

Отчёт по лабораторной работе 9

Архитектура компьютера

Душаев Азимбек Юсуфович НКАбд-02-23

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	28

Список иллюстраций

2.1	Программа lab9-1.asm	7
2.2	Запуск программы lab9-1.asm	8
2.3	Программа lab9-1.asm	9
2.4	Запуск программы lab9-1.asm	10
2.5	Программа lab9-2.asm	11
2.6	Запуск программы lab9-2.asm в отладчике	12
2.7	Дизассемблированный код	13
2.8	Дизассемблированный код в режиме интел	14
2.9	Точка остановки	15
2.10	Изменение регистров	16
2.11	Изменение регистров	17
2.12	Изменение значения переменной	18
2.13	Вывод значения регистра	19
2.14	Вывод значения регистра	20
2.15	Вывод значения регистра	21
2.16	Программа prog1.asm	22
2.17	Запуск программы prog1.asm	23
2.18	Код с ошибкой	24
2.19	Отладка	25
2.20	Код исправлен	26
2.21	Проверка работы	27

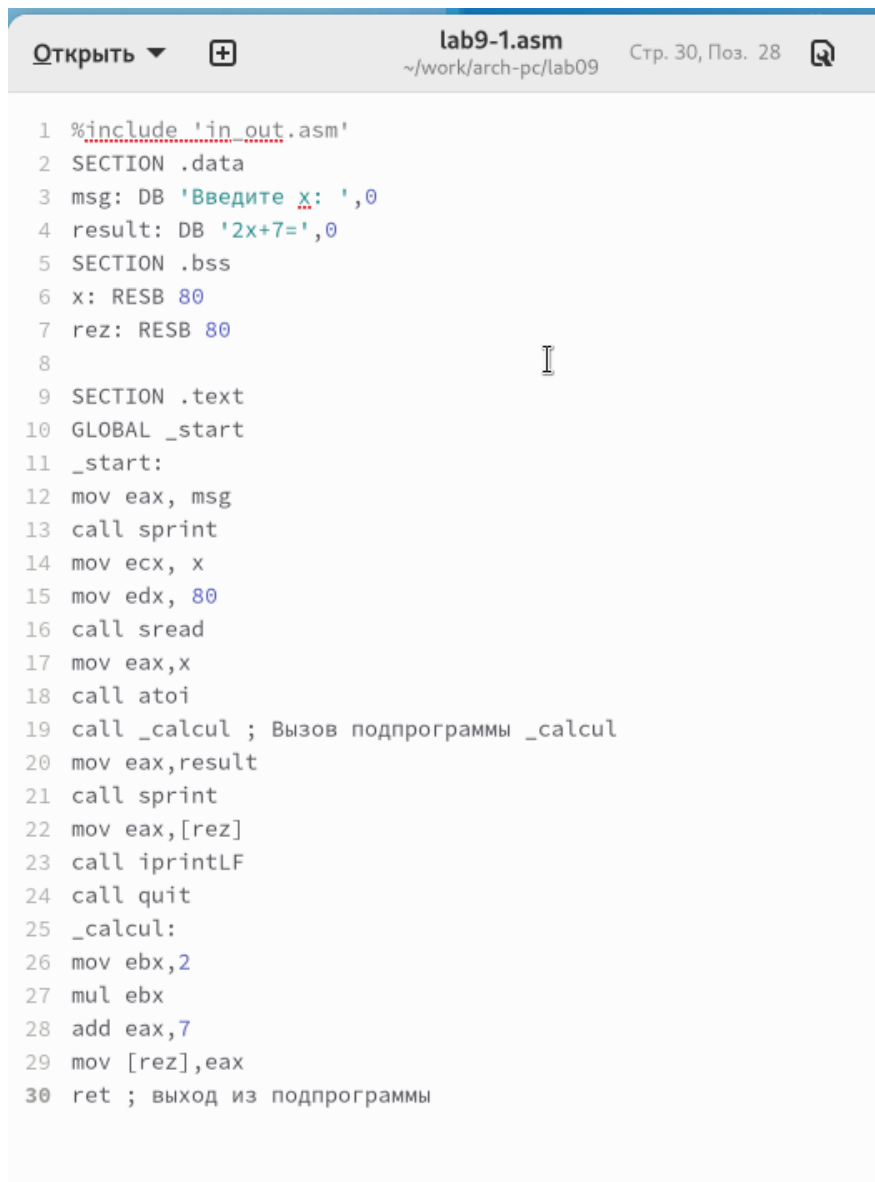
Список таблиц

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

1. Создал каталог для выполнения лабораторной работы № 9 и перешел в него. Затем создал файл lab9-1.asm.
2. В качестве примера рассмотрим программу, которая вычисляет арифметическое выражение $f(x) = 2x + 7$ с помощью подпрограммы calcul. В данном примере значение x вводится с клавиатуры, а само выражение вычисляется внутри подпрограммы.



