
(gdb) x/s *(void**)(esp + 1080)
0xffffd8f4: "267"
(gdb) x/s *(void**)(esp + 1084)
0xffffd905: "268"
(gdb) x/s *(void**)(esp + 1088)
0xffffd916: "269"
(gdb) x/s *(void**)(esp + 1092)
0xffffd927: "270"
(gdb) x/s *(void**)(esp + 1096)
0xffffd938: "271"
(gdb) x/s *(void**)(esp + 1100)
0xffffd949: "272"
(gdb) x/s *(void**)(esp + 1104)
0xffffd95a: "273"
(gdb) x/s *(void**)(esp + 1108)
0xffffd96b: "274"
(gdb) x/s *(void**)(esp + 1112)
0xffffd97c: "275"
(gdb) x/s *(void**)(esp + 1116)
0xffffd98d: "276"
(gdb) x/s *(void**)(esp + 1120)
0xffffd99e: "277"
(gdb) x/s *(void**)(esp + 1124)
0xffffd9af: "278"
(gdb) x/s *(void**)(esp + 1128)
0xffffd9b0: "279"
(gdb) x/s *(void**)(esp + 1132)
0xffffd9c1: "280"
(gdb) x/s *(void**)(esp + 1136)
0xffffd9d2: "281"
(gdb) x/s *(void**)(esp + 1140)
0xffffd9e3: "282"
(gdb) x/s *(void**)(esp + 1144)
0xffffd9f4: "283"
(gdb) x/s *(void**)(esp + 1148)
0xffffd005: "284"
(gdb) x/s *(void**)(esp + 1152)
0xffffd016: "285"
(gdb) x/s *(void**)(esp + 1156)
0xffffd027: "286"
(gdb) x/s *(void**)(esp + 1160)
0xffffd038: "287"
(gdb) x/s *(void**)(esp + 1164)
0xffffd049: "288"
(gdb) x/s *(void**)(esp + 1168)
0xffffd05a: "289"
(gdb) x/s *(void**)(esp + 1172)
0xffffd06b: "290"
(gdb) x/s *(void**)(esp + 1176)
0xffffd07c: "291"
(gdb) x/s *(void**)(esp + 1180)
0xffffd08d: "292"
(gdb) x/s *(void**)(esp + 1184)
0xffffd09e: "293"
(gdb) x/s *(void**)(esp + 1188)
0xffffd0af: "294"
(gdb) x/s *(void**)(esp + 1192)
0xffffd0b0: "295"
(gdb) x/s *(void**)(esp + 1196)
0xffffd0c1: "296"
(gdb) x/s *(void**)(esp + 1200)
0xffffd0d2: "297"
(gdb) x/s *(void**)(esp + 1204)
0xffffd0e3: "298"
(gdb) x/s *(void**)(esp + 1208)
0xffffd0f4: "299"
(gdb) x/s *(void**)(esp + 1212)
0xffffd105: "300"
(gdb) x/s *(void**)(esp + 1216)
0xffffd116: "301"
(gdb) x/s *(void**)(esp + 1220)
0xffffd127: "302"
(gdb) x/s *(void**)(esp + 1224)
0xffffd138: "303"
(gdb) x/s *(void**)(esp + 1228)
0xffffd149: "304"
(gdb) x/s *(void**)(esp + 1232)
0xffffd15a: "305"
(gdb) x/s *(void**)(esp + 1236)
0xffffd16b: "306"
(gdb) x/s *(void**)(esp + 1240)
0xffffd17c: "307"
(gdb) x/s *(void**)(esp + 1244)
0xffffd18d: "308"
(gdb) x/s *(void**)(esp + 1248)
