

**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 9**

дисциплина: *Архитектура компьютера*

Студент: *Юсупова Амина Руслановна*

Группа: НКАбд-06-25

**МОСКВА**

2025 г.

# Содержание

1	Цель работы .....	2
2	Задание .....	2
3	Теоретическое введение.....	2
4	Выполнение лабораторной работы.....	3
4.1	Реализация подпрограмм в NASM .....	3
4.1.1	Отладка программ с помощью GDB.....	7
4.1.2	Добавление точек останова.....	11
4.1.3	Работа с данными программы в GDB .....	12
4.1.4	Обработка аргументов командной строки в GDB.....	16
4.2	Задание для самостоятельной работы .....	17
5	Выводы .....	26
6	Список литературы .....	26

## 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, отрабатывает, 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-таки 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

Создаю каталог lab09 для выполнения лабораторной работы №9, перехожу в него и создаю файл lab09-1.asm: (рис. 1).

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

Рисунок 1: Создание рабочего каталога и переход в него

Копирую в файл код из листинга 9.1 (рис. 2), компилирую и запускаю его, данная программа выполняет вычисление функции (рис. 3).

The screenshot shows a Windows Notepad window with the file name \*lab9-1.asm. The code is written in NASM assembly language. It includes sections for data and text, defines variables msg and result, and uses subroutines for input and output. The assembly code is as follows:

```
%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:

    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
|
_calcul:
    mov ebx,2
    mul ebx
    add eax,7

    mov [res],eax
    ret ; выход из подпрограммы
```

Рисунок 2: Копирование кода из листинга 9.1

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ nasm -f elf lab9-1.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-1 lab9-1.o
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ./lab9-1
Введите x: 10
2x+7=27
```

Рисунок 3: Запуск программы lab9-1.asm

Изменяю текст программы, добавив в нее подпрограмму (рис. 4).

Компилирую и запускаю программу lab9-1.asm, теперь она вычисляет значение функции для выражения  $f(g(x))$  (рис. 5).

The screenshot shows a Windows Notepad window with the file name \*lab9-1.asm. The code is as follows:

```
%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:  
|
```

Рисунок 4: Изменение кода программы lab9-1.asm

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ nasm -f elf lab9-1.asm  
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-1 lab9-1.o  
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ./lab9-1  
Введите x: 10  
2(3x-1)+7=65
```

Рисунок 5: Запуск изменённой программы lab9-1.asm

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
```

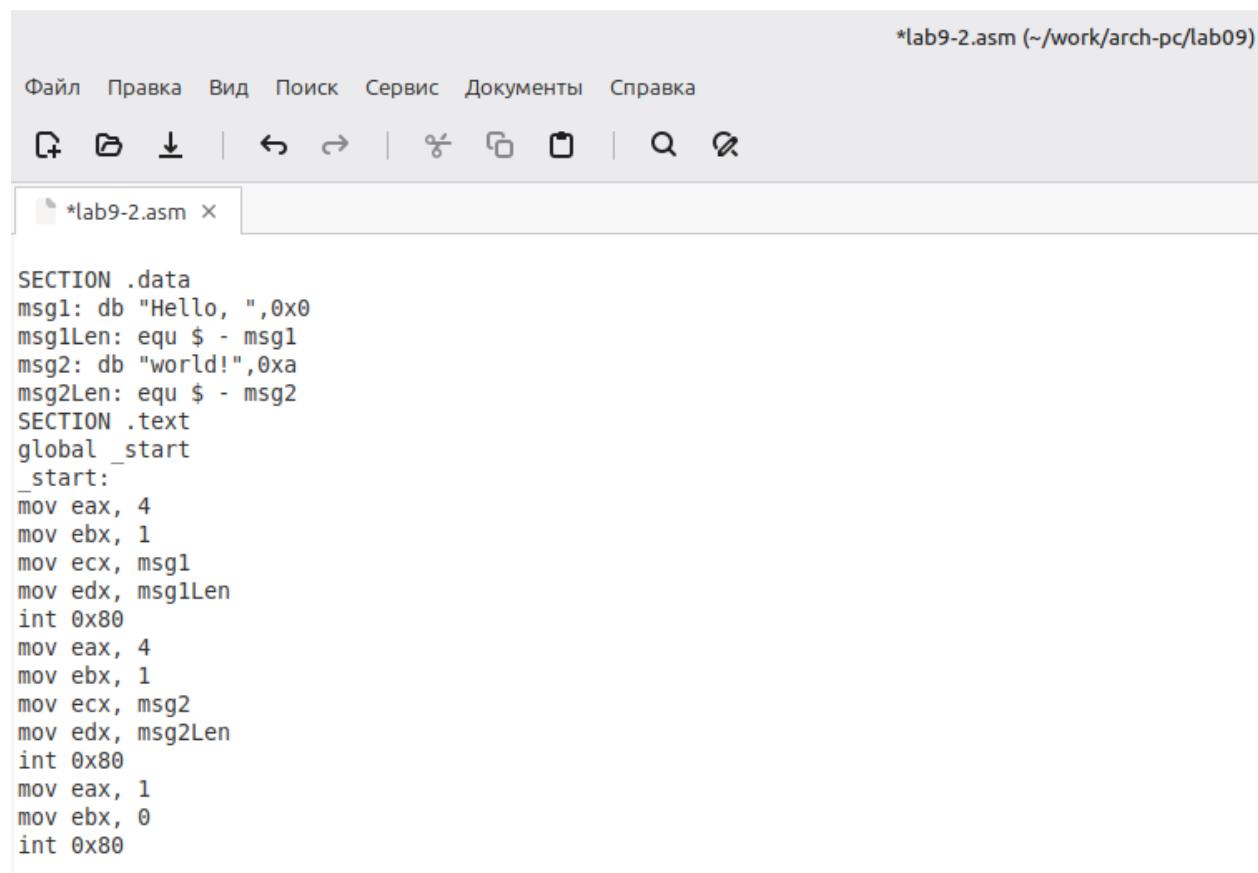
```
ret
```

#### 4.1.1 Отладка программ с помощью GDB

Создаю файл lab9-2.asm (рис.6), копирую программу листинга 9.2 (рис. 7), транслирую с созданием файла листинга и отладки, компоную и запускаю в отладчике (рис. 8).

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ touch lab9-2.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$
```

Рисунок 6: Создание файла lab9-2.asm



The screenshot shows a text editor window with the title bar "\*lab9-2.asm (~/work/arch-pc/lab09)". The menu bar includes "Файл", "Правка", "Вид", "Поиск", "Сервис", "Документы", and "Справка". Below the menu is a toolbar with icons for file operations like Open, Save, Copy, Paste, and Search. The main editor area contains the following assembly code:

```
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
```

Рисунок 7: Копирование кода из листинга 9.2 в файл lab9-2.asm

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
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-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) 
```

**Рисунок 8: Запуск программы lab9-2.asm**

Запустив программу командой run, я убедилась в том, что она работает исправно (рис. 9).

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
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-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/aryusupova/work/arch-pc/lab09/lab9-2
Hello, world!
[Inferior 1 (process 8997) exited normally]
(gdb) 
```

**Рисунок 9: Проверка программы отладчиком**

Для более подробного анализа программы добавляю брейкпоинт на метку `_start` и снова запускаю отладку (рис. 10).

