

Отчет по лабораторной работе №9

Дисциплина: Архитектура компьютера

Юсуфов Джабар Артикович

Содержание

1	Цель работы	1
2	Задание	1
3	Выполнение лабораторной работы	1
3.1	Реализация подпрограмм в NASM	1
3.1.1	Отладка программ с помощью GDB	3
3.1.2	Добавление точек останова	7
3.1.3	Работа с данными программы в GDB	9
3.1.4	Обработка аргументов командной строки в GDB	13
3.2	Задание для самостоятельной работы	15
4	Выводы	18

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. Задание для самостоятельной работы.

3 Выполнение лабораторной работы

3.1 Реализация подпрограмм в NASM

Создаю каталог для выполнения лабораторной работы №9.

```
neroun@fedora:~$ mkdir ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/lab09
neroun@fedora:~$ cd ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc/lab09
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ touch lab9-1.asm
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$
```

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

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ nasm -f elf lab9-1.asm
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ld -m elf_i386 -o lab9-1 lab9-1.o
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ./lab9-1
Введите x: 10
2x+7=27
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$
```

Изменяю текст программы, добавив в нее подпрограмму, теперь она вычисляет значение функции для выражения.

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ nasm -f elf lab9-1.asm
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ld -m elf_i386 -o lab9-1 lab9-1.o
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ./lab9-1
Введите x: 10
2(3x-1)+7=65
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$
```

Код 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
mov eax, result
call sprint
mov eax, [res]
call iprintLF
call quit
_calcul:
push eax
call _subcalcul
mov ebx, 2
mul ebx
add eax, 7
mov [res], eax
```

```
pop eax
ret
_subcalcul:
mov ebx, 3
mul ebx
sub eax, 1
ret
```

3.1.1 Отладка программ с помощью GDB

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

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ touch lab9-2.asm
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ nasm -f elf -g -l
lab9-2.lst lab9-2.asm
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ld -m elf_i386 -o
lab9-2 lab9-2.o
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ gdb lab9-2
GNU gdb (Fedora Linux) 14.2-1.fc40
Copyright (C) 2023 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 lab9-2...
(gdb) █
```

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

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ gdb lab9-2
GNU gdb (Fedora Linux) 14.2-1.fc40
Copyright (C) 2023 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 lab9-2...
(gdb) run
Starting program: /home/neroun/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09/lab9-2

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 255598) exited normally]
(gdb)
```

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

```
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-2...
(gdb) run
Starting program: /home/neroun/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09/lab9-2

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 255598) exited normally]
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab9-2.asm, line 9.
(gdb) run
Starting program: /home/neroun/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09/lab9-2

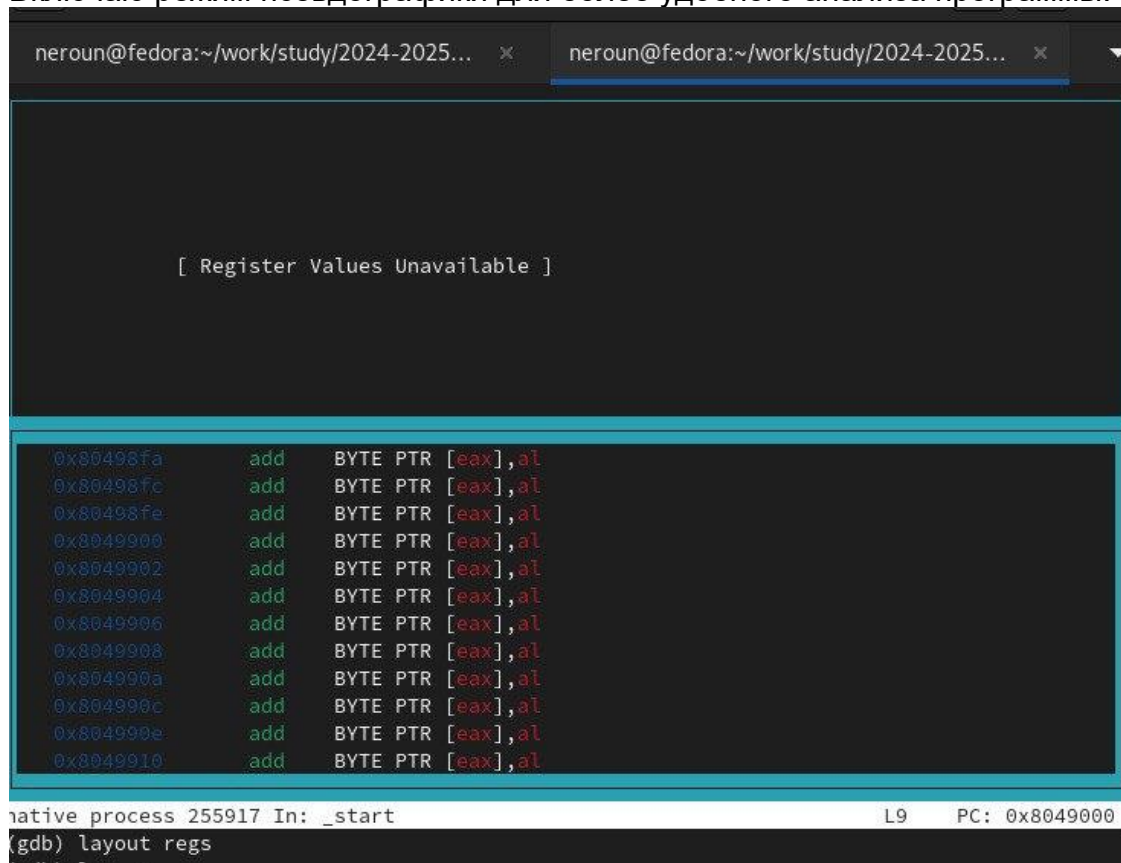
Breakpoint 1, _start () at lab9-2.asm:9
9      mov eax, 4
(gdb) |
```

Далее смотрю дисассимилированный код программы, перевожу на команды с синтаксисом Intel. Различия между синтаксисом АТТ и Intel заключаются в порядке

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

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc...
neroun@fedora:~/work/study/2024-2025... x neroun@fedora:~/work/study/2024-2025... x
Starting program: /home/neroun/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09/lab9-2
Breakpoint 1, _start () at lab9-2.asm:9
9      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) set disassemble-flavor intel
No symbol "disassemble" in current context.
(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) |
```


Включаю режим псевдографики для более удобного анализа программы.