0xffffd19e: "309"
(gdb) x/s *(void**)(esp + 1252)
0xffffd1af: "310"
(gdb) x/s *(void**)(esp + 1256)
0xffffd1b0: "311"
(gdb) x/s *(void**)(esp + 1260)
0xffffd1c1: "312"
(gdb) x/s *(void**)(esp + 1264)
0xffffd1d2: "313"
(gdb) x/s *(void**)(esp + 1268)
0xffffd1e3: "314"
(gdb) x/s *(void**)(esp + 1272)
0xffffd1f4: "315"
(gdb) x/s *(void**)(esp + 1276)
0xffffd205: "316"
(gdb) x/s *(void**)(esp + 1280)
0xffffd216: "317"
(gdb) x/s *(void**)(esp + 1284)
0xffffd227: "318"
(gdb) x/s *(void**)(esp + 1288)
0xffffd238: "319"
(gdb) x/s *(void**)(esp + 1292)
0xffffd249: "320"
(gdb) x/s *(void**)(esp + 1296)
0xffffd25a: "321"
(gdb) x/s *(void**)(esp + 1300)
0xffffd26b: "322"
(gdb) x/s *(void**)(esp + 1304)
0xffffd27c: "323"
(gdb) x/s *(void**)(esp + 1308)
0xffffd28d: "324"
(gdb) x/s *(void**)(esp + 1312)
0xffffd29e: "325"
(gdb) x/s *(void**)(esp + 1316)
0xffffd2af: "326"
(gdb) x/s *(void**)(esp + 1320)
0xffffd2b0: "327"
(gdb) x/s *(void**)(esp + 1324)
0xffffd2c1: "328"
(gdb) x/s *(void**)(esp + 1328)
0xffffd2d2: "329"
(gdb) x/s *(void**)(esp + 1332)
0xffffd2e3: "330"
(gdb) x/s *(void**)(esp + 1336)
0xffffd2f4: "331"
(gdb) x/s *(void**)(esp + 1340)
0xffffd305: "332"
(gdb) x/s *(void**)(esp + 1344)
0xffffd316: "333"
(gdb) x/s *(void**)(esp + 1348)
0xffffd327: "334"
(gdb) x/s *(void**)(esp + 1352)
0xffffd338: "335"
(gdb) x/s *(void**)(esp + 1356)
0xffffd349: "336"
(gdb) x/s *(void**)(esp + 1360)
0xffffd35a: "337"
(gdb) x/s *(void**)(esp + 1364)
0xffffd36b: "338"
(gdb) x/s *(void**)(esp + 1368)
0xffffd37c: "339"
(gdb) x/s *(void**)(esp + 1372)
0xffffd38d: "340"
(gdb) x/s *(void**)(esp + 1376)
0xffffd39e: "341"
(gdb) x/s *(void**)(esp + 1380)
0xffffd3af: "342"
(gdb) x/s *(void**)(esp + 1384)
0xffffd3b0: "343"
(gdb) x/s *(void**)(esp + 1388)
0xffffd3c1: "344"
(gdb) x/s *(void**)(esp + 1392)
0xffffd3d2: "345"
(gdb) x/s *(void**)(esp + 1396)
0xffffd3e3: "346"
(gdb) x/s *(void**)(esp + 1400)
0xffffd3f4: "347"
(gdb) x/s *(void**)(esp + 1404)
0xffffd405: "348"
(gdb) x/s *(void**)(esp + 1408)
0xffffd416: "349"
(gdb) x/s *(void**)(esp + 1412)
0xffffd427: "350"
(gdb) x/s *(void**)(esp + 1416)
0xffffd438: "351"