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ nasm -f elf -g -l lab9-2.lst lab9-2.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-2 lab9-2.o
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ gdb lab9-2
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
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-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/aryusupova/work/arch-pc/lab09/lab9-2
Hello, world!
[Inferior 1 (process 8997) exited normally]
(gdb) break _start
Breakpoint 1 at 0x8049000: file lab9-2.asm, line 10.
(gdb) run
Starting program: /home/aryusupova/work/arch-pc/lab09/lab9-2

Breakpoint 1, _start () at lab9-2.asm:10
10      mov eax, 4
(gdb)
```

**Рисунок 10: Запуск отладчика с брейкпоинт**

Далее смотрю дисассимилированный код программы, перевожу на команд с синтаксисом Intel *амд топчик* (рис. 11).

Различия между синтаксисом ATT и Intel заключаются в порядке операндов (ATT - Операнд источника указан первым. Intel - Операнд назначения указан первым), их размере (ATT - размер operandов указывается явно с помощью суффиксов, непосредственные operandы предваряются символом \$; Intel - Размер operandов неявно определяется контекстом, как ax, eax, непосредственные operandы пишутся напрямую), именах регистров (ATT - имена регистров предваряются символом %, Intel - имена регистров пишутся без префиксов).

```
aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Файл Правка Вид Поиск Терминал Справка
(gdb) run
Starting program: /home/aryusupova/work/arch-pc/lab09/lab9-2

Breakpoint 1, _start () at lab9-2.asm:10
10      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 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) █
```

**Рисунок 11: Дисассимилирование программы**

Включаю режим псевдографики для более удобного анализа программы (рис. 12, рис. 13).

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Файл Правка Вид Поиск Терминал Справка

```
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
0x804902a < start+42>   int    0x80
0x804902c < start+44>   mov    eax,0x1
0x8049031 < start+49>   mov    ebx,0x0
0x8049036 < start+54>   int    0x80
0x8049038 add    BYTE PTR [eax],al
0x804903a add    BYTE PTR [eax],al
0x804903c add    BYTE PTR [eax],al
0x804903e add    BYTE PTR [eax],al
0x8049040 add    BYTE PTR [eax],al
0x8049042 add    BYTE PTR [eax],al
0x8049044 add    BYTE PTR [eax],al
0x8049046 add    BYTE PTR [eax],al
0x8049048 add    BYTE PTR [eax],al
0x804904a add    BYTE PTR [eax],al
```

native process 9090 (asm) In: \_start  
(gdb) layout regs

Рисунок 12: Включение режима псевдографики

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Файл Правка Вид Поиск Терминал Справка

Register group: general

регистр	значение	регистр	значение	регистр	значение
eax	0x0	0		ecx	0x0
edx	0x0			ebx	0x0
esp	0xfffffd080	0xfffffd080		ebp	0x0
esi	0x0	0		edi	0x0
eip	0x8049000	0x8049000 < start>		eflags	0x202 [ IF ]
cs	0x23	35		ss	0x2b
ds	0x2b	43		es	0x2b
fs	0x0	0		gs	0x0

```
B+>0x8049000 < start>    mov    eax,0x4
0x8049005 < start+5>    mov    ebx,0x1
0x804900a < start+10>   mov    ecx,0x804a000
0x804900f < start+15>   mov    edx,0x8
0x8049014 < start+20>   int    0x80
0x8049016 < start+22>   mov    eax,0x4
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 9090 (asm) In: start  
(gdb) layout regs  
(gdb)

Рисунок 13: Режим псевдографики

#### 4.1.2 Добавление точек останова

Проверяю в режиме псевдографики, что брейкпойнт сохранился (рис. 14).

```

Registers
eax          0x0          0
edx          0x0          0
esp          0xfffffd080  0xfffffd080
esi          0x0          0
eip          0x8049000  0x8049000 <_start>
cs           0x23         35
ds           0x2b         43
fs           0x0          0
ecx          0x0          0
ebx          0x0          0
ebp          0x0          0x0
edi          0x0          0
eflags       0x202      [ IF ]
ss            0x2b        43
es            0x2b        43
gs            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
0x804902a < start+42>   int    0x80
0x804902c < start+44>   mov    eax,0x1

native process 9090 (asm) In: start
(gdb) layout regs
(gdb) info breakpoints
Num  Type      Disp Enb Address  What
1   breakpoint  keep y  0x08049000 lab9-2.asm:10
breakpoint already hit 1 time
(gdb) b *0x8049031
Breakpoint 2 at 0x8049031: file lab9-2.asm, line 21.

```

**Рисунок 14: Список брейкпоинтов**

Установим еще одну точку останова по адресу инструкции (с помощью команды (gdb) break \*) и посмотрим информацию о всех установленных точках останова ( с помощью команды i b) (рис. 15).

```

Registers
eax          0x0          0
edx          0x0          0
esp          0xfffffd080  0xfffffd080
esi          0x0          0
eip          0x8049000  0x8049000 <_start>
cs           0x23         35
ds           0x2b         43
fs           0x0          0
ecx          0x0          0
ebx          0x0          0
ebp          0x0          0x0
edi          0x0          0
eflags       0x202      [ IF ]
ss            0x2b        43
es            0x2b        43
gs            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
0x804902a < start+42>   int    0x80
0x804902c < start+44>   mov    eax,0x1

native process 9090 (asm) In: start
(gdb) layout regs
(gdb) info breakpoints
Num  Type      Disp Enb Address  What
1   breakpoint  keep y  0x08049000 lab9-2.asm:10
breakpoint already hit 1 time
(gdb) b *0x8049031
Breakpoint 2 at 0x8049031: file lab9-2.asm, line 21.
(gdb) i b
Num  Type      Disp Enb Address  What
1   breakpoint  keep y  0x08049000 lab9-2.asm:10
breakpoint already hit 1 time
2   breakpoint  keep y  0x08049031 lab9-2.asm:21
(gdb)