The screenshot shows a GDB terminal window with two tabs. The active tab displays assembly code and register values. The assembly code is as follows:

Address	Disassembly
0x80498fa	add BYTE PTR [eax],al
0x80498fc	add BYTE PTR [eax],al
0x80498fe	add BYTE PTR [eax],al
0x8049900	add BYTE PTR [eax],al
0x8049902	add BYTE PTR [eax],al
0x8049904	add BYTE PTR [eax],al
0x8049906	add BYTE PTR [eax],al
0x8049908	add BYTE PTR [eax],al
0x804990a	add BYTE PTR [eax],al
0x804990c	add BYTE PTR [eax],al
0x804990e	add BYTE PTR [eax],al
0x8049910	add BYTE PTR [eax],al

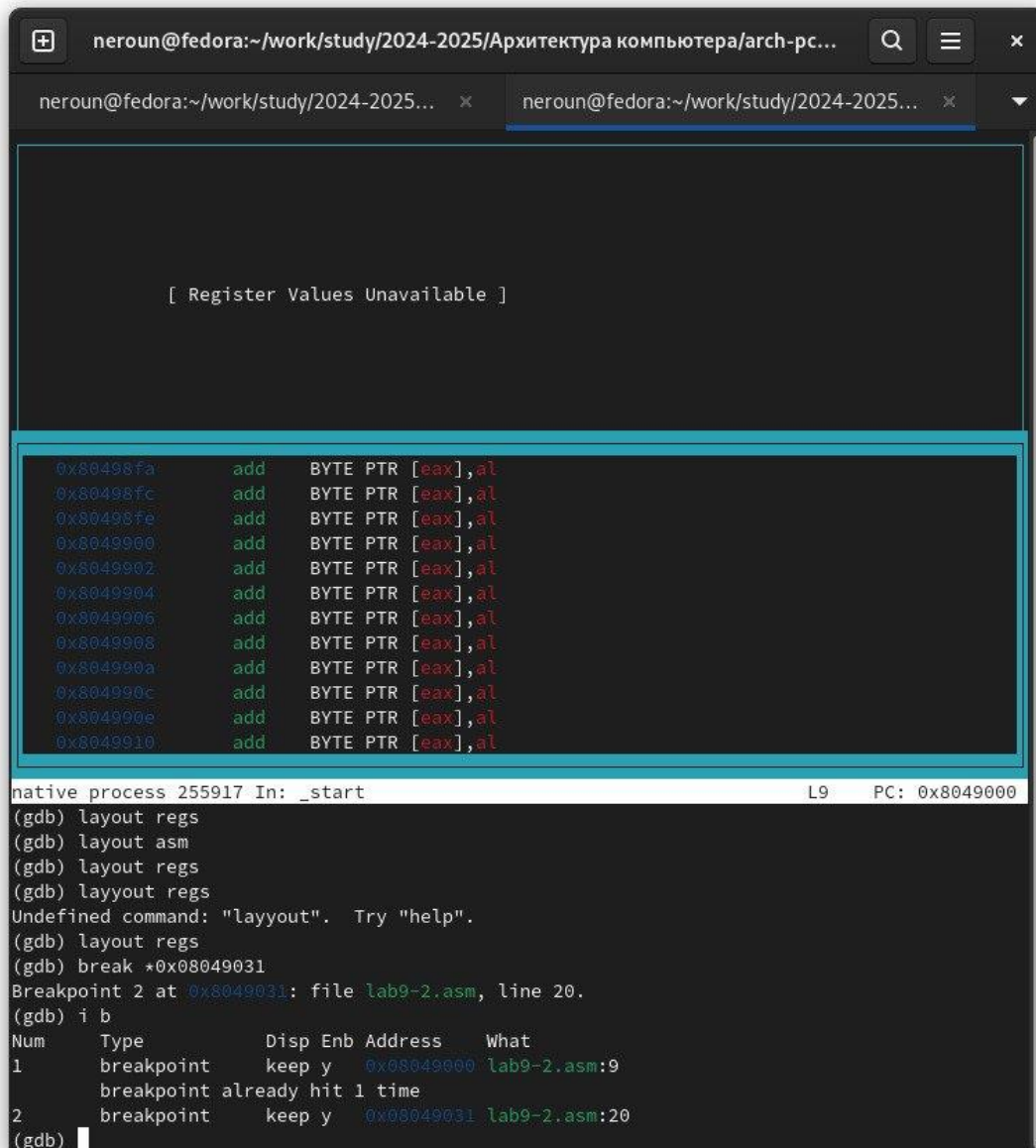
Below the assembly code, the register values are displayed:

```
(gdb) layout regs
[ Register Values Unavailable ]
```

The status bar at the bottom of the window shows: native process 255917 In: _start L9 PC: 0x8049000.

3.1.2 Добавление точек останова

Проверяю в режиме псевдографики, что брейкпоинт сохранился.