(gdb) x/s *(void**)(esp + 1420)
0xffffd449: "352"
(gdb) x/s *(void**)(esp + 1424)
0xffffd45a: "353"
(gdb) x/s *(void**)(esp + 1428)
0xffffd46b: "354"
(gdb) x/s *(void**)(esp + 1432)
0xffffd47c: "355"
(gdb) x/s *(void**)(esp + 1436)
0xffffd48d: "356"
(gdb) x/s *(void**)(esp + 1440)
0xffffd49e: "357"
(gdb) x/s *(void**)(esp + 1444)
0xffffd4af: "358"
(gdb) x/s *(void**)(esp + 1448)
0xffffd4b0: "359"
(gdb) x/s *(void**)(esp + 1452)
0xffffd4c1: "360"
(gdb) x/s *(void**)(esp + 1456)
0xffffd4d2: "361"
(gdb) x/s *(void**)(esp + 1460)
0xffffd4e3: "362"
(gdb) x/s *(void**)(esp + 1464)
0xffffd4f4: "363"
(gdb) x/s *(void**)(esp + 1468)
0xffffd505: "364"
(gdb) x/s *(void**)(esp + 1472)
0xffffd516: "365"
(gdb) x/s *(void**)(esp + 1476)
0xffffd527: "366"
(gdb) x/s *(void**)(esp + 1480)
0xffffd538: "367"
(gdb) x/s *(void**)(esp + 1484)
0xffffd549: "368"
(gdb) x/s *(void**)(esp + 1488)
0xffffd55a: "369"
(gdb) x/s *(void**)(esp + 1492)
0xffffd56b: "370"
(gdb) x/s *(void**)(esp + 1496)
0xffffd57c: "371"
(gdb) x/s *(void**)(esp + 1500)
0xffffd58d: "372"
(gdb) x/s *(void**)(esp + 1504)
0xffffd59e: "373"
(gdb) x/s *(void**)(esp + 1508)
0xffffd5af: "374"
(gdb) x/s *(void**)(esp + 1512)
0xffffd5b0: "375"
(gdb) x/s *(void**)(esp + 1516)
0xffffd5c1: "376"
(gdb) x/s *(void**)(esp + 1520)
0xffffd5d2: "377"
(gdb) x/s *(void**)(esp + 1524)
0xffffd5e3: "378"
(gdb) x/s *(void**)(esp + 1528)
0xffffd5f4: "379"
(gdb) x/s *(void**)(esp + 1532)
0xffffd605: "380"
(gdb) x/s *(void**)(esp + 1536)
0xffffd616: "381"
(gdb) x/s *(void**)(esp + 1540)
0xffffd627: "382"
(gdb) x/s *(void**)(esp + 1544)
0xffffd638: "383"
(gdb) x/s *(void**)(esp + 1548)
0xffffd649: "384"
(gdb) x/s *(void**)(esp + 1552)
0xffffd65a: "385"
(gdb) x/s *(void**)(esp + 1556)
0xffffd66b: "386"
(gdb) x/s *(void**)(esp + 1560)
0xffffd67c: "387"
(gdb) x/s *(void**)(esp + 1564)
0xffffd68d: "388"
(gdb) x/s *(void**)(esp + 1568)
0xffffd69e: "389"
(gdb) x/s *(void**)(esp + 1572)
0xffffd6af: "390"
(gdb) x/s *(void**)(esp + 1576)
0xffffd6b0: "391"
(gdb) x/s *(void**)(esp + 1580)
0xffffd6c
```

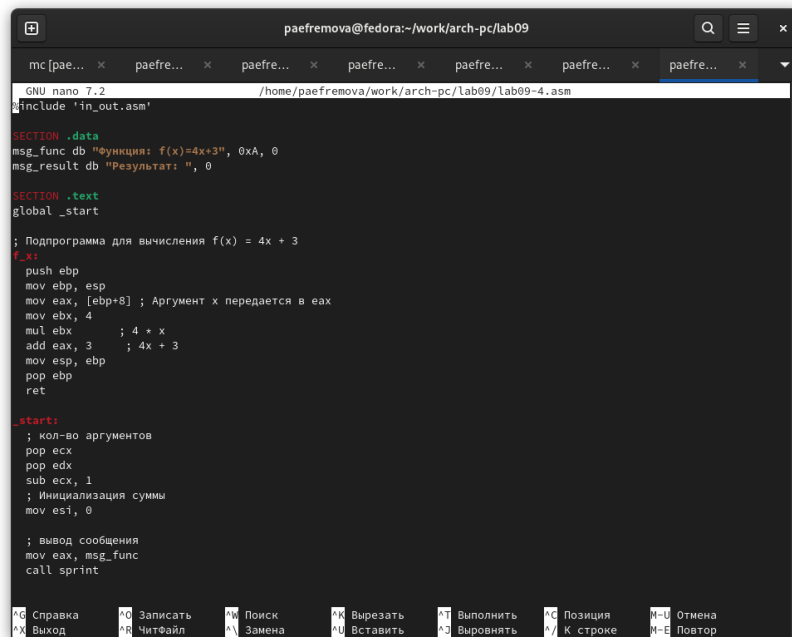


Рис. 4.23: задание 1 ср

Вот код программы:

```
%include 'in_out.asm'
```

```
SECTION .data
```

```
msg_func db "Функция: f(x)=4x+3", 0xA, 0
```

```
msg_result db "Результат: ", 0
```

```
SECTION .text
```

```
global _start
```

```
; Подпрограмма для вычисления  $f(x) = 4x + 3$ 
```

```
f_x:
```

```
    push ebp
```

```
    mov ebp, esp
```

```
    mov eax, [ebp+8] ; Аргумент x передается в eax
```

```

mov ebx, 4
mul ebx      ;  $4 * x$ 
add eax, 3    ;  $4x + 3$ 
mov esp, ebp
pop ebp
ret

```

_start:

```

; кол-во аргументов
pop ecx
pop edx
sub ecx, 1
; Инициализация суммы
mov esi, 0

```

```

; вывод сообщения
mov eax, msg_func
call sprint

```

next:

```

cmp ecx, 0
jz _end
; Извлекаю аргумент со стека
pop eax
; Преобразую аргумент в число
push eax
call atoi
add esp, 4

```


; Вычисляю $f(x)$ используя подпрограмму

push eax

call f_x

add esp, 4

add esi, eax ; Добавляем результат функции к сумме

loop next

_end:

mov eax, msg_result

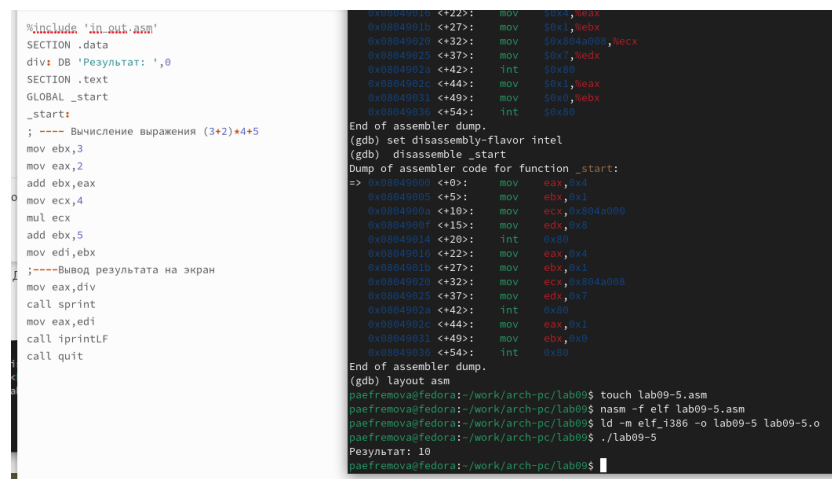
call sprint

mov eax, esi

call iprintLF

call quit

2. В листинге 9.3 приведена программа вычисления выражения $(3+2)*4+5$. При запуске данная программа дает неверный результат. Проверяю это. Для этого создаю файл, куда ввожу программу из листинга 9.3. Запускаю и проверяю. (рис. 4.24).



```
%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 ebx,eax
mov ecx,4
mul ecx
add ebx,5
mov edi,ebx
; ---- Вывод результата на экран
mov eax,div
call sprint
mov eax,edi
call iprintLF
call quit