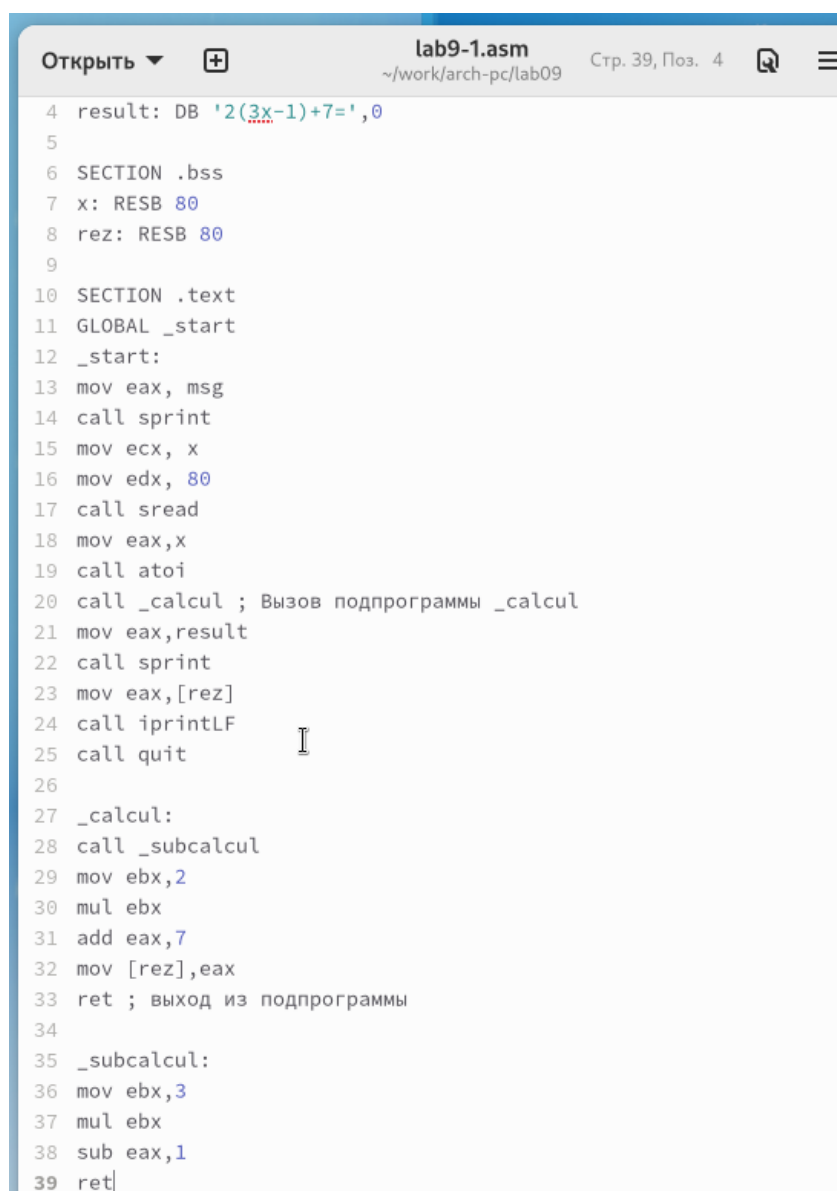
```
1 %include 'in_out.asm'
2 SECTION .data
3 msg: DB 'Введите x: ',0
4 result: DB '2x+7=',0
5 SECTION .bss
6 x: RESB 80
7 rez: RESB 80
8
9 SECTION .text
10 GLOBAL _start
11 _start:
12 mov eax, msg
13 call sprint
14 mov ecx, x
15 mov edx, 80
16 call sread
17 mov eax,x
18 call atoi
19 call _calcul ; Вызов подпрограммы _calcul
20 mov eax,result
21 call sprint
22 mov eax,[rez]
23 call iprintLF
24 call quit
25 _calcul:
26 mov ebx,2
27 mul ebx
28 add eax,7
29 mov [rez],eax
30 ret ; выход из подпрограммы
```

Рис. 2.1: Программа lab9-1.asm

```
[adushaev@fedora lab09]$ nasm -f elf lab9-1.asm
[adushaev@fedora lab09]$ ld -m elf_i386 -o lab9-1 lab9-1.o
[adushaev@fedora lab09]$ ./lab9-1
Введите x: 4
2x+7=15
[adushaev@fedora lab09]$
```

Рис. 2.2: Запуск программы lab9-1.asm

3. Внесены изменения в текст программы, добавлена подпрограмма `subcalcul` внутри подпрограммы `calcul` для вычисления выражения $f(g(x))$, где значение x также вводится с клавиатуры, а функции $f(x) = 2x + 7$ и $g(x) = 3x - 1$ вычисляются внутри подпрограмм.



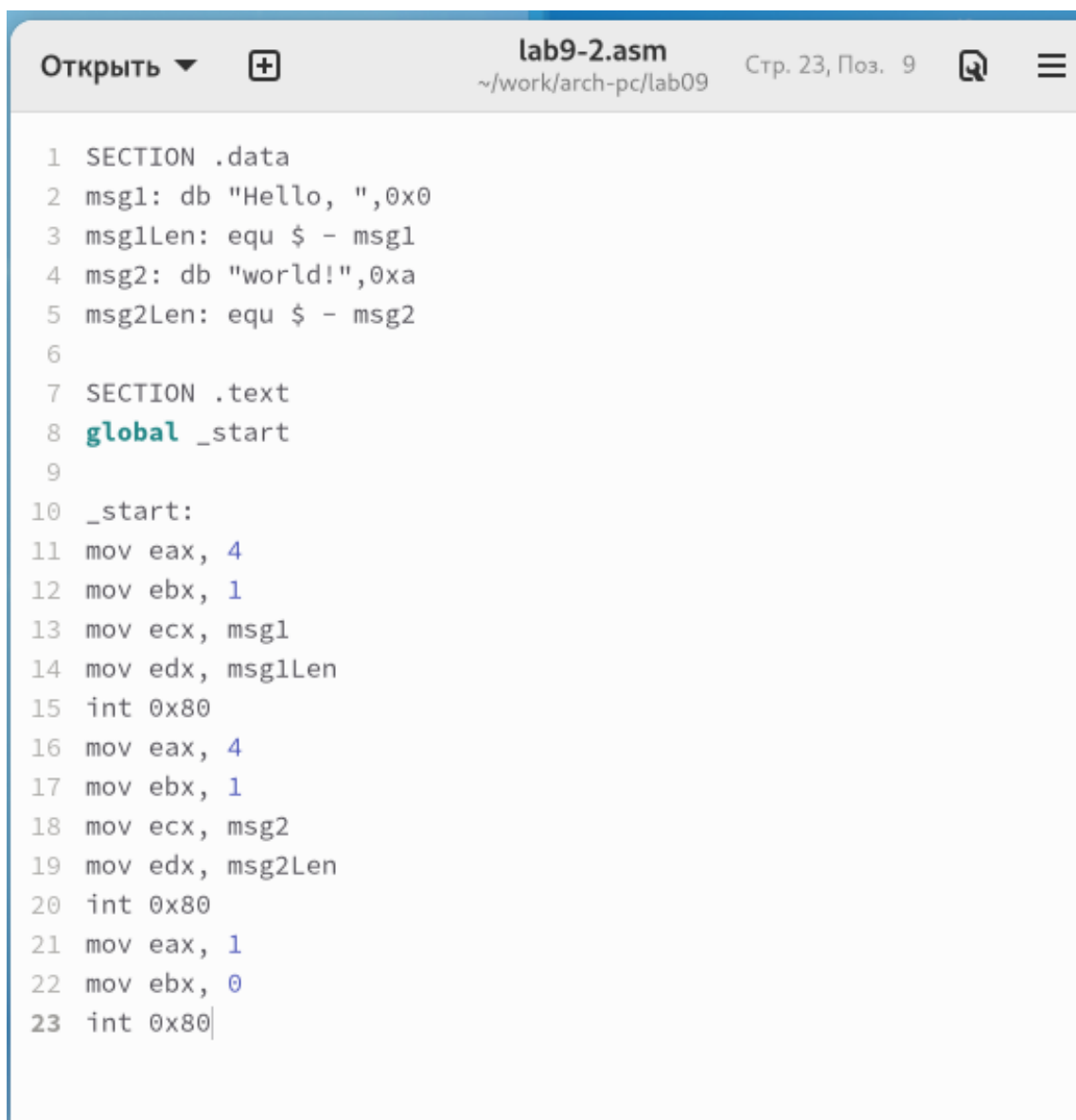
```
4 result: DB '2(3x-1)+7=',0
5
6 SECTION .bss
7 x: RESB 80
8 rez: RESB 80
9
10 SECTION .text
11 GLOBAL _start
12 _start:
13 mov eax, msg
14 call sprint
15 mov ecx, x
16 mov edx, 80
17 call sread
18 mov eax, x
19 call atoi
20 call _calcul ; Вызов подпрограммы _calcul
21 mov eax, result
22 call sprint
23 mov eax, [rez]
24 call iprintLF
25 call quit
26
27 _calcul:
28 call _subcalcul
29 mov ebx, 2
30 mul ebx
31 add eax, 7
32 mov [rez], eax
33 ret ; выход из подпрограммы
34
35 _subcalcul:
36 mov ebx, 3
37 mul ebx
38 sub eax, 1
39 ret
```

Рис. 2.3: Программа lab9-1.asm

```
[adushaev@fedora lab09]$ nasm -f elf lab9-1.asm
[adushaev@fedora lab09]$ ld -m elf_i386 -o lab9-1 lab9-1.o
[adushaev@fedora lab09]$ ./lab9-1
Введите x: 4
 $2(3x-1)+7=29$ 
[adushaev@fedora lab09]$
```

Рис. 2.4: Запуск программы lab9-1.asm

4. Создан файл lab9-2.asm с текстом программы из Листинга 9.2, который представляет программу печати сообщения “Hello world!”.



```
1 SECTION .data
2 msg1: db "Hello, ",0x0
3 msg1Len: equ $ - msg1
4 msg2: db "world!",0xa
5 msg2Len: equ $ - msg2
6
7 SECTION .text
8 global _start
9
10 _start:
11 mov eax, 4
12 mov ebx, 1
13 mov ecx, msg1
14 mov edx, msg1Len
15 int 0x80
16 mov eax, 4
17 mov ebx, 1
18 mov ecx, msg2
19 mov edx, msg2Len
20 int 0x80
21 mov eax, 1
22 mov ebx, 0
23 int 0x80
```

Рис. 2.5: Программа lab9-2.asm

Был получен исполняемый файл. Для возможности работы с отладчиком GDB необходимо добавить отладочную информацию в исполняемый файл, для этого трансляция программы должна быть выполнена с использованием ключа “-g”.

Загрузка исполняемого файла в отладчик GDB была выполнена успешно. Для проверки работоспособности программы была использована команда run (сокращенно r) в оболочке GDB.

```

[adushaev@fedora lab09]$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
[adushaev@fedora lab09]$ ld -m elf_i386 -o lab9-2 lab9-2.o
[adushaev@fedora lab09]$ gdb lab9-2
GNU gdb (GDB) Fedora 12.1-2.fc36
Copyright (C) 2022 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) r
Starting program: /home/adushaev/work/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])
Debuginfod has been disabled.
To make this setting permanent, add 'set debuginfod enabled off' to .gdbinit.
Hello, world!
[Inferior 1 (process 3382) exited normally]
(gdb)

```

Рис. 2.6: Запуск программы lab9-2.asm в отладчике

Для более детального анализа программы, установите точку остановки на метке “start”, с которой начинается выполнение любой ассемблерной программы, и запустите программу. Затем просмотрите дизассемблированный код программы.

```
adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2

This GDB supports auto-downloading debuginfo from the following URLs:
https://debuginfod.fedoraproject.org/
Enable debuginfod for this session? (y or [n])
Debuginfod has been disabled.
To make this setting permanent, add 'set debuginfod enabled off' to .gdbinit.
Hello, world!
[Inferior 1 (process 3382) exited normally]
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab9-2.asm, line 11.
(gdb) r
Starting program: /home/adushaev/work/arch-pc/lab09/lab9-2

Breakpoint 1, _start () at lab9-2.asm:11
11      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. 2.7: Дизассемблированный код

```
adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2

=> 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
(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: Дизассемблированный код в режиме интел

На предыдущих шагах была установлена точка остановки на метке “_start”. Чтобы проверить это, воспользуйтесь командой “info breakpoints” (сокращенно “i b”). Добавьте еще одну точку остановки на адресе инструкции. Адрес инструкции можно увидеть в средней части экрана в левом столбце соответствующей инструкции. Определите адрес предпоследней инструкции “mov ebx, 0x0” и установите точку остановки.

The screenshot shows a GDB terminal window with the title bar "adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2". The main window displays "[Register Values Unavailable]". A list of assembly instructions is shown, with the first instruction highlighted in a blue box:

```
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
0x8049020 <_start+32>   mov     ecx,0x804a008
0x8049025 <_start+37>   mov     edx,0x7
```

Below the assembly list, the terminal shows the following commands and output:

```
native process 3388 In: _start L11 PC: 0x8049000
(gdb) layout regs
(gdb) b *0x8049031
Breakpoint 2 at 0x8049031: file lab9-2.asm, line 22.
(gdb) i b
Num    Type           Disp Enb Address      What
1      breakpoint      keep y   0x08049000 lab9-2.asm:11
       breakpoint already hit 1 time
2      breakpoint      keep y   0x08049031 lab9-2.asm:22
(gdb)
```

Рис. 2.9: Точка остановки

Отладчик может отображать содержимое ячеек памяти и регистров, а также позволяет вручную изменять значения регистров и переменных. Выполните 5 инструкций с помощью команды “stepi” (или “si”) и проследите за изменением значений регистров.

```
adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2

Register group: general
eax      0x4      4
ecx      0x0      0
edx      0x0      0
ebx      0x0      0
esp      0xffffd230 0xffffd230
ebp      0x0      0x0
esi      0x0      0
edi      0x0      0

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
0x8049020 <_start+32> mov ecx,0x804a008
0x8049025 <_start+37> mov edx,0x7

native process 3388 In: _start L12 PC: 0x8049005
edi      0x0      0
eip      0x8049000 0x8049000 <_start>
eflags   0x202    [ IF ]
--Type <RET> for more, q to quit, c to continue without paging-- 0x23 35
ss       0x2b     43
ds       0x2b     43
es       0x2b     43
fs       0x0      0
gs       0x0      0
(gdb) si
(gdb)
```

Рис. 2.10: Изменение регистров

The screenshot shows a GDB terminal window with the title bar "adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2". The window is divided into three main sections. The top section, titled "Register group: general", lists the values of general-purpose registers: `eax` (0x8, 8), `ecx` (0x804a000, 134520832), `edx` (0x8, 8), `ebx` (0x1, 1), `esp` (0xffffd230, 0xffffd230), `ebp` (0x0, 0x0), `esi` (0x0, 0), and `edi` (0x0, 0). The middle section displays assembly code with addresses and instructions: `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` (highlighted), `0x804901b <_start+27> mov ebx,0x1`, `0x8049020 <_start+32> mov ecx,0x804a008`, and `0x8049025 <_start+37> mov edx,0x7`. The bottom section shows the state of the native process: `native process 3388 In: _start` with segment registers `ss`, `ds`, `es`, `fs`, and `gs` all set to 0x2b or 0x0, and the instruction pointer `PC` at `0x8049016`. The bottom-most part of the window shows several `(gdb) si` commands.

Рис. 2.11: Изменение регистров

Просмотрите значение переменной “msg1” по имени и значение переменной “msg2” по адресу.

Чтобы изменить значение регистра или ячейки памяти, воспользуйтесь командой “set”, указав имя регистра или адрес в качестве аргумента. Измените первый символ переменной “msg1”.

```
adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2

Register group: general
eax      0x8      8
ecx      0x804a000 134520832
edx      0x8      8
ebx      0x1      1
esp      0xffffd230 0xffffd230
ebp      0x0      0
esi      0x0      0
edi      0x0      0

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
0x8049020 <_start+32>  mov     ecx,0x804a008
0x8049025 <_start+37>  mov     edx,0x7

native process 3388 In: _start L16 PC: 0x8049016
(gdb) x/1sb &msg1
0x804a000 <msg1>: "Hello, "
(gdb) x/1sb 0x804a008
0x804a008 <msg2>: "world!\n\034"
(gdb) set {char}&msg1='h'
(gdb) x/1sb &msg1
0x804a000 <msg1>: "hello, "
(gdb) set {char}0x804a008='L'
(gdb) x/1sb 0x804a008
0x804a008 <msg2>: "Lor!d!\n\034"
(gdb) 
```

Рис. 2.12: Изменение значения переменной

Выведите значение регистра “edx” в различных форматах (шестнадцатеричном, двоичном и символьном).

The screenshot shows a GDB terminal window with the title bar "adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2". The window is divided into three main sections. The top section, titled "Register group: general", displays the current values of general-purpose registers:

eax	0x8	8
ecx	0x804a000	134520832
edx	0x8	8
ebx	0x1	1
esp	0xffffd230	0xffffd230
ebp	0x0	0x0
esi	0x0	0
edi	0x0	0

. The middle section shows assembly code with addresses and instructions:

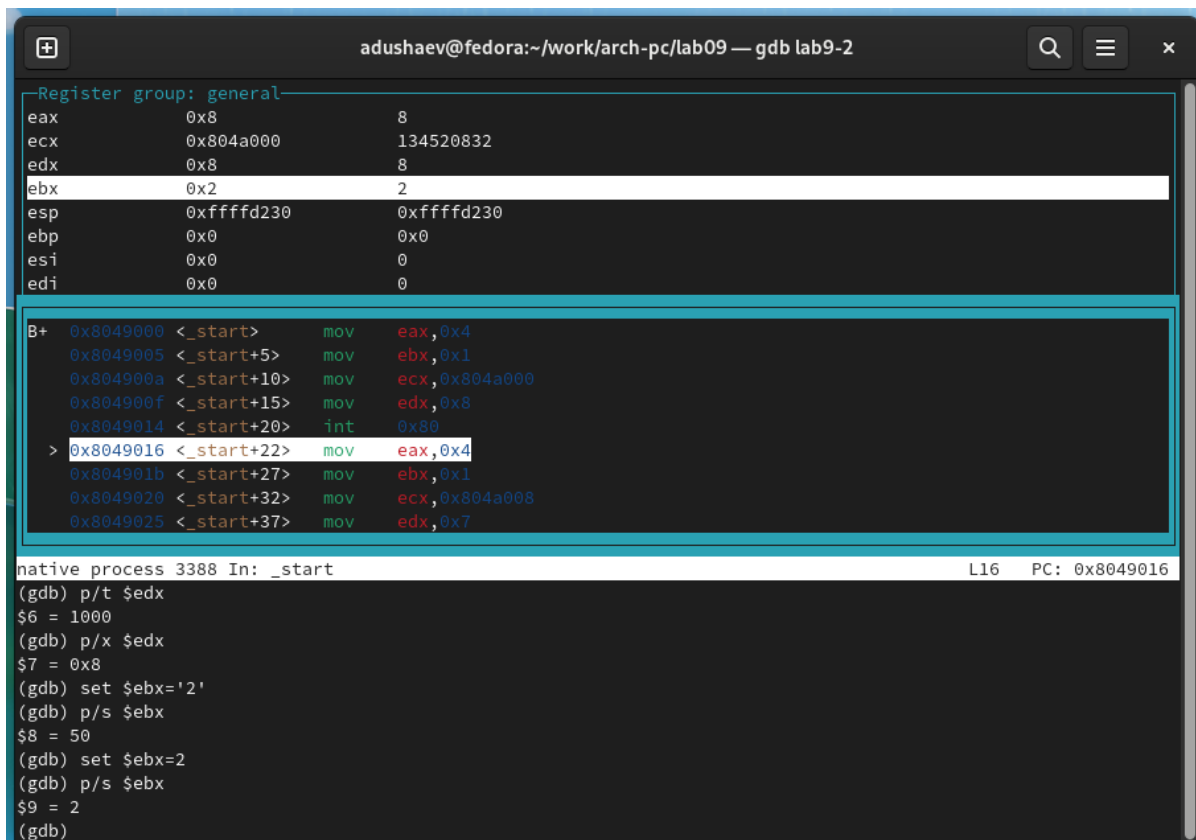
```
B+ 0x8049000 <_start>    mov    eax,0x4
0x8049005 <_start+5>    mov    ebx,0x1
0x804900a <_start+10>   mov    ecx,0x804a000
0x804900f <_start+15>   mov    edx,0x8
0x8049014 <_start+20>   int     0x80
> 0x8049016 <_start+22>  mov    eax,0x4
0x804901b <_start+27>   mov    ebx,0x1
0x8049020 <_start+32>   mov    ecx,0x804a008
0x8049025 <_start+37>   mov    edx,0x7
```

. The bottom section shows the command prompt and the execution of several GDB commands:

```
native process 3388 In: _start L16 PC: 0x8049016
(gdb) p/s $ecx
$3 = 134520832
(gdb) p/x $ecx
$4 = 0x804a000
(gdb) p/s $edx
$5 = 8
(gdb) p/t $edx
$6 = 1000
(gdb) p/x $edx
$7 = 0x8
(gdb) 
```

Рис. 2.13: Вывод значения регистра

Используя команду “set”, измените значение регистра “ebx”.



```
adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-2

Register group: general
eax      0x8      8
ecx      0x804a000 134520832
edx      0x8      8
ebx      0x2      2
esp      0xffffd230 0xffffd230
ebp      0x0      0x0
esi      0x0      0
edi      0x0      0

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
0x8049020 <_start+32>   mov     ecx,0x804a008
0x8049025 <_start+37>   mov     edx,0x7

native process 3388 In: _start L16 PC: 0x8049016
(gdb) p/t $edx
$6 = 1000
(gdb) p/x $edx
$7 = 0x8
(gdb) set $ebx='2'
(gdb) p/s $ebx
$8 = 50
(gdb) set $ebx=2
(gdb) p/s $ebx
$9 = 2
(gdb)
```

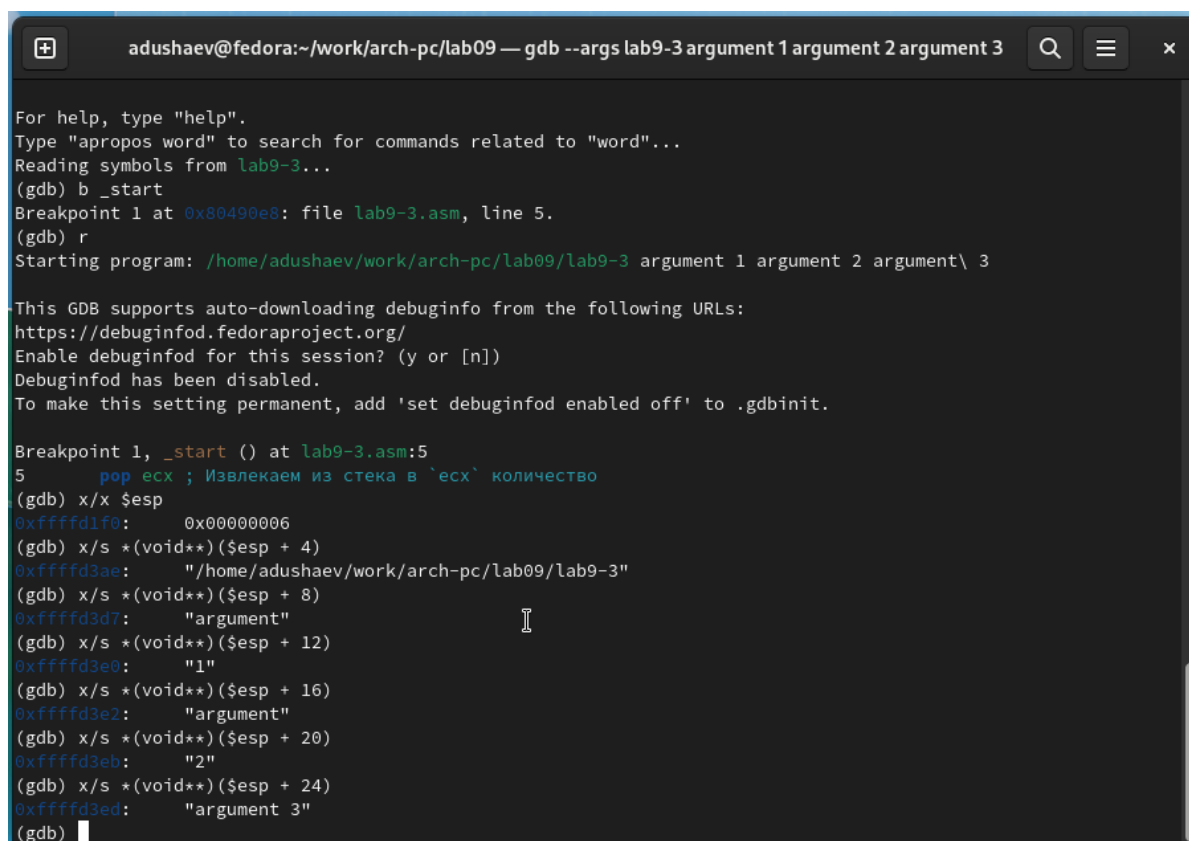
Рис. 2.14: Вывод значения регистра

5. Скопировал файл lab8-2.asm, созданный в процессе выполнения лабораторной работы №8, который содержит программу для вывода аргументов командной строки. Затем создал исполняемый файл. Для загрузки программы с аргументами в отладчик GDB необходимо использовать ключ “-args”. Загрузил исполняемый файл в отладчик, указав аргументы командной строки.

Для начала установил точку останова перед первой инструкцией в программе и запустил ее.

Адрес вершины стека, который хранится в регистре ESP, указывает на число аргументов командной строки (включая имя программы), которое находится по этому адресу. В данном случае число аргументов равно 5, включая имя программы “lab9-3” и сами аргументы: “аргумент1”, “аргумент2” и “аргумент3”.

Далее были просмотрены остальные позиции в стеке. По адресу [ESP+4] находится адрес в памяти, где располагается имя программы, по адресу [ESP+8] хранится адрес первого аргумента, по адресу [ESP+12] – второго аргумента и так далее.

A screenshot of a GDB terminal window. The title bar shows the user 'adushaev@fedora' and the command 'gdb --args lab9-3 argument 1 argument 2 argument 3'. The terminal output shows the GDB startup sequence, including reading symbols from 'lab9-3.asm' and starting the program. A breakpoint is set at the '_start' function. The user then runs the program. The GDB prompt shows the current location is 'lab9-3.asm:5'. The user enters the command 'x/x \$esp' to display the stack contents. The output shows a series of memory addresses and their corresponding values: '0x00000006', '/home/adushaev/work/arch-pc/lab09/lab9-3', 'argument', '1', 'argument', '2', and 'argument 3'. The user's cursor is positioned at the end of the last line of output.

```
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) r
Starting program: /home/adushaev/work/arch-pc/lab09/lab9-3 argument 1 argument 2 argument\ 3

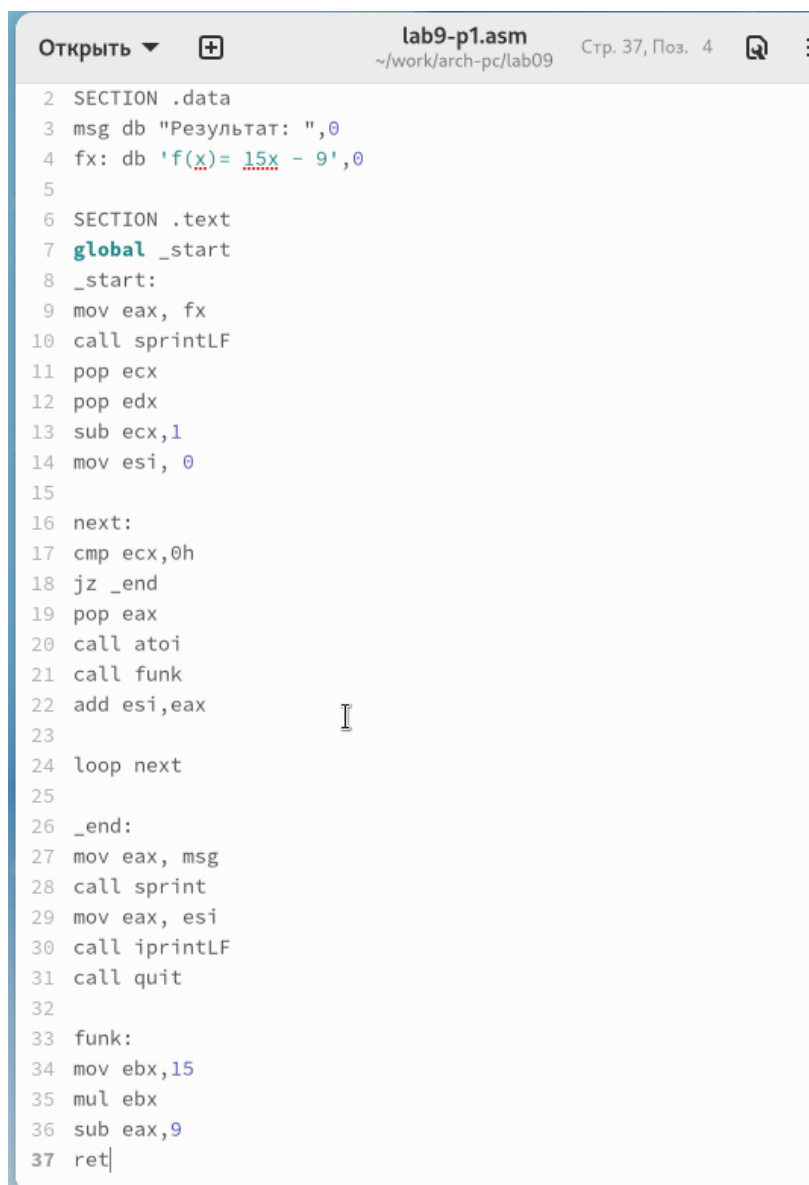
This GDB supports auto-downloading debuginfo from the following URLs:
https://debuginfod.fedoraproject.org/
Enable debuginfod for this session? (y or [n])
Debuginfod has been disabled.
To make this setting permanent, add 'set debuginfod enabled off' to .gdbinit.

Breakpoint 1, _start () at lab9-3.asm:5
5      pop ecx ; Извлекаем из стека в `ecx` количество
(gdb) x/x $esp
0xffffd1f0: 0x00000006
(gdb) x/s *(void**)($esp + 4)
0xffffd3ae: "/home/adushaev/work/arch-pc/lab09/lab9-3"
(gdb) x/s *(void**)($esp + 8)
0xffffd3d7: "argument"
(gdb) x/s *(void**)($esp + 12)
0xffffd3e0: "1"
(gdb) x/s *(void**)($esp + 16)
0xffffd3e2: "argument"
(gdb) x/s *(void**)($esp + 20)
0xffffd3eb: "2"
(gdb) x/s *(void**)($esp + 24)
0xffffd3ed: "argument 3"
(gdb)
```

Рис. 2.15: Вывод значения регистра

Шаг изменения адреса равен 4, так как каждая следующая позиция в стеке смещается на размер переменной, который составляет 4 байта.

6. Внес изменения в программу из лабораторной работы №8 (Задание №1 для самостоятельной работы), чтобы реализовать вычисление значения функции $f(x)$ в виде подпрограммы.



```
Открыть + lab9-p1.asm Стр. 37, Поз. 4
~/work/arch-pc/lab09

2 SECTION .data
3 msg db "Результат: ",0
4 fx: db 'f(x)= 15x - 9',0
5
6 SECTION .text
7 global _start
8 _start:
9 mov eax, fx
10 call sprintf
11 pop ecx
12 pop edx
13 sub ecx,1
14 mov esi, 0
15
16 next:
17 cmp ecx,0h
18 jz _end
19 pop eax
20 call atoi
21 call funk
22 add esi,eax
23
24 loop next
25
26 _end:
27 mov eax, msg
28 call sprintf
29 mov eax, esi
30 call iprintf
31 call quit
32
33 funk:
34 mov ebx,15
35 mul ebx
36 sub eax,9
37 ret
```

Рис. 2.16: Программа prog1.asm

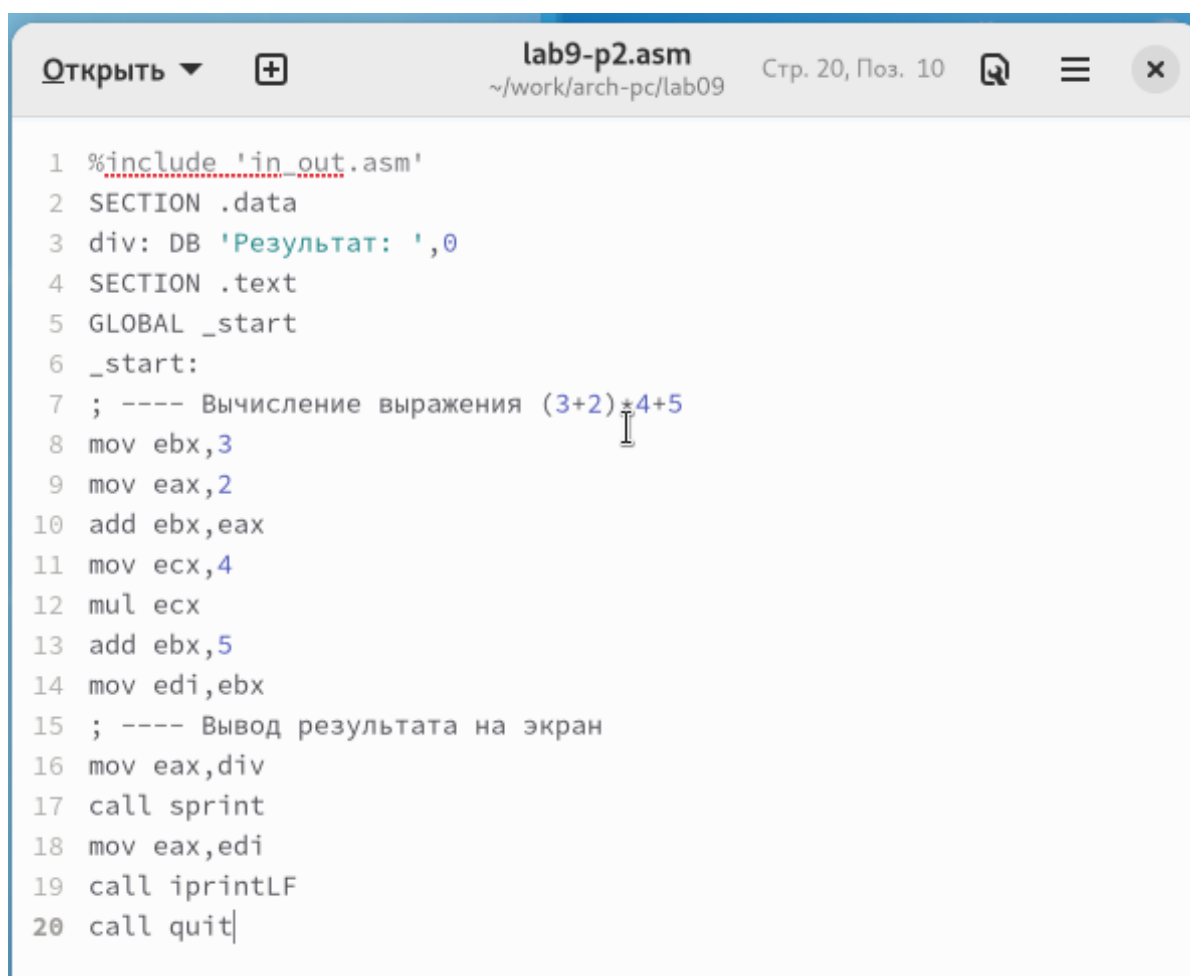
```

[adushaev@fedora lab09]$
[adushaev@fedora lab09]$ nasm -f elf lab9-p1.asm
[adushaev@fedora lab09]$ ld -m elf_i386 -o lab9-p1 lab9-p1.o
[adushaev@fedora lab09]$ ./lab9-p1
f(x)= 15x - 9
Результат: 0
[adushaev@fedora lab09]$ ./lab9-p1 1 2 3 4 5
f(x)= 15x - 9
Результат: 180
[adushaev@fedora lab09]$

```

Рис. 2.17: Запуск программы prog1.asm

7. Приведенный ниже листинг содержит программу для вычисления выражения $(3 + 2) * 4 + 5$. После запуска данная программа дает неверный результат. Проверил это с помощью отладчика GDB и анализа изменений значений регистров. Определил ошибку и внес соответствующие исправления



The screenshot shows a code editor window titled "lab9-p2.asm" with the path "~/work/arch-pc/lab09". The editor displays 20 lines of assembly code. Line 7 contains a comment in Russian: "; ---- Вычисление выражения (3+2)*4+5". The code uses various x86-64 instructions like mov, add, mul, and call. A red dotted line underlines the word "include" in line 1. A cursor is positioned at the end of line 20.

```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 SECTION .text
5 GLOBAL _start
6 _start:
7 ; ---- Вычисление выражения (3+2)*4+5
8 mov ebx,3
9 mov eax,2
10 add ebx,eax
11 mov ecx,4
12 mul ecx
13 add ebx,5
14 mov edi,ebx
15 ; ---- Вывод результата на экран
16 mov eax,div
17 call sprint
18 mov eax,edi
19 call iprintLF
20 call quit|
```

Рис. 2.18: Код с ошибкой


```

adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-p2
eax      0x8      8      ecx      0x4      4
edx      0x0      0      ebx      0x5      5
esp      0xffffd220 0xffffd220  ebp      0x0      0x0
esi      0x0      0      edi      0x0      0
eip      0x80490fb 0x80490fb <_start+  eflags    0x202    [ IF ]
cs       0x23     35     ss       0x2b     43
ds       0x2b     43     es       0x2b     43
fs       0x0      0      gs       0x0      0

B+ 0x80490e8 <_start>    mov     ebx,0x3
0x80490fb <_start+19>    add     eax,0x5
0x80490fe <_start+22>    add     edi,ebx
0x8049100 <_start+24>    mov     eax,0x804a000
0x8049105 <_start+29>    call    0x804900f <sprint>
> 0x804910a <_start+34>    mov     eax,edi
0x804910c <_start+36>    call    0x8049086 <iprintLF>
0x8049111 <_start+41>    call    0x80490db <quit>
                                rint>

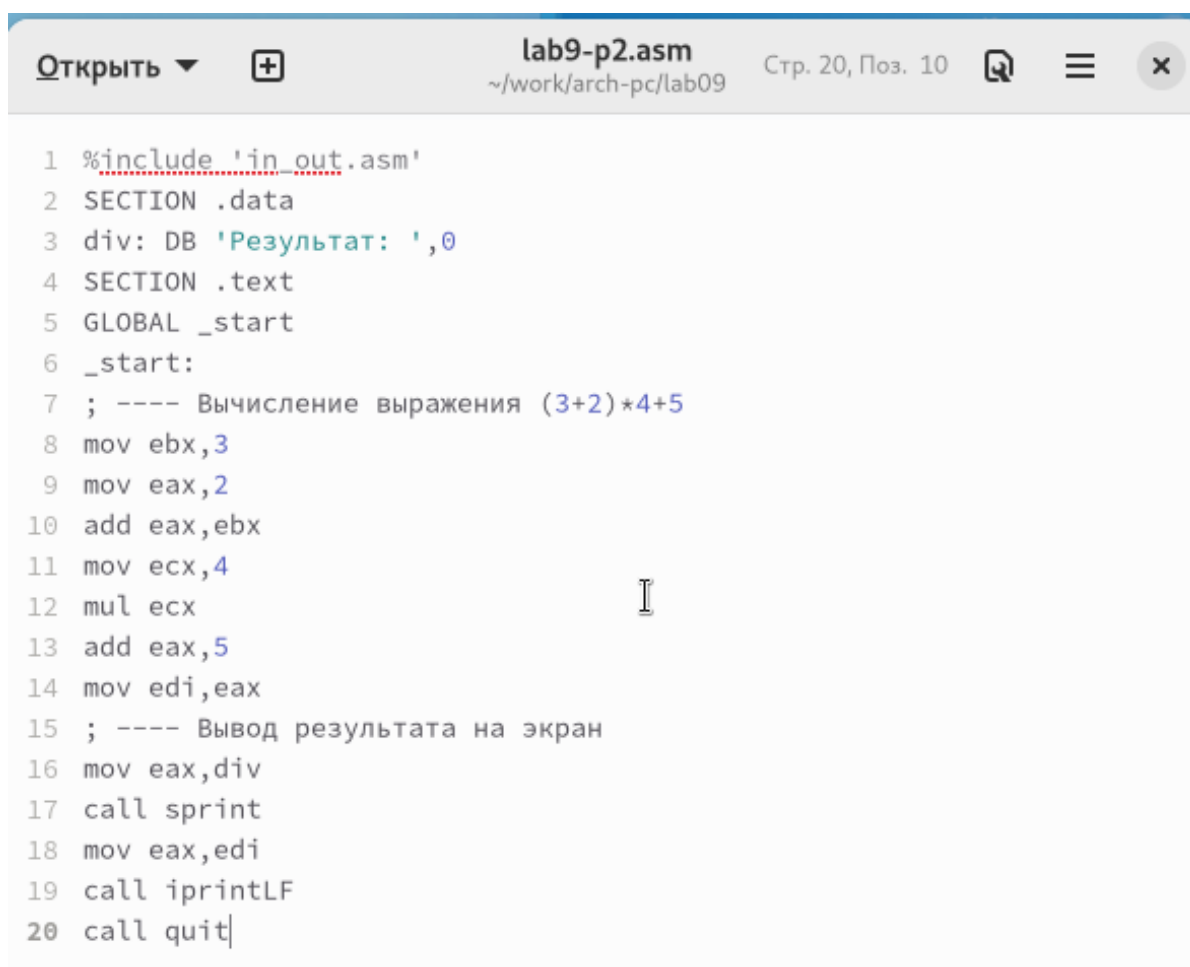
native process 3519 In: _start L13 PC: 0x80490fb
To makeNo process In: , add 'set debuginfod enabled off' to .gdbinit. L?? PC: ??
Breakpoint 1, _start () at lab9-p2.asm:8
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) cont
Continuing.
Результат: 10
[Inferior 1 (process 3519) exited normally]
(gdb)

```

Рис. 2.19: Отладка

Отмечу, что в исходной программе был перепутан порядок аргументов у инструкции `add`, а также в конце работы значение `ebx` передается в `edi` вместо `eax`.

Исправленный код программы



```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 SECTION .text
5 GLOBAL _start
6 _start:
7 ; ---- Вычисление выражения (3+2)*4+5
8 mov ebx,3
9 mov eax,2
10 add eax,ebx
11 mov ecx,4
12 mul ecx
13 add eax,5
14 mov edi,eax
15 ; ---- Вывод результата на экран
16 mov eax,div
17 call sprint
18 mov eax,edi
19 call iprintLF
20 call quit
```

Рис. 2.20: Код исправлен

```
adushaev@fedora:~/work/arch-pc/lab09 — gdb lab9-p2

eax      0x19      25      ecx      0x4       4
edx      0x0       0       ebx      0x3       3
esp      0xffffd220 0xffffd220 ebp      0x0       0x0
esi      0x0       0       edi      0x19      25
eip      0x8049100 0x8049100 <_start+
cs       0x23      35      ss       0x2b      43
ds       0x2b      43      es       0x2b      43
fs       0x0       0       gs       0x0       0

0x80490f4 <_start+12> mov     ecx,0x4
0x8049100 <_start+24> mov     eax,0x804a000rint>
0x804910a <_start+34> mov     eax,edi
0x804910c <_start+36> call    0x8049086 <iprintLF>
0x8049111 <_start+41> call    0x80490db <quit>

native process 3565 In: _start L16 PC: 0x8049100
Breakpoint process In: 9-p2.asm:8 L?? PC: ??
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) si
(gdb) cont
Continuing.
Результат: 25
[Inferior 1 (process 3565) exited normally]
(gdb)
```

Рис. 2.21: Проверка работы

3 Выводы

Освоили работу с подпрограммами и отладчиком.