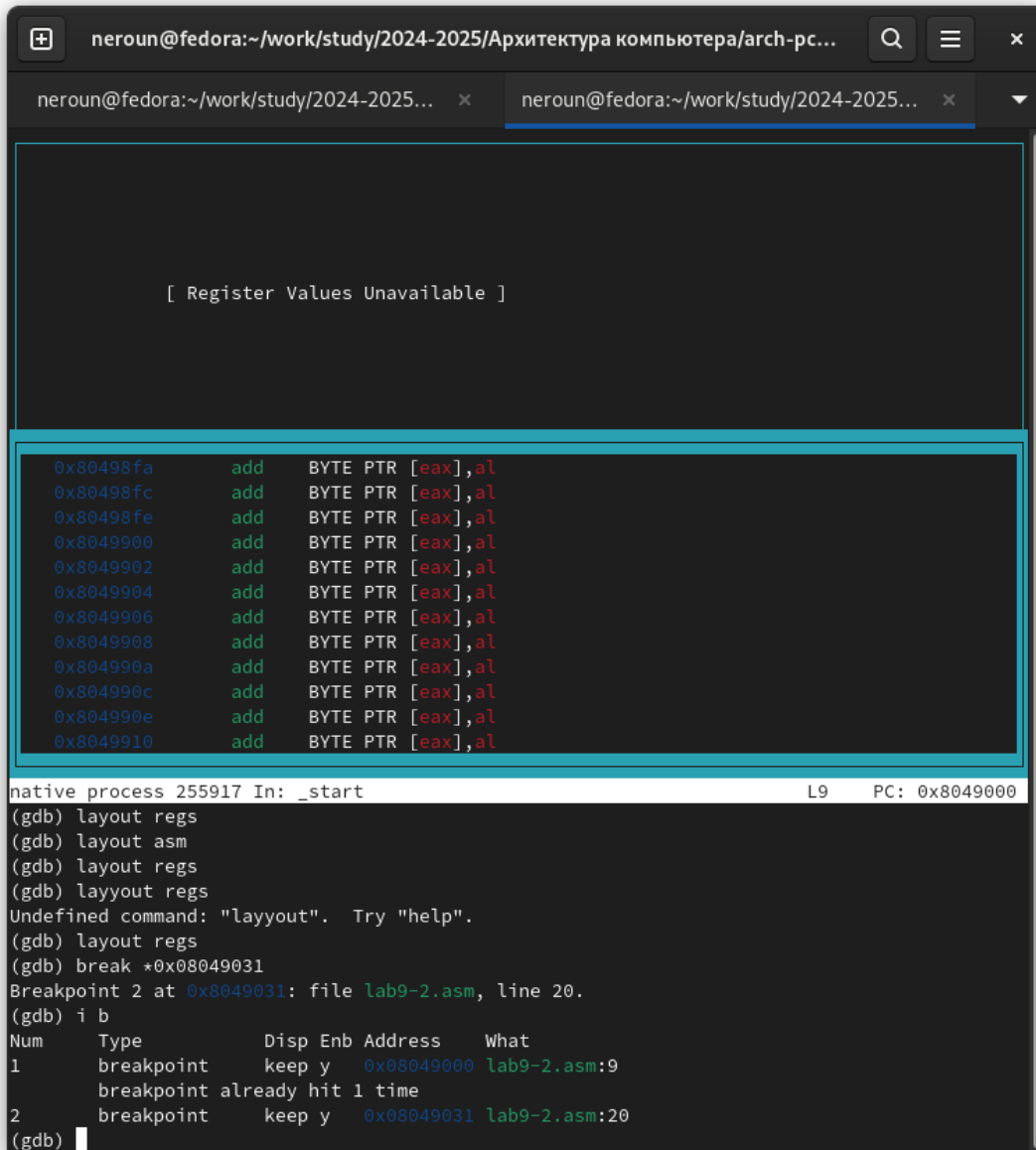
The screenshot shows a GDB terminal window with the following content:

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc...
neroun@fedora:~/work/study/2024-2025... x neroun@fedora:~/work/study/2024-2025... x
[ Register Values Unavailable ]

0x80498fa    add    BYTE PTR [eax],al
0x80498fc    add    BYTE PTR [eax],al
0x80498fe    add    BYTE PTR [eax],al
0x8049900    add    BYTE PTR [eax],al
0x8049902    add    BYTE PTR [eax],al
0x8049904    add    BYTE PTR [eax],al
0x8049906    add    BYTE PTR [eax],al
0x8049908    add    BYTE PTR [eax],al
0x804990a    add    BYTE PTR [eax],al
0x804990c    add    BYTE PTR [eax],al
0x804990e    add    BYTE PTR [eax],al
0x8049910    add    BYTE PTR [eax],al