```

**Рисунок 15: Добавление второй точки останова**

#### 4.1.3 Работа с данными программы в GDB

Просматриваю содержимое регистров командой info registers (рис. 16).

```

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x0          0
edx      0x0          0
esp     0xfffffd070  0xfffffd070
esi      0x0          0
eip     0x8049000  0x8049000 <_start>
cs       0x23         35
ds       0x2b         43
fs       0x0          0
ecx      0x0          0
ebx      0x0          0
ebp      0x0          0x0
edi      0x0          0
eflags   0x202        [ IF ]
ss       0x2b         43
es       0x2b         43
gs       0x0          0

--lab9-2.asm--
1
2 SECTION .data
3 msg1: db "Hello, ",0x0
4 msg1Len: equ $ - msg1
5 msg2: db "world!",0xa
6 msg2Len: equ $ - msg2
7 SECTION .text
8 global _start
9 start:
B+> 10 mov eax, 4
11 mov ebx, 1
12 mov ecx, msg1

native process 16104 (src) In:  start
eax      0x0          0
ecx      0x0          0
edx      0x0          0
ebx      0x0          0
esp     0xfffffd070  0xfffffd070
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
gs       0x0          0
--Type <RET> for more, q to quit, c to continue without paging--

```

**Рисунок 16: Просмотр содержимого регистров**

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

```

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x0          0
edx      0x0          0
esp     0xfffffd070  0xfffffd070
esi      0x0          0
eip     0x8049000  0x8049000 <_start>
cs       0x23         35
ds       0x2b         43
fs       0x0          0
ecx      0x0          0
ebx      0x0          0
ebp      0x0          0x0
edi      0x0          0
eflags   0x292        [ IF ]
ss       0x2b         43
es       0x2b         43
gs       0x0          0

--lab9-2.asm--
1
2 SECTION .data
3 msg1: db "Hello, ",0x0
4 msg1Len: equ $ - msg1
5 msg2: db "world!",0xa
6 msg2Len: equ $ - msg2
7 SECTION .text
8 global _start
9 start:
B+> 10 mov eax, 4
11 mov ebx, 1

native process 16104 (src) In:  start
--Type <RET> for more, q to quit, c to continue without paging--
ds       0x2b         43
es       0x2b         43
fs       0x0          0
gs       0x0          0
(gdb) x/1sb &msg1
0x804a000 <msg1>:      "Hello, "
(gdb) x/1sb 0x804a008
0x804a008 <msg2>:      "world!\n\034"
(gdb) 

```

**Рисунок 17: Просмотр содержимого переменных двумя способами**

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

```
aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x0          0           ecx      0x0          0
edx      0x0          0           ebx      0x0          0
esp     0xfffffd070  0xfffffd070  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

lab9-2.asm-
1
2 SECTION .data
3 msg1: db "Hello, ",0x0
4 msg1Len: equ $ - msg1
5 msg2: db "world!",0xa
6 msg2Len: equ $ - msg2
7 SECTION .text
8 global _start
9 start:
B+> 10 mov eax, 4
11 mov ebx, 1

native process 16104 (src) In:  start
es      0x2b         43
fs      0x0          0
gs      0x0          0
(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}&msg2='x'
(gdb) x/1sb &msg2
0x804a008 <msg2>:      "xorl!\\n\\034"
(gdb) 
```

Рисунок 18: Изменение содержимого переменных двумя способами

Вывожу в различных форматах значение регистра edx (рис. 19). Выводит нули, не совсем понимаю по какой причине.

```
aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x0          0           ecx      0x0          0
edx      0x0          0           ebx      0x0          0
esp     0xfffffd070  0xfffffd070  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

lab9-2.asm-
1
2 SECTION data
3 msg1: db "Hello, ",0x0
4 msg1Len: equ $ - msg1
5 msg2: db "world!",0xa
6 msg2Len: equ $ - msg2
7 SECTION text
8 global _start
9 start:
B+> 10 mov eax, 4
11 mov ebx, 1

native process 16104 (src) In:  start
(gdb) set {char}&msg2='x'
(gdb) x/1sb &msg2
0x804a008 <msg2>:      "xorl!\\n\\034"
(gdb) p/t $ecx
$1 = 0
(gdb) print /t $ecx
$2 = 0
(gdb) p /s $edx
$3 = 0
(gdb) p/t $edx
$4 = 0
(gdb) p/x $edx
$5 = 0x0
(gdb) 
```

Рисунок 19: Просмотр значения регистра разными представлениями

С помощью команды set меняю содержимое регистра ebx (рис. 20).

```
aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09
Register group: general
eax      0x0          0
edx      0x0          0
esp     0xfffffd070  0xfffffd070
esi      0x0          0
eip     0x8049000  0x8049000 <_start>
cs       0x23         35
ds       0x2b         43
fs       0x0          0
ecx      0x0          0
ebx      0x2          2
ebp      0x0          0x0
edi      0x0          0
eflags   0x202      [ IF ]
ss       0x2b         43
es       0x2b         43
gs       0x0          0

--lab9-2.asm--
1
2 SECTION .data
3 msg1: db "Hello, ",0x0
4 msg1Len: equ $ - msg1
5 msg2: db "world!",0xa
6 msg2Len: equ $ - msg2
7 SECTION .text
8 global _start
9 _start:
B+> 10 mov eax, 4
11 mov ebx, 1

native process 16104 (src) In: _start
$3 = 0
(gdb) p/t $edx
$4 = 0
(gdb) p/x $edx
$5 = 0x0
(gdb) set $ebx='2'
(gdb) p/s
$6 = 0
(gdb) p/s $ebx
$7 = 50
(gdb) set $ebx=2
(gdb) p/s $ebx
$8 = 2
(gdb) 
```

Рисунок 20: Пример использования команды set

p/s \$ebx: Интерпретирует значение регистра как указатель на строку (адрес в памяти) и выводит строку, начиная с этого адреса, до нулевого терминатора.

Разница в выводе (50 vs 2) обусловлена разницей между ASCII-кодом символа '2' и числом 2.

Завершаю выполнение программы с помощью команды continue (сокращенно c) и выхожу из GDB с помощью команды quit (сокращенно q) (рис. 21).