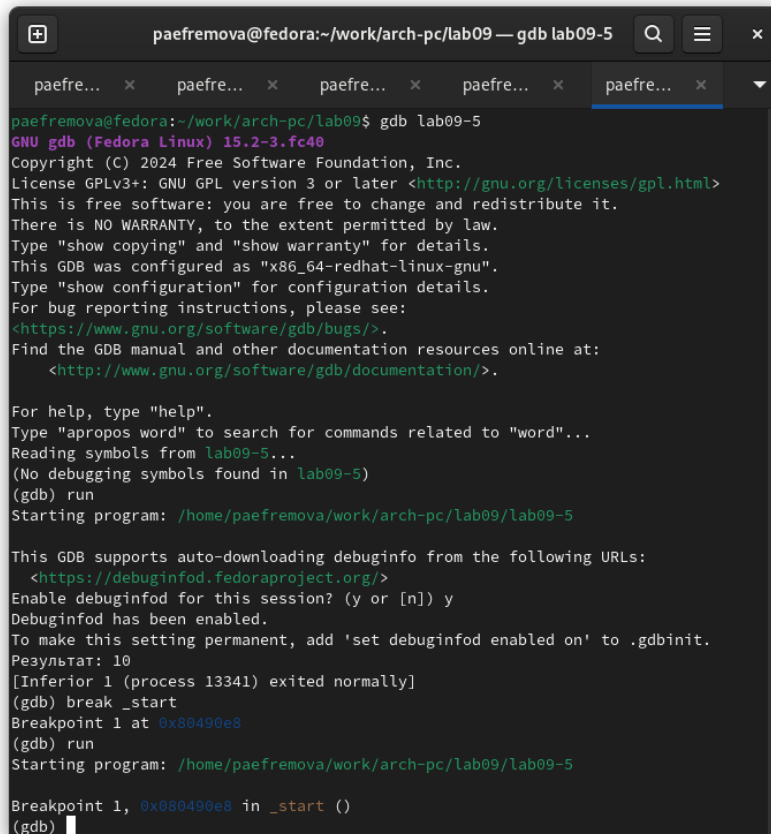
0x00040010 <+22>: mov     eax,ebx
0x0004001b <+27>: mov     ebx,ecx
0x00040020 <+32>: mov     ecx,edi
0x00040025 <+37>: mov     edi,ebx
0x0004002c <+42>: int     $x86
0x0004002c <+44>: mov     eax,ebx
0x00040031 <+49>: mov     ebx,ecx
0x00040036 <+54>: int     $x86
End of assembler dump.
(gdb) set disassembly-flavor intel
(gdb) disassemble _start
Dump of assembler code for function _start:
=> 0x00040010 <+0>: mov     eax,edx
0x00040015 <+5>: mov     ebx,edx
0x0004001a <+10>: mov     ecx,edx
0x0004001f <+15>: mov     edx,edx
0x00040024 <+20>: int     $x86
0x00040029 <+25>: mov     eax,edx
0x0004002c <+30>: mov     ebx,ecx
0x0004002f <+35>: mov     ecx,edx
0x00040032 <+40>: mov     edi,edx
0x00040035 <+45>: mov     eax,edi
0x00040038 <+50>: mov     ebx,edx
0x0004003b <+55>: int     $x86
End of assembler dump.
(gdb) layout asm
paefremova@fedora:~/work/arch-pc/lab09$ touch lab09-5.asm
paefremova@fedora:~/work/arch-pc/lab09$ nasm -f elf lab09-5.asm
paefremova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-5 lab09-5.o
paefremova@fedora:~/work/arch-pc/lab09$ ./lab09-5
Результат: 10
paefremova@fedora:~/work/arch-pc/lab09$
```

Рис. 4.24: Запуск программы с ошибкой

Код программы с ошибкой:

```
%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 ebx,eax
mov ecx,4
mul ecx
add ebx,5
mov edi,ebx
;----Вывод результата на экран
mov eax,div
call sprint
mov eax,edi
call iprintLF
call quit
```

С помощью отладчика GDB, анализируя изменения значений регистров, определяю ошибку и исправляю ее.. 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. (рис. 4.25).



The screenshot shows a terminal window titled "paefremova@fedora:~/work/arch-pc/lab09 — gdb lab09-5". The user has entered the command "gdb lab09-5". The terminal displays the GDB startup sequence, including the GNU GDB version (15.2-3.fc40), copyright information, and license details. The user then enters "run", and the program starts. The terminal shows the program's output: "Результат: 10". The user then enters "break _start", and a breakpoint is set at address 0x080490e8. The user enters "run" again, and the program starts. The terminal shows the breakpoint hit at address 0x080490e8 in the function _start.