native process 255917 In: _start L9 PC: 0x8049000
(gdb) layout regs
(gdb) layout asm
(gdb) layout regs
(gdb) layout regs
Undefined command: "layment". Try "help".
(gdb) layout regs
(gdb) break *0x08049031
Breakpoint 2 at 0x08049031: file lab9-2.asm, line 20.
(gdb) i b
Num    Type           Disp Enb Address      What
1      breakpoint      keep y   0x08049000 lab9-2.asm:9
       breakpoint already hit 1 time
2      breakpoint      keep y   0x08049031 lab9-2.asm:20
(gdb)
```

Устанавливаю еще одну точку останова по адресу инструкции.



The screenshot shows a terminal window with a dark background. At the top, there's a window title bar with the text "neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc...". Below the title bar, there are two tabs, both labeled "neroun@fedora:~/work/study/2024-2025...". The main content area is divided into two sections. The top section, which is highlighted with a light blue border, displays the text "[Register Values Unavailable]". The bottom section shows a list of assembly instructions, each preceded by an address and the instruction itself. The instructions are: "add BYTE PTR [eax],al" repeated 11 times, with addresses ranging from 0x80498fa to 0x8049910. Below this, there's a status bar that reads "native process 255917 In: _start L9 PC: 0x8049000". The bottom section of the terminal shows the GDB session, including commands like "layout regs", "layout asm", "break *0x08049031", and "i b". The output of the "i b" command shows two breakpoints: one at 0x08049000 (lab9-2.asm:9) and another at 0x08049031 (lab9-2.asm:20).

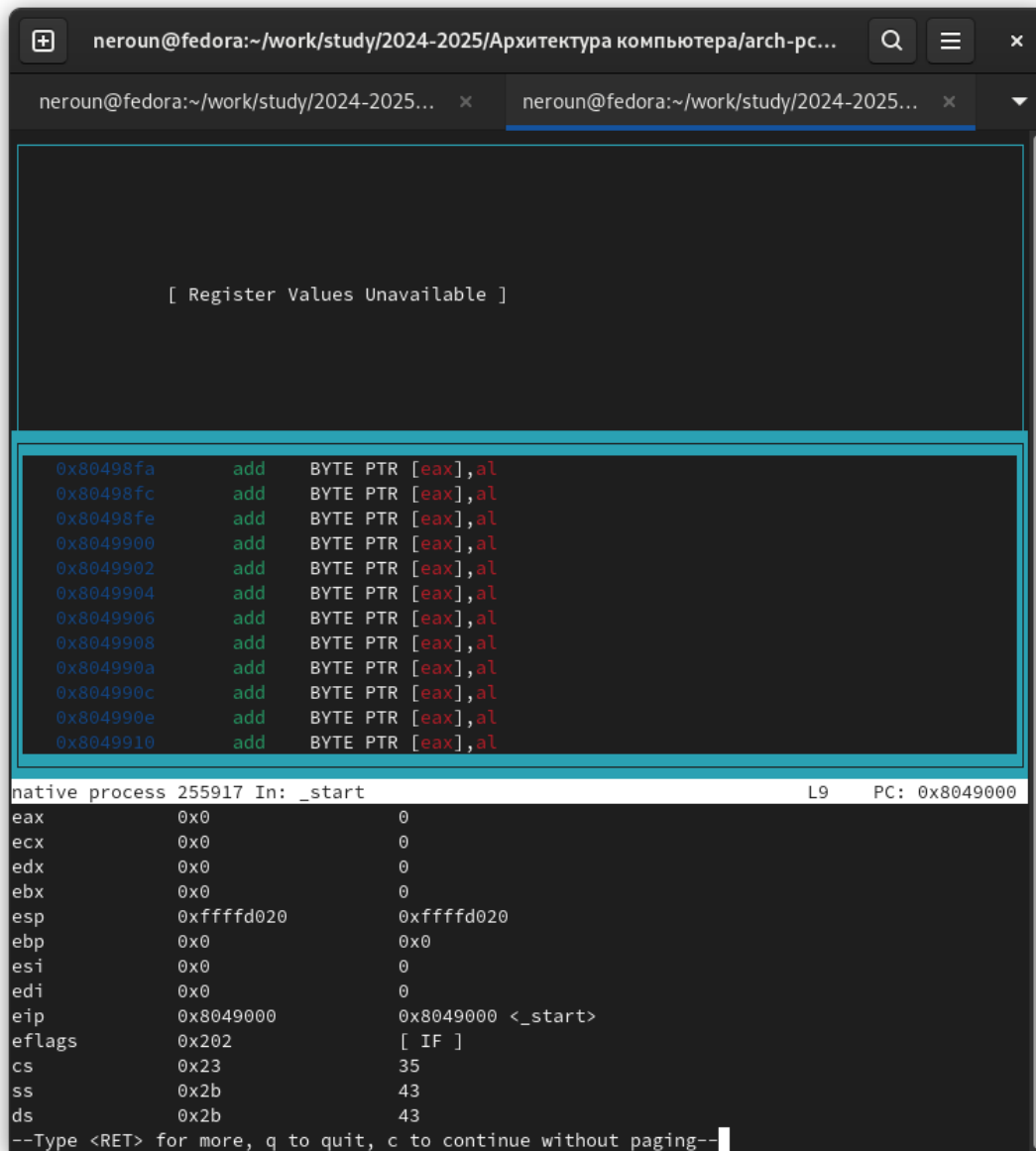
```
[ Register Values Unavailable ]
```

0x80498fa	add	BYTE PTR [eax],al
0x80498fc	add	BYTE PTR [eax],al
0x80498fe	add	BYTE PTR [eax],al
0x8049900	add	BYTE PTR [eax],al
0x8049902	add	BYTE PTR [eax],al
0x8049904	add	BYTE PTR [eax],al
0x8049906	add	BYTE PTR [eax],al
0x8049908	add	BYTE PTR [eax],al
0x804990a	add	BYTE PTR [eax],al
0x804990c	add	BYTE PTR [eax],al
0x804990e	add	BYTE PTR [eax],al
0x8049910	add	BYTE PTR [eax],al

```
native process 255917 In: _start L9 PC: 0x8049000
(gdb) layout regs
(gdb) layout asm
(gdb) layout regs
(gdb) layyout regs
Undefined command: "layyout". Try "help".
(gdb) layout regs
(gdb) break *0x08049031
Breakpoint 2 at 0x08049031: file lab9-2.asm, line 20.
(gdb) i b
Num    Type           Disp Enb Address      What
1      breakpoint     keep y   0x08049000 lab9-2.asm:9
       breakpoint already hit 1 time
2      breakpoint     keep y   0x08049031 lab9-2.asm:20
(gdb)
```


3.1.3 Работа с данными программы в GDB

Просматриваю содержимое регистров командой `info registers`.



The screenshot shows a GDB terminal window with two tabs. The active tab displays the following content:

```
[ Register Values Unavailable ]
```

```
0x80498fa    add    BYTE PTR [eax],al
0x80498fc    add    BYTE PTR [eax],al
0x80498fe    add    BYTE PTR [eax],al
0x8049900    add    BYTE PTR [eax],al
0x8049902    add    BYTE PTR [eax],al
0x8049904    add    BYTE PTR [eax],al
0x8049906    add    BYTE PTR [eax],al
0x8049908    add    BYTE PTR [eax],al
0x804990a    add    BYTE PTR [eax],al
0x804990c    add    BYTE PTR [eax],al
0x804990e    add    BYTE PTR [eax],al
0x8049910    add    BYTE PTR [eax],al
```

```
native process 255917 In: _start          L9    PC: 0x8049000
eax          0x0          0
ecx          0x0          0
edx          0x0          0
ebx          0x0          0
esp          0xffffd020    0xffffd020
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
--Type <RET> for more, q to quit, c to continue without paging--
```

Смотрю содержимое переменных по имени и по адресу.

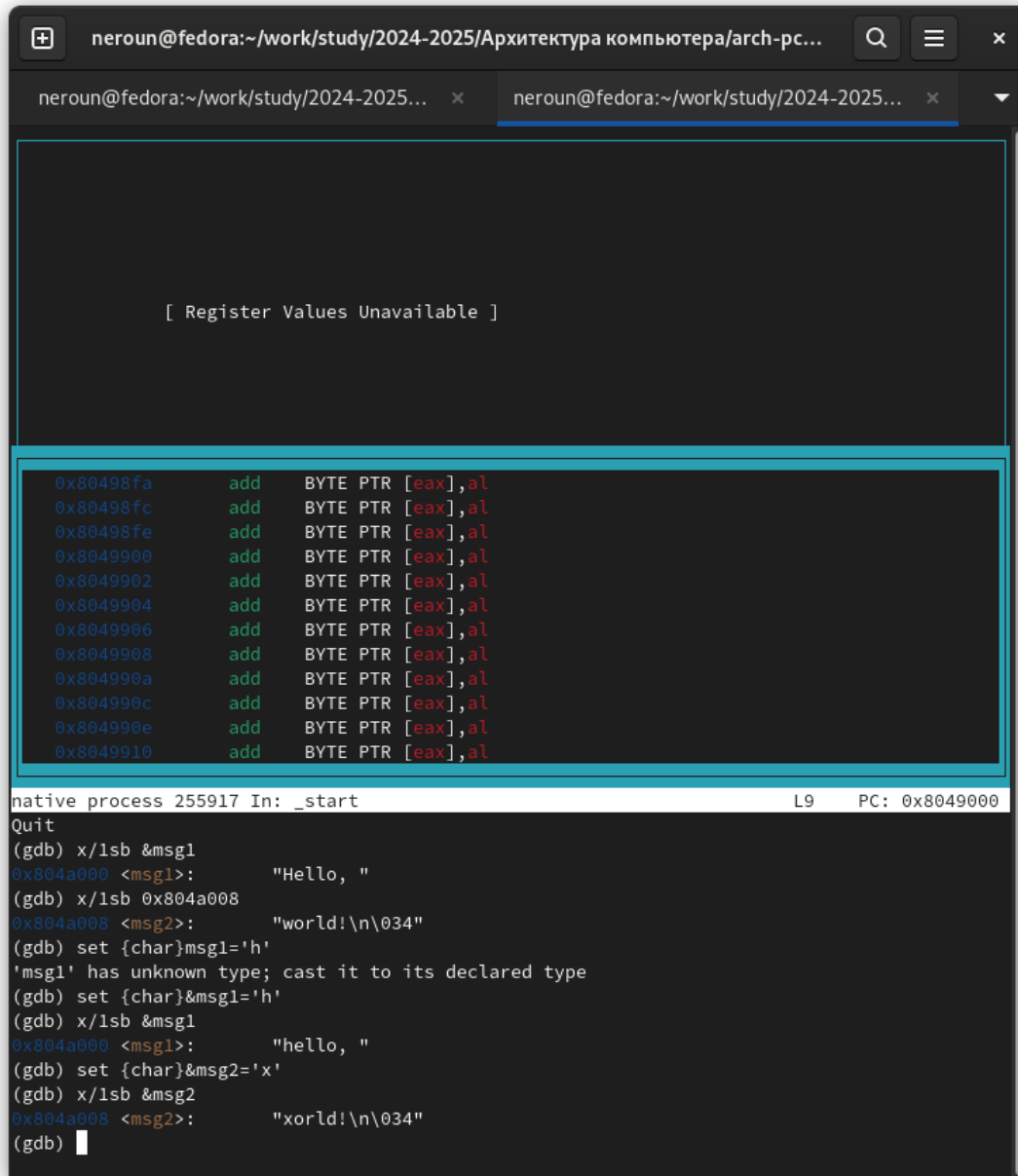
```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc...
neroun@fedora:~/work/study/2024-2025... x neroun@fedora:~/work/study/2024-2025... x

[ Register Values Unavailable ]

0x80498fa    add    BYTE PTR [eax],al
0x80498fc    add    BYTE PTR [eax],al
0x80498fe    add    BYTE PTR [eax],al
0x8049900    add    BYTE PTR [eax],al
0x8049902    add    BYTE PTR [eax],al
0x8049904    add    BYTE PTR [eax],al
0x8049906    add    BYTE PTR [eax],al
0x8049908    add    BYTE PTR [eax],al
0x804990a    add    BYTE PTR [eax],al
0x804990c    add    BYTE PTR [eax],al
0x804990e    add    BYTE PTR [eax],al
0x8049910    add    BYTE PTR [eax],al

native process 255917 In: _start L9 PC: 0x8049000
esi         0x0          0
edi         0x0          0
eip         0x8049000    0x8049000 <_start>
eflags     0x202        [ IF ]
cs          0x23         35
ss          0x2b         43
ds          0x2b         43
--Type <RET> for more, q to quit, c to continue without paging--q
Quit
(gdb) x/1sb &msg1
0x804a000 <msg1>:      "Hello, "
(gdb) x/1sb 0x804a008
0x804a008 <msg2>:      "world!\n\034"
(gdb) █
```

Меняю содержимое переменных по имени и по адресу.



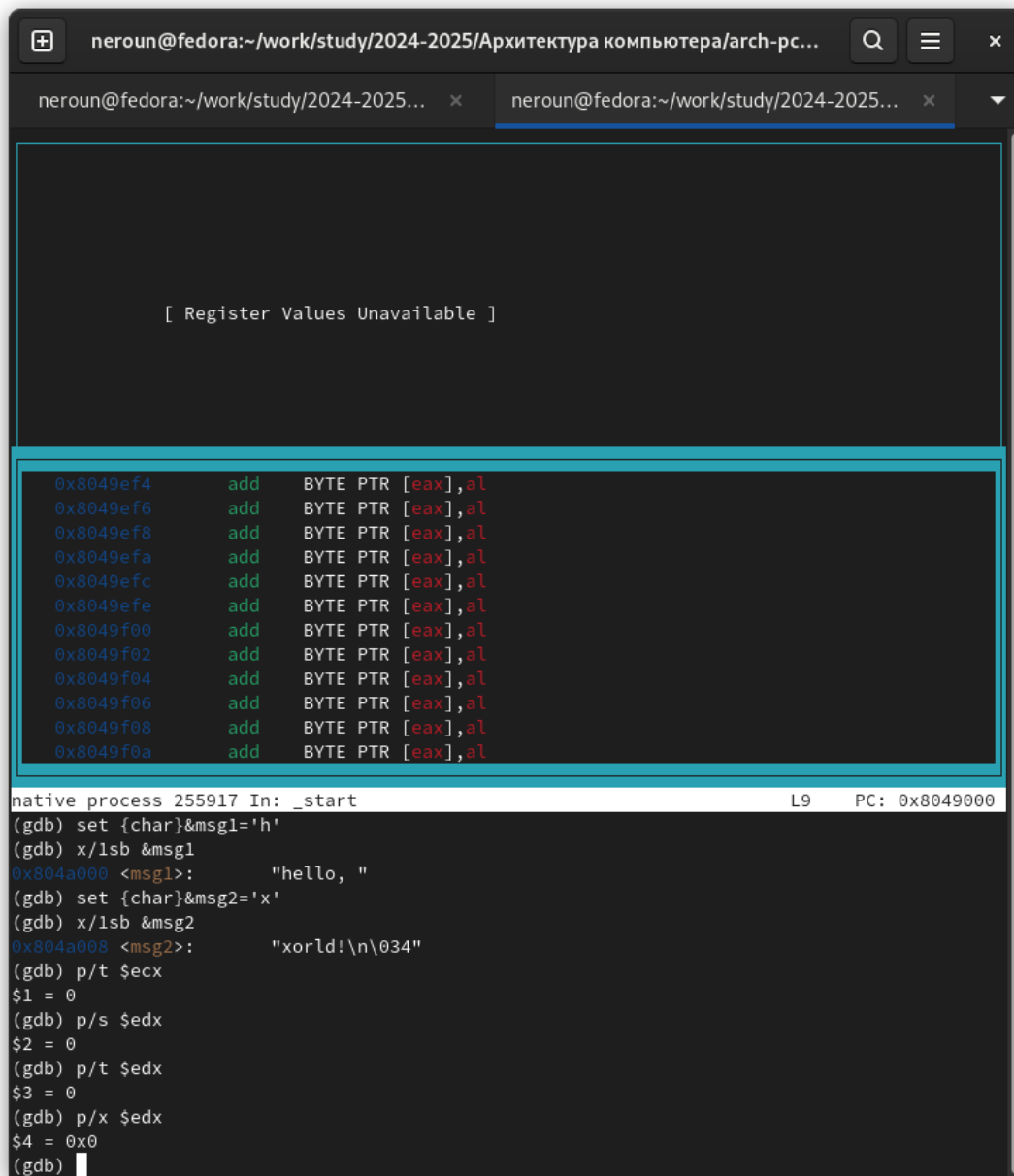
The screenshot shows a GDB terminal window with the following content:

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-рс...
neroun@fedora:~/work/study/2024-2025... x neroun@fedora:~/work/study/2024-2025... x
[ Register Values Unavailable ]

0x80498fa  add  BYTE PTR [eax],al
0x80498fc  add  BYTE PTR [eax],al
0x80498fe  add  BYTE PTR [eax],al
0x8049900  add  BYTE PTR [eax],al
0x8049902  add  BYTE PTR [eax],al
0x8049904  add  BYTE PTR [eax],al
0x8049906  add  BYTE PTR [eax],al
0x8049908  add  BYTE PTR [eax],al
0x804990a  add  BYTE PTR [eax],al
0x804990c  add  BYTE PTR [eax],al
0x804990e  add  BYTE PTR [eax],al
0x8049910  add  BYTE PTR [eax],al

native process 255917 In: _start L9 PC: 0x8049000
Quit
(gdb) x/1sb &msg1
0x804a000 <msg1>: "Hello, "
(gdb) x/1sb 0x804a008
0x804a008 <msg2>: "world!\n\034"
(gdb) set {char}msg1='h'
'msg1' has unknown type; cast it to its declared type
(gdb) set {char}&msg1='h'
(gdb) x/1sb &msg1
0x804a000 <msg1>: "hello, "
(gdb) set {char}&msg2='x'
(gdb) x/1sb &msg2
0x804a008 <msg2>: "xorld!\n\034"
(gdb) █
```

Вывожу в различных форматах значение регистра edx.



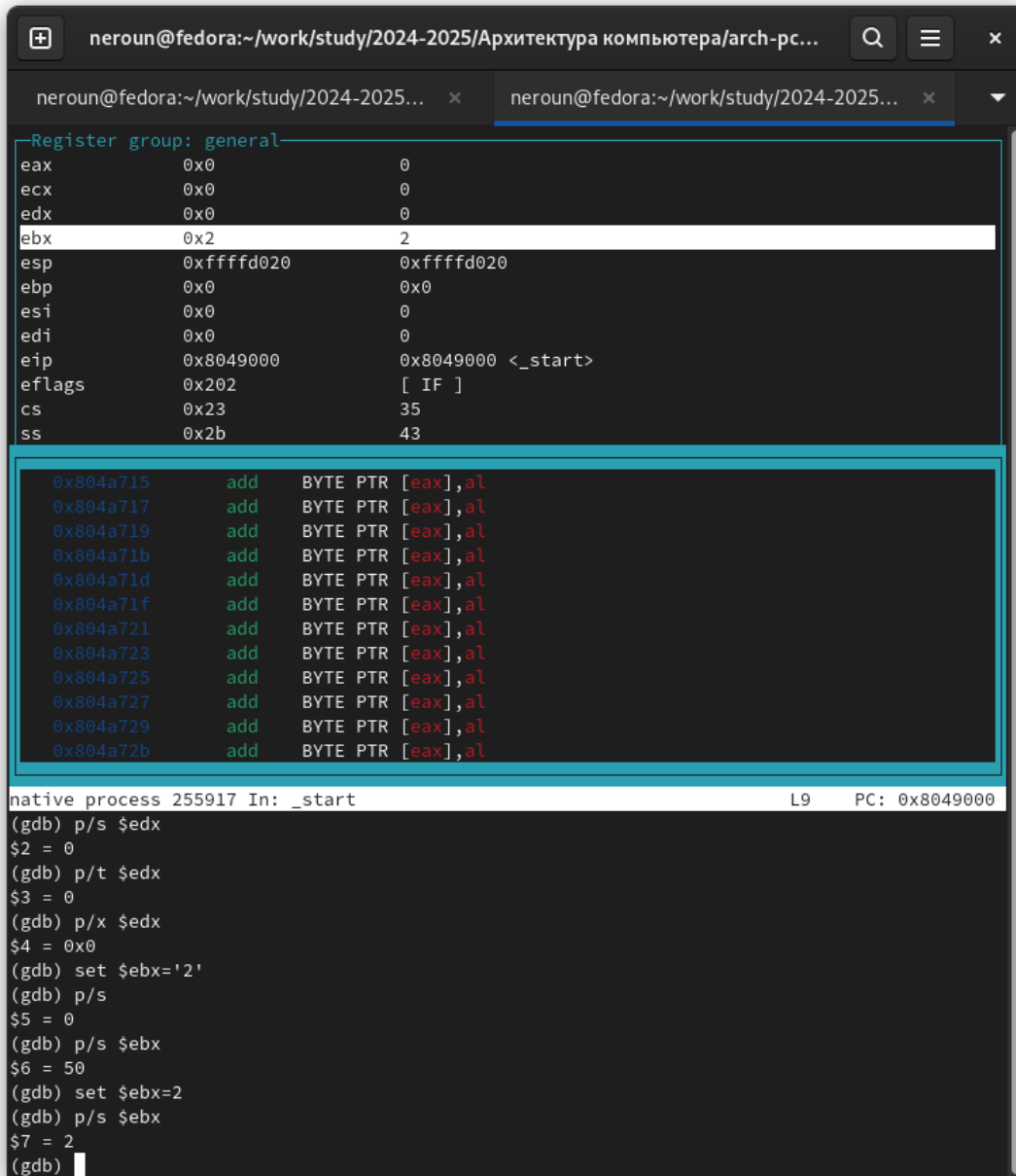
The screenshot shows a GDB terminal window with the following content:

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc...
neroun@fedora:~/work/study/2024-2025... x neroun@fedora:~/work/study/2024-2025... x
[ Register Values Unavailable ]

0x8049ef4 add BYTE PTR [eax],al
0x8049ef6 add BYTE PTR [eax],al
0x8049ef8 add BYTE PTR [eax],al
0x8049efa add BYTE PTR [eax],al
0x8049efc add BYTE PTR [eax],al
0x8049efe add BYTE PTR [eax],al
0x8049f00 add BYTE PTR [eax],al
0x8049f02 add BYTE PTR [eax],al
0x8049f04 add BYTE PTR [eax],al
0x8049f06 add BYTE PTR [eax],al
0x8049f08 add BYTE PTR [eax],al
0x8049f0a add BYTE PTR [eax],al

native process 255917 In: _start L9 PC: 0x8049000
(gdb) set {char}&msg1='h'
(gdb) x/1sb &msg1
0x804a000 <msg1>: "hello, "
(gdb) set {char}&msg2='x'
(gdb) x/1sb &msg2
0x804a008 <msg2>: "xor!d!\n\034"
(gdb) p/t $ecx
$1 = 0
(gdb) p/s $edx
$2 = 0
(gdb) p/t $edx
$3 = 0
(gdb) p/x $edx
$4 = 0x0
(gdb)
```

С помощью команды set меняю содержимое регистра ebx.



The screenshot shows a GDB session window with two tabs. The top tab displays the 'Register group: general' window, which lists various registers and their values. The 'ebx' register is highlighted with a white background and shows a value of 0x2. Below the register window, a list of assembly instructions is shown, all starting with 'add BYTE PTR [eax], al'. The bottom tab shows the GDB command window with the following commands and output:

```
native process 255917 In: _start L9 PC: 0x8049000
(gdb) p/s $edx
$2 = 0
(gdb) p/t $edx
$3 = 0
(gdb) p/x $edx
$4 = 0x0
(gdb) set $ebx='2'
(gdb) p/s
$5 = 0
(gdb) p/s $ebx
$6 = 50
(gdb) set $ebx=2
(gdb) p/s $ebx
$7 = 2
(gdb)
```

3.1.4 Обработка аргументов командной строки в GDB

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

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ nasm -f elf -g -l lab9-3.1
st lab9-3.asm
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ld -m elf_i386 -o lab9-3 l
ab9-3.o
```

Запускаю программу в режиме отладки с указанием аргументов, указываю брейкпоинт и запускаю отладку. Проверяю работу стека, изменяя аргумент команды просмотра регистра esp на +4, число обусловлено разрядностью

системы, а указатель void занимает как раз 4 байта, ошибка при аргументе +24 означает, что аргументы на вход программы закончились.

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09
neroun@fedora:~/work/study/2024-2025/Архи... x neroun@fedora:~/work/study/2024-2025/Архи... x
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ gdb --args lab9-3 "аргумент1" "аргумент2" 2 "аргумент 3"
GNU gdb (Fedora Linux) 14.2-1.fc40
Copyright (C) 2023 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/>.

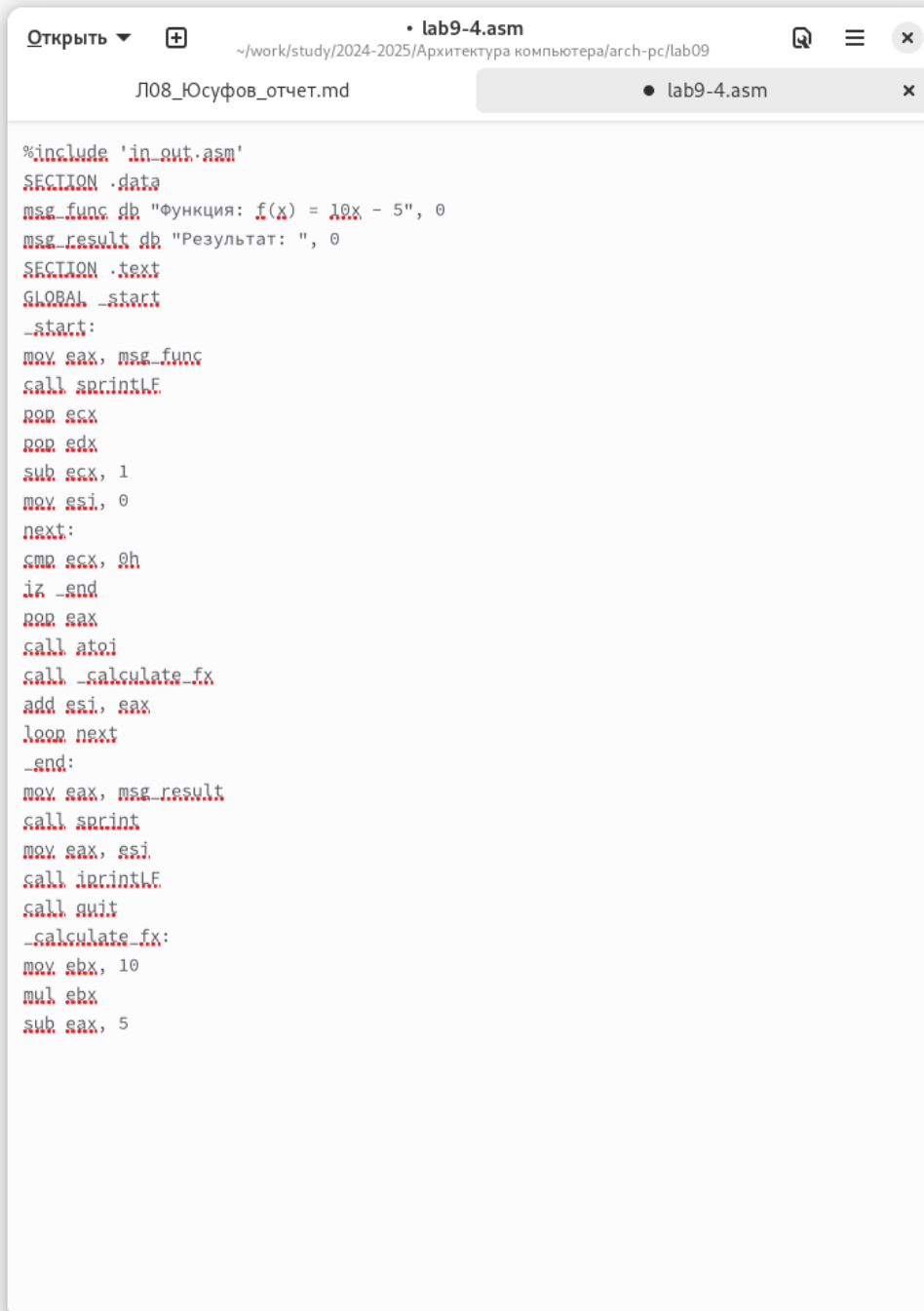
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab9-3...
(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab9-3.asm, line 5.
(gdb) run
Starting program: /home/neroun/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09/lab9-3 аргумент1 аргумент2 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 lab9-3.asm:5
5      pop ecx ; Извлекаем из стека в `ecx` количество
(gdb) x/s *(void**)(esp + 4)
0xffffd1a0: "/home/neroun/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09/lab9-3"
(gdb) x/s *(void**)(esp + 8)
0xffffd203: "аргумент1"
(gdb) x/s *(void**)(esp + 12)
0xffffd215: "аргумент2"
(gdb) x/s *(void**)(esp + 16)
0xffffd227: "2"
(gdb) x/s *(void**)(esp + 20)
0xffffd229: "аргумент 3"
(gdb) x/s *(void**)(esp + 24)
0x0: <error: Cannot access memory at address 0x0>
(gdb)
```


3.2 Задание для самостоятельной работы

Меняю программу самостоятельной части предыдущей лабораторной работы с использованием подпрограммы.



```
%include 'in_out.asm'
SECTION .data
msg_func db "Функция: f(x) = 10x - 5", 0
msg_result db "Результат: ", 0
SECTION .text
GLOBAL _start
_start:
mov eax, msg_func
call sprintf
pop ecx
pop edx
sub ecx, 1
mov esi, 0
next:
cmp ecx, 0h
jz _end
pop eax
call atoi
call _calculate_fx
add esi, eax
loop next
_end:
mov eax, msg_result
call sprintf
mov eax, esi
call iprintf
call quit
_calculate_fx:
mov ebx, 10
mul ebx
sub eax, 5
```

Код программы:

```

#include 'in_out.asm'
SECTION .data
msg_func db "Функция: f(x) = 10x - 5", 0
msg_result db "Результат: ", 0
SECTION .text
GLOBAL _start
_start:
mov eax, msg_func
call sprintLF
pop ecx
pop edx
sub ecx, 1
mov esi, 0
next:
cmp ecx, 0h
jz _end
pop eax
call atoi
call _calculate_fx
add esi, eax
loop next
_end:
mov eax, msg_result
call sprint
mov eax, esi
call iprintLF
call quit
_calculate_fx:
mov ebx, 10
mul ebx
sub eax, 5

```

Запускаю программу в режиме отладчика и пошагово через si просматриваю изменение значений регистров через i r. При выполнении инструкции mul esx 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.

```
--Register group: general
eax      0x2      2      ecx      0x4      4
edx      0x0      0      ebx      0x5      5
esp      0xffffcf10 0xffffcf10  ebp      0x0      0x0
esi      0x0      0      edi      0x0      0
eip      0x80490f9 0x80490f9 <_start+17>  eflags   0x206    [ PF IF ]
cs       0x23     35     ss       0x2b     43
ds       0x2b     43     es       0x2b     43
fs       0x0      0      gs       0x0      0

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

native process 8526 (asm) In: _start L14 PC: 0x80490f9
eax      0x2      2
ecx      0x4      4
edx      0x0      0
ebx      0x5      5
esp      0xffffcf10 0xffffcf10
ebp      0x0      0x0
esi      0x0      0
edi      0x0      0
eip      0x80490f9 0x80490f9 <_start+17>
eflags   0x206    [ PF IF ]
cs       0x23     35
--Type <RET> for more, q to quit, c to continue without paging--
```

Исправляю найденную ошибку, теперь программа верно считает значение функции.

```
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ nasm -f elf lab9-5.asm
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ld -m elf_i386 -o lab9-5 l
ab9-5.o
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$ ./lab9-5
Результат: 25
neroun@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc/lab09$
```

Код измененной 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
```

4 Выводы

В результате выполнения данной лабораторной работы я приобрел навыки написания программ с использованием подпрограмм, а также познакомился с методами отладки при помощи GDB и его основными возможностями.