The screenshot shows a terminal window with several panes. The top pane displays assembly code for `lab9-2.asm`. The second pane shows the state of registers (eax, ecx, edx, etc.) and memory. The third pane shows a stack dump. The bottom pane is a GDB session where the program has been run and is at a breakpoint. The command `disas` is shown, followed by the assembly code. Then, the command `break _start` is issued, and the program starts executing. The output shows "hello, world!". Finally, the command `quit` is entered to exit GDB.

```

aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ disas
      1    .text:00401000 17 mov    ecx, msg
      2    .text:00401004 18 mov    edx, msgLen
      3    .text:00401008 19 int   0x80
      4    .text:0040100C 20 mov    eax, 1
B+>  5    .text:00401010 21 mov    ebx, 0
      6    .text:00401014 22 int   0x80

B+>  1    ,4
      7    .text:00401018 23 mov    eax, 0
      8    .text:0040101C 24 mov    ebx, 0
      9    .text:00401020 25 mov    ecx, 0
      A    .text:00401024 26 mov    edx, 0
      B    .text:00401028 27 mov    eax, 0
      C    .text:0040102C 28 mov    ebx, 0
      D    .text:00401030 29 mov    ecx, 0
      E    .text:00401034 30 mov    edx, 0
      F    .text:00401038 31 mov    eax, 0
      10   .text:0040103C 32 mov    ebx, 0
      11   .text:00401040 33 mov    ecx, 0
      12   .text:00401044 34 mov    edx, 0
      13   .text:00401048 35 mov    eax, 0
      14   .text:0040104C 36 mov    ebx, 0
      15   .text:00401050 37 mov    ecx, 0
      16   .text:00401054 38 mov    edx, 0
      17   .text:00401058 39 mov    eax, 0
      18   .text:0040105C 40 mov    ebx, 0
      19   .text:00401060 41 mov    ecx, 0
      20   .text:00401064 42 mov    edx, 0
      21   .text:00401068 43 mov    eax, 0
      22   .text:0040106C 44 mov    ebx, 0
      23   .text:00401070 45 mov    ecx, 0
      24   .text:00401074 46 mov    edx, 0
      25   .text:00401078 47 mov    eax, 0
      26   .text:0040107C 48 mov    ebx, 0
      27   .text:00401080 49 mov    ecx, 0
      28   .text:00401084 50 mov    edx, 0
      29   .text:00401088 51 mov    eax, 0
      30   .text:0040108C 52 mov    ebx, 0
      31   .text:00401090 53 mov    ecx, 0
      32   .text:00401094 54 mov    edx, 0
      33   .text:00401098 55 mov    eax, 0
      34   .text:0040109C 56 mov    ebx, 0
      35   .text:004010A0 57 mov    ecx, 0
      36   .text:004010A4 58 mov    edx, 0
      37   .text:004010A8 59 mov    eax, 0
      38   .text:004010AC 60 mov    ebx, 0
      39   .text:004010B0 61 mov    ecx, 0
      40   .text:004010B4 62 mov    edx, 0
      41   .text:004010B8 63 mov    eax, 0
      42   .text:004010BC 64 mov    ebx, 0
      43   .text:004010C0 65 mov    ecx, 0
      44   .text:004010C4 66 mov    edx, 0
      45   .text:004010C8 67 mov    eax, 0
      46   .text:004010CC 68 mov    ebx, 0
      47   .text:004010D0 69 mov    ecx, 0
      48   .text:004010D4 70 mov    edx, 0
      49   .text:004010D8 71 mov    eax, 0
      50   .text:004010DC 72 mov    ebx, 0
      51   .text:004010E0 73 mov    ecx, 0
      52   .text:004010E4 74 mov    edx, 0
      53   .text:004010E8 75 mov    eax, 0
      54   .text:004010EC 76 mov    ebx, 0
      55   .text:004010F0 77 mov    ecx, 0
      56   .text:004010F4 78 mov    edx, 0
      57   .text:004010F8 79 mov    eax, 0
      58   .text:004010FC 80 mov    ebx, 0
      59   .text:00401100 81 mov    ecx, 0
      60   .text:00401104 82 mov    edx, 0
      61   .text:00401108 83 mov    eax, 0
      62   .text:0040110C 84 mov    ebx, 0
      63   .text:00401110 85 mov    ecx, 0
      64   .text:00401114 86 mov    edx, 0
      65   .text:00401118 87 mov    eax, 0
      66   .text:0040111C 88 mov    ebx, 0
      67   .text:00401120 89 mov    ecx, 0
      68   .text:00401124 90 mov    edx, 0
      69   .text:00401128 91 mov    eax, 0
      70   .text:0040112C 92 mov    ebx, 0
      71   .text:00401130 93 mov    ecx, 0
      72   .text:00401134 94 mov    edx, 0
      73   .text:00401138 95 mov    eax, 0
      74   .text:0040113C 96 mov    ebx, 0
      75   .text:00401140 97 mov    ecx, 0
      76   .text:00401144 98 mov    edx, 0
      77   .text:00401148 99 mov    eax, 0
      78   .text:0040114C 100 mov   ebx, 0
      79   .text:00401150 101 mov   ecx, 0
      80   .text:00401154 102 mov   edx, 0
      81   .text:00401158 103 mov   eax, 0
      82   .text:0040115C 104 mov   ebx, 0
      83   .text:00401160 105 mov   ecx, 0
      84   .text:00401164 106 mov   edx, 0
      85   .text:00401168 107 mov   eax, 0
      86   .text:0040116C 108 mov   ebx, 0
      87   .text:00401170 109 mov   ecx, 0
      88   .text:00401174 110 mov   edx, 0
      89   .text:00401178 111 mov   eax, 0
      90   .text:0040117C 112 mov   ebx, 0
      91   .text:00401180 113 mov   ecx, 0
      92   .text:00401184 114 mov   edx, 0
      93   .text:00401188 115 mov   eax, 0
      94   .text:0040118C 116 mov   ebx, 0
      95   .text:00401190 117 mov   ecx, 0
      96   .text:00401194 118 mov   edx, 0
      97   .text:00401198 119 mov   eax, 0
      98   .text:0040119C 120 mov   ebx, 0
      99   .text:004011A0 121 mov   ecx, 0
      100  .text:004011A4 122 mov   edx, 0
      101  .text:004011A8 123 mov   eax, 0
      102  .text:004011AC 124 mov   ebx, 0
      103  .text:004011B0 125 mov   ecx, 0
      104  .text:004011B4 126 mov   edx, 0
      105  .text:004011B8 127 mov   eax, 0
      106  .text:004011BC 128 mov   ebx, 0
      107  .text:004011C0 129 mov   ecx, 0
      108  .text:004011C4 130 mov   edx, 0
      109  .text:004011C8 131 mov   eax, 0
      110  .text:004011CC 132 mov   ebx, 0
      111  .text:004011D0 133 mov   ecx, 0
      112  .text:004011D4 134 mov   edx, 0
      113  .text:004011D8 135 mov   eax, 0
      114  .text:004011DC 136 mov   ebx, 0
      115  .text:004011E0 137 mov   ecx, 0
      116  .text:004011E4 138 mov   edx, 0
      117  .text:004011E8 139 mov   eax, 0
      118  .text:004011EC 140 mov   ebx, 0
      119  .text:004011F0 141 mov   ecx, 0
      120  .text:004011F4 142 mov   edx, 0
      121  .text:004011F8 143 mov   eax, 0
      122  .text:004011FC 144 mov   ebx, 0
      123  .text:004011E0 145 mov   ecx, 0
      124  .text:004011E4 146 mov   edx, 0
      125  .text:004011E8 147 mov   eax, 0
      126  .text:004011EC 148 mov   ebx, 0
      127  .text:004011F0 149 mov   ecx, 0
      128  .text:004011F4 150 mov   edx, 0
      129  .text:004011F8 151 mov   eax, 0
      130  .text:004011FC 152 mov   ebx, 0
      131  .text:004011E0 153 mov   ecx, 0
      132  .text:004011E4 154 mov   edx, 0
      133  .text:004011E8 155 mov   eax, 0
      134  .text:004011EC 156 mov   ebx, 0
      135  .text:004011F0 157 mov   ecx, 0
      136  .text:004011F4 158 mov   edx, 0
      137  .text:004011F8 159 mov   eax, 0
      138  .text:004011FC 160 mov   ebx, 0
      139  .text:004011E0 161 mov   ecx, 0
      140  .text:004011E4 162 mov   edx, 0
      141  .text:004011E8 163 mov   eax, 0
      142  .text:004011EC 164 mov   ebx, 0
      143  .text:004011F0 165 mov   ecx, 0
      144  .text:004011F4 166 mov   edx, 0
      145  .text:004011F8 167 mov   eax, 0
      146  .text:004011FC 168 mov   ebx, 0
      147  .text:004011E0 169 mov   ecx, 0
      148  .text:004011E4 170 mov   edx, 0
      149  .text:004011E8 171 mov   eax, 0
      150  .text:004011EC 172 mov   ebx, 0
      151  .text:004011F0 173 mov   ecx, 0
      152  .text:004011F4 174 mov   edx, 0
      153  .text:004011F8 175 mov   eax, 0
      154  .text:004011FC 176 mov   ebx, 0
      155  .text:004011E0 177 mov   ecx, 0
      156  .text:004011E4 178 mov   edx, 0
      157  .text:004011E8 179 mov   eax, 0
      158  .text:004011EC 180 mov   ebx, 0
      159  .text:004011F0 181 mov   ecx, 0
      160  .text:004011F4 182 mov   edx, 0
      161  .text:004011F8 183 mov   eax, 0
      162  .text:004011FC 184 mov   ebx, 0
      163  .text:004011E0 185 mov   ecx, 0
      164  .text:004011E4 186 mov   edx, 0
      165  .text:004011E8 187 mov   eax, 0
      166  .text:004011EC 188 mov   ebx, 0
      167  .text:004011F0 189 mov   ecx, 0
      168  .text:004011F4 190 mov   edx, 0
      169  .text:004011F8 191 mov   eax, 0
      170  .text:004011FC 192 mov   ebx, 0
      171  .text:004011E0 193 mov   ecx, 0
      172  .text:004011E4 194 mov   edx, 0
      173  .text:004011E8 195 mov   eax, 0
      174  .text:004011EC 196 mov   ebx, 0
      175  .text:004011F0 197 mov   ecx, 0
      176  .text:004011F4 198 mov   edx, 0
      177  .text:004011F8 199 mov   eax, 0
      178  .text:004011FC 200 mov   ebx, 0
      179  .text:004011E0 201 mov   ecx, 0
      180  .text:004011E4 202 mov   edx, 0
      181  .text:004011E8 203 mov   eax, 0
      182  .text:004011EC 204 mov   ebx, 0
      183  .text:004011F0 205 mov   ecx, 0
      184  .text:004011F4 206 mov   edx, 0
      185  .text:004011F8 207 mov   eax, 0
      186  .text:004011FC 208 mov   ebx, 0
      187  .text:004011E0 209 mov   ecx, 0
      188  .text:004011E4 210 mov   edx, 0
      189  .text:004011E8 211 mov   eax, 0
      190  .text:004011EC 212 mov   ebx, 0
      191  .text:004011F0 213 mov   ecx, 0
      192  .text:004011F4 214 mov   edx, 0
      193  .text:004011F8 215 mov   eax, 0
      194  .text:004011FC 216 mov   ebx, 0
      195  .text:004011E0 217 mov   ecx, 0
      196  .text:004011E4 218 mov   edx, 0
      197  .text:004011E8 219 mov   eax, 0
      198  .text:004011EC 220 mov   ebx, 0
      199  .text:004011F0 221 mov   ecx, 0
      200  .text:004011F4 222 mov   edx, 0
      201  .text:004011F8 223 mov   eax, 0
      202  .text:004011FC 224 mov   ebx, 0
      203  .text:004011E0 225 mov   ecx, 0
      204  .text:004011E4 226 mov   edx, 0
      205  .text:004011E8 227 mov   eax, 0
      206  .text:004011EC 228 mov   ebx, 0
      207  .text:004011F0 229 mov   ecx, 0
      208  .text:004011F4 230 mov   edx, 0
      209  .text:004011F8 231 mov   eax, 0
      210  .text:004011FC 232 mov   ebx, 0
      211  .text:004011E0 233 mov   ecx, 0
      212  .text:004011E4 234 mov   edx, 0
      213  .text:004011E8 235 mov   eax, 0
      214  .text:004011EC 236 mov   ebx, 0
      215  .text:004011F0 237 mov   ecx, 0
      216  .text:004011F4 238 mov   edx, 0
      217  .text:004011F8 239 mov   eax, 0
      218  .text:004011FC 240 mov   ebx, 0
      219  .text:004011E0 241 mov   ecx, 0
      220  .text:004011E4 242 mov   edx, 0
      221  .text:004011E8 243 mov   eax, 0
      222  .text:004011EC 244 mov   ebx, 0
      223  .text:004011F0 245 mov   ecx, 0
      224  .text:004011F4 246 mov   edx, 0
      225  .text:004011F8 247 mov   eax, 0
      226  .text:004011FC 248 mov   ebx, 0
      227  .text:004011E0 249 mov   ecx, 0
      228  .text:004011E4 250 mov   edx, 0
      229  .text:004011E8 251 mov   eax, 0
      230  .text:004011EC 252 mov   ebx, 0
      231  .text:004011F0 253 mov   ecx, 0
      232  .text:004011F4 254 mov   edx, 0
      233  .text:004011F8 255 mov   eax, 0
      234  .text:004011FC 256 mov   ebx, 0
      235  .text:004011E0 257 mov   ecx, 0
      236  .text:004011E4 258 mov   edx, 0
      237  .text:004011E8 259 mov   eax, 0
      238  .text:004011EC 260 mov   ebx, 0
      239  .text:004011F0 261 mov   ecx, 0
      240  .text:004011F4 262 mov   edx, 0
      241  .text:004011F8 263 mov   eax, 0
      242  .text:004011FC 264 mov   ebx, 0
      243  .text:004011E0 265 mov   ecx, 0
      244  .text:004011E4 266 mov   edx, 0
      245  .text:004011E8 267 mov   eax, 0
      246  .text:004011EC 268 mov   ebx, 0
      247  .text:004011F0 269 mov   ecx, 0
      248  .text:004011F4 270 mov   edx, 0
      249  .text:004011F8 271 mov   eax, 0
      250  .text:004011FC 272 mov   ebx, 0
      251  .text:004011E0 273 mov   ecx, 0
      252  .text:004011E4 274 mov   edx, 0
      253  .text:004011E8 275 mov   eax, 0
      254  .text:004011EC 276 mov   ebx, 0
      255  .text:004011F0 277 mov   ecx, 0
      256  .text:004011F4 278 mov   edx, 0
      257  .text:004011F8 279 mov   eax, 0
      258  .text:004011FC 280 mov   ebx, 0
      259  .text:004011E0 281 mov   ecx, 0
      260  .text:004011E4 282 mov   edx, 0
      261  .text:004011E8 283 mov   eax, 0
      262  .text:004011EC 284 mov   ebx, 0
      263  .text:004011F0 285 mov   ecx, 0
      264  .text:004011F4 286 mov   edx, 0
      265  .text:004011F8 287 mov   eax, 0
      266  .text:004011FC 288 mov   ebx, 0
      267  .text:004011E0 289 mov   ecx, 0
      268  .text:004011E4 290 mov   edx, 0
      269  .text:004011E8 291 mov   eax, 0
      270  .text:004011EC 292 mov   ebx, 0
      271  .text:004011F0 293 mov   ecx, 0
      272  .text:004011F4 294 mov   edx, 0
      273  .text:004011F8 295 mov   eax, 0
      274  .text:004011FC 296 mov   ebx, 0
      275  .text:004011E0 297 mov   ecx, 0
      276  .text:004011E4 298 mov   edx, 0
      277  .text:004011E8 299 mov   eax, 0
      278  .text:004011EC 300 mov   ebx, 0
      279  .text:004011F0 301 mov   ecx, 0
      280  .text:004011F4 302 mov   edx, 0
      281  .text:004011F8 303 mov   eax, 0
      282  .text:004011FC 304 mov   ebx, 0
      283  .text:004011E0 305 mov   ecx, 0
      284  .text:004011E4 306 mov   edx, 0
      285  .text:004011E8 307 mov   eax, 0
      286  .text:004011EC 308 mov   ebx, 0
      287  .text:004011F0 309 mov   ecx, 0
      288  .text:004011F4 310 mov   edx, 0
      289  .text:004011F8 311 mov   eax, 0
      290  .text:004011FC 312 mov   ebx, 0
      291  .text:004011E0 313 mov   ecx, 0
      292  .text:004011E4 314 mov   edx, 0
      293  .text:004011E8 315 mov   eax, 0
      294  .text:004011EC 316 mov   ebx, 0
      295  .text:004011F0 317 mov   ecx, 0
      296  .text:004011F4 318 mov   edx, 0
      297  .text:004011F8 319 mov   eax, 0
      298  .text:004011FC 320 mov   ebx, 0
      299  .text:004011E0 321 mov   ecx, 0
      300  .text:004011E4 322 mov   edx, 0
      301  .text:004011E8 323 mov   eax, 0
      302  .text:004011EC 324 mov   ebx, 0
      303  .text:004011F0 325 mov   ecx, 0
      304  .text:004011F4 326 mov   edx, 0
      305  .text:004011F8 327 mov   eax, 0
      306  .text:004011FC 328 mov   ebx, 0
      307  .text:004011E0 329 mov   ecx, 0
      308  .text:004011E4 330 mov   edx, 0
      309  .text:004011E8 331 mov   eax, 0
      310  .text:004011EC 332 mov   ebx, 0
      311  .text:004011F0 333 mov   ecx, 0
      312  .text:004011F4 334 mov   edx, 0
      313  .text:004011F8 335 mov   eax, 0
      314  .text:004011FC 336 mov   ebx, 0
      315  .text:004011E0 337 mov   ecx, 0
      316  .text:004011E4 338 mov   edx, 0
      317  .text:004011E8 339 mov   eax, 0
      318  .text:004011EC 340 mov   ebx, 0
      319  .text:004011F0 341 mov   ecx, 0
      320  .text:004011F4 342 mov   edx, 0
      321  .text:004011F8 343 mov   eax, 0
      322  .text:004011FC 344 mov   ebx, 0
      323  .text:004011E0 345 mov   ecx, 0
      324  .text:004011E4 346 mov   edx, 0
      325  .text:004011E8 347 mov   eax, 0
      326  .text:004011EC 348 mov   ebx, 0
      327  .text:004011F0 349 mov   ecx, 0
      328  .text:004011F4 350 mov   edx, 0
      329  .text:004011F8 351 mov   eax, 0
      330  .text:004011FC 352 mov   ebx, 0
      331  .text:004011E0 353 mov   ecx, 0
      332  .text:004011E4 354 mov   edx, 0
      333  .text:004011E8 355 mov   eax, 0
      334  .text:004011EC 356 mov   ebx, 0
      335  .text:004011F0 357 mov   ecx, 0
      336  .text:004011F4 358 mov   edx, 0
      337  .text:004011F8 359 mov   eax, 0
      338  .text:004011FC 360 mov   ebx, 0
      339  .text:004011E0 361 mov   ecx, 0
      340  .text:004011E4 362 mov   edx, 0
      341  .text:004011E8 363 mov   eax, 0
      342  .text:004011EC 364 mov   ebx, 0
      343  .text:004011F0 365 mov   ecx, 0
      344  .text:004011F4 366 mov   edx, 0
      345  .text:004011F8 367 mov   eax, 0
      346  .text:004011FC 368 mov   ebx, 0
      347  .text:004011E0 369 mov   ecx, 0
      348  .text:004011E4 370 mov   edx, 0
      349  .text:004011E8 371 mov   eax, 0
      350  .text:004011EC 372 mov   ebx, 0
      351  .text:004011F0 373 mov   ecx, 0
      352  .text:004011F4 374 mov   edx, 0
      353  .text:004011F8 375 mov   eax, 0
      354  .text:004011FC 376 mov   ebx, 0
      355  .text:004011E0 377 mov   ecx, 0
      356  .text:004011E4 378 mov   edx, 0
      357  .text:004011E8 379 mov   eax, 0
      358  .text:004011EC 380 mov   ebx, 0
      359  .text:004011F0 381 mov   ecx, 0
      360  .text:004011F4 382 mov   edx, 0
      361  .text:004011F8 383 mov   eax, 0
      362  .text:004011FC 384 mov   ebx, 0
      363  .text:004011E0 385 mov   ecx, 0
      364  .text:004011E4 386 mov   edx, 0
      365  .text:004011E8 387 mov   eax, 0
      366  .text:004011EC 388 mov   ebx, 0
      367  .text:004011F0 389 mov   ecx, 0
      368  .text:004011F4 390 mov   edx, 0
      369  .text:004011F8 391 mov   eax, 0
      370  .text:004011FC 392 mov   ebx, 0
      371  .text:004011E0 393 mov   ecx, 0
      372  .text:004011E4 394 mov   edx, 0
      373  .text:004011E8 395 mov   eax, 0
      374  .text:004011EC 396 mov   ebx, 0
      375  .text:004011F0 397 mov   ecx, 0
      376  .text:004011F4 398 mov   edx, 0
      377  .text:004011F8 399 mov   eax, 0
      378  .text:004011FC 400 mov   ebx, 0
      379  .text:004011E0 401 mov   ecx, 0
      380  .text:004011E4 402 mov   edx, 0
      381  .text:004011E8 403 mov   eax, 0
      382  .text:004011EC 404 mov   ebx, 0
      383  .text:004011F0 405 mov   ecx, 0
      384  .text:004011F4 406 mov   edx, 0
      385  .text:004011F8 407 mov   eax, 0
      386  .text:004011FC 408 mov   ebx, 0
      387  .text:004011E0 409 mov   ecx, 0
      388  .text:004011E4 410 mov   edx, 0
      389  .text:004011E8 411 mov   eax, 0
      390  .text:004011EC 412 mov   ebx, 0
      391  .text:004011F0 413 mov   ecx, 0
      392  .text:004011F4 414 mov   edx, 0
      393  .text:004011F8 415 mov   eax, 0
      394  .text:004011FC 416 mov   ebx, 0
      395  .text:004011E0 417 mov  
```

Исследую расположение аргументов командной строки в стеке после запуска программы с помощью gdb. Для начала установлю точку останова перед первой инструкцией в программе и запущу ее.

```
(gdb) b start
Breakpoint 1 at 0x80490e8: file lab9-3.asm, line 5.
(gdb) run
Starting program: /home/aryusupova/work/arch-pc/lab09/lab9-3 аргумент1 аргумент2 аргумент3

Breakpoint 1, start () at lab9-3.asm:5
5      pop  ecx
(gdb) █
```

**Рисунок 25: Установка точки останова и её запуск**

В 32-битных системах размер указателя (адреса) составляет 4 байта. Каждый элемент массива argv[] - это указатель на строку. Поэтому каждый следующий аргумент смещен на 4 байта относительно предыдущего

\$esp + 0:	argc	(количество аргументов)
\$esp + 4:	argv[0]	→ указатель на имя программы
\$esp + 8:	argv[1]	→ указатель на первый аргумент
\$esp + 12:	argv[2]	→ указатель на второй аргумент
\$esp + 16:	argv[3]	→ указатель на третий аргумент
\$esp + 20:	argv[4]	→ NULL (0x0) - конец массива
\$esp + 24:	envp[0]	→ первая переменная окружения

## 4.2 Задание для самостоятельной работы

### 1. Задание

Для выполнения этого задания для начала копирую файл lab8-1.asm в каталог ~/work/arch-pc/lab09 (рис. 26).

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ cp ~/work/arch-pc/lab08/lab8-4.asm ~/work/arch-pc/lab09/lab9-4.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ls
in_out.asm  lab9-1  lab9-1.asm  lab9-1.o  lab9-2  lab9-2.asm  lab9-2.lst  lab9-2.o  lab9-3  lab9-3.asm  lab9-3.lst  lab9-3.o  lab9-4.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ █
```

**Рисунок 26: Копирование файла в рабочий каталог**

Меняю программу lab9-4.asm из самостоятельной части предыдущей лабораторной работы с использованием подпрограммы (рис. 27).

The screenshot shows a Windows Notepad window with the title bar "\*lab9-4.asm (~/work/arch-pc/lab09)". The menu bar includes "Файл", "Правка", "Вид", "Поиск", "Сервис", "Документы", and "Справка". Below the menu is a toolbar with icons for file operations like Open, Save, Print, and Search. The main text area contains the following assembly code:

```
%include 'in_out.asm'

SECTION .data
msg_func db "Функция: f(x) = 10x - 4", 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
```

Рисунок 27: Изменение программы lab9-4.asm

Код программы:

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

**SECTION** .data

msg\_func **db** "Функция: f(x) = 10x - 4", 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, 4
```

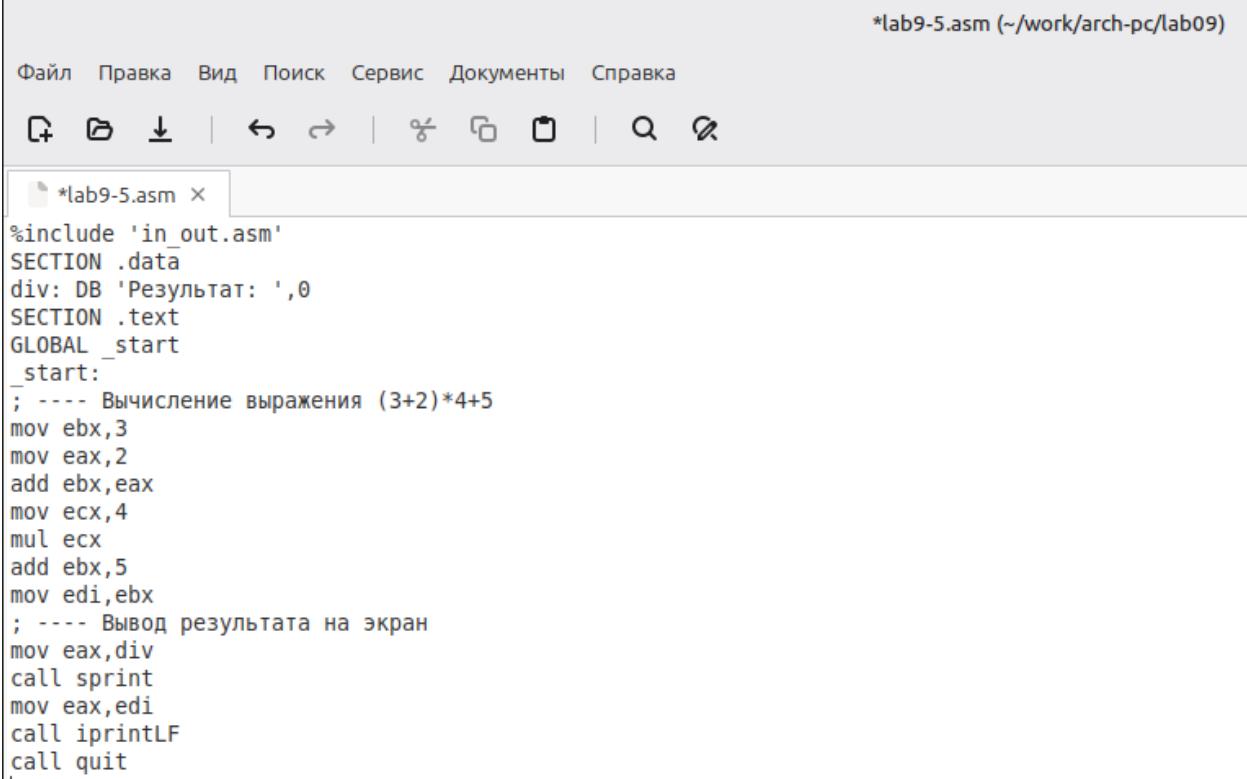
## 2. Задание

Для начала создаю файл lab9-5.asm для работы в этом файле (рис.28).

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ touch lab9-5.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ls
in_out.asm lab9-1.lab9-1.o lab9-2.lab9-2.asm lab9-2.lst lab9-2.o lab9-3.lab9-3.asm lab9-3.lst lab9-3.o lab9-4.asm lab9-5.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$
```

Рисунок 28: Создание файла lab9-5.asm

Копируем код из листинга 9.3 и вставляем в файл lab9-5.asm (рис. 29). Далее делаем отладку в gdb файла lab9-5.asm (рис. 30).



The screenshot shows a Windows Notepad window titled '\*lab9-5.asm (~/work/arch-pc/lab09)'. The menu bar includes 'Файл', 'Правка', 'Вид', 'Поиск', 'Сервис', 'Документы', and 'Справка'. Below the menu is a toolbar with icons for file operations. The main text area contains the following assembly code:

```
%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
||
```

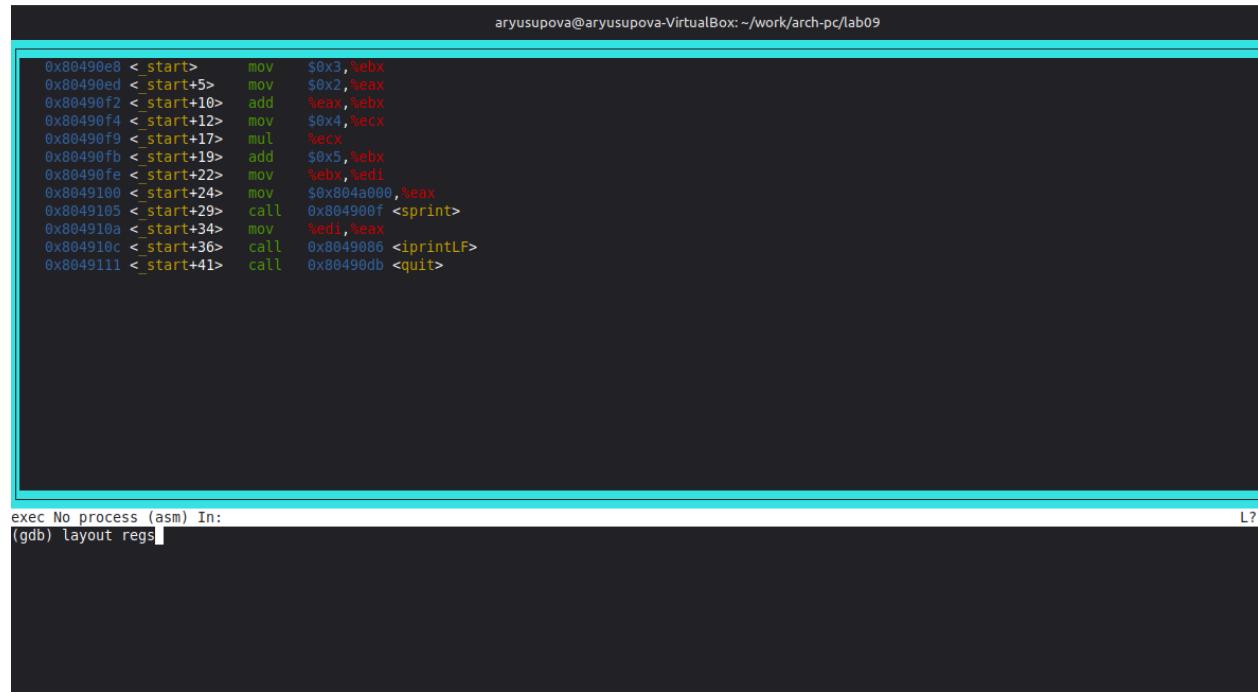
Рисунок 29: Код из листинга

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ nasm -f elf -g lab9-5.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-5 lab9-5.o
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ gdb ./lab9-5
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
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-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 ./lab9-5...
(gdb)
```

Рисунок 30: Отладка в gdb