```
paefremova@fedora:~/work/arch-pc/lab09$ gdb lab09-5
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 lab09-5...
(No debugging symbols found in lab09-5)
(gdb) run
Starting program: /home/paefremova/work/arch-pc/lab09/lab09-5

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.
Результат: 10
[Inferior 1 (process 13341) exited normally]
(gdb) break _start
Breakpoint 1 at 0x080490e8
(gdb) run
Starting program: /home/paefremova/work/arch-pc/lab09/lab09-5

Breakpoint 1, 0x080490e8 in _start ()
(gdb)
```

Рис. 4.25: Нахождение ошибки

Теперь исправлю это недоразумение Вот текст исправленной программы:

```
%include 'in_out.asm'

SECTION .data
div: DB 'Результат: ', 0

SECTION .text
GLOBAL _start
_start:

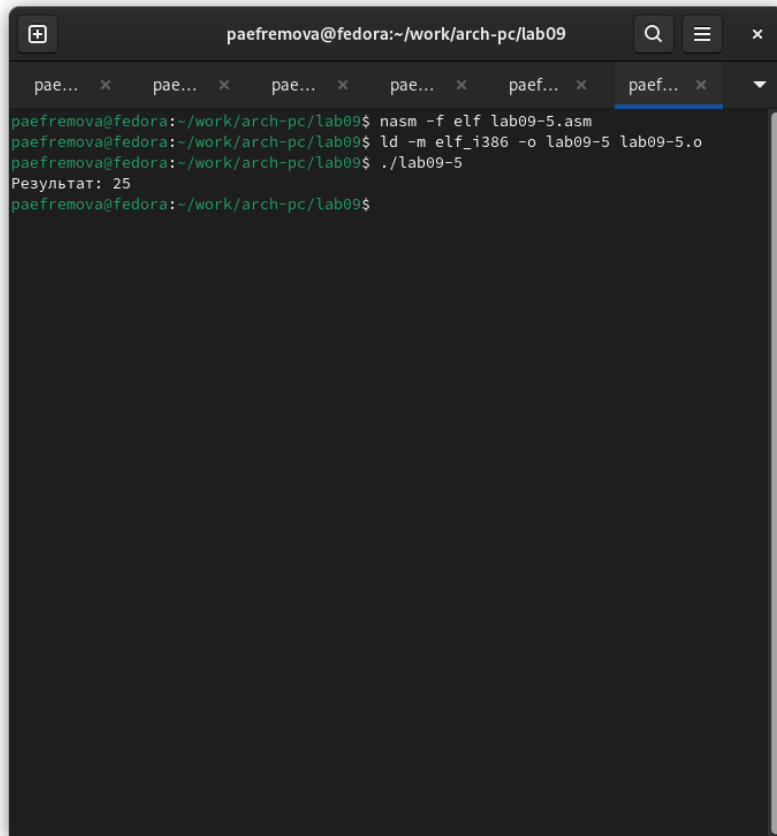
mov ebx, 3
```

```
mov eax, 2
add ebx, eax
mov eax, ebx
mov ecx, 4
mul ecx
add eax, 5
mov edi, eax
```

```
mov eax, div
call sprint
mov eax, edi
call iprintLF
```

```
call quit
```

Теперь проверяю изменения! (рис. 4.26).



A terminal window titled 'paefremova@fedora:~/work/arch-pc/lab09'. The window contains the following commands and output:

```
paefremova@fedora:~/work/arch-pc/lab09$ nasm -f elf lab09-5.asm
paefremova@fedora:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab09-5 lab09-5.o
paefremova@fedora:~/work/arch-pc/lab09$ ./lab09-5
Результат: 25
paefremova@fedora:~/work/arch-pc/lab09$
```

Рис. 4.26: Проверка

5 Выводы

В результате выполнения данной лабораторной работы я приобрела навыки написания программ с использованием подпрограмм, а так же познакомилась с методами отладки при помощи GDB и его основными возможностями.

Список литературы

1. Курс на ТУИС
2. Лабораторная работа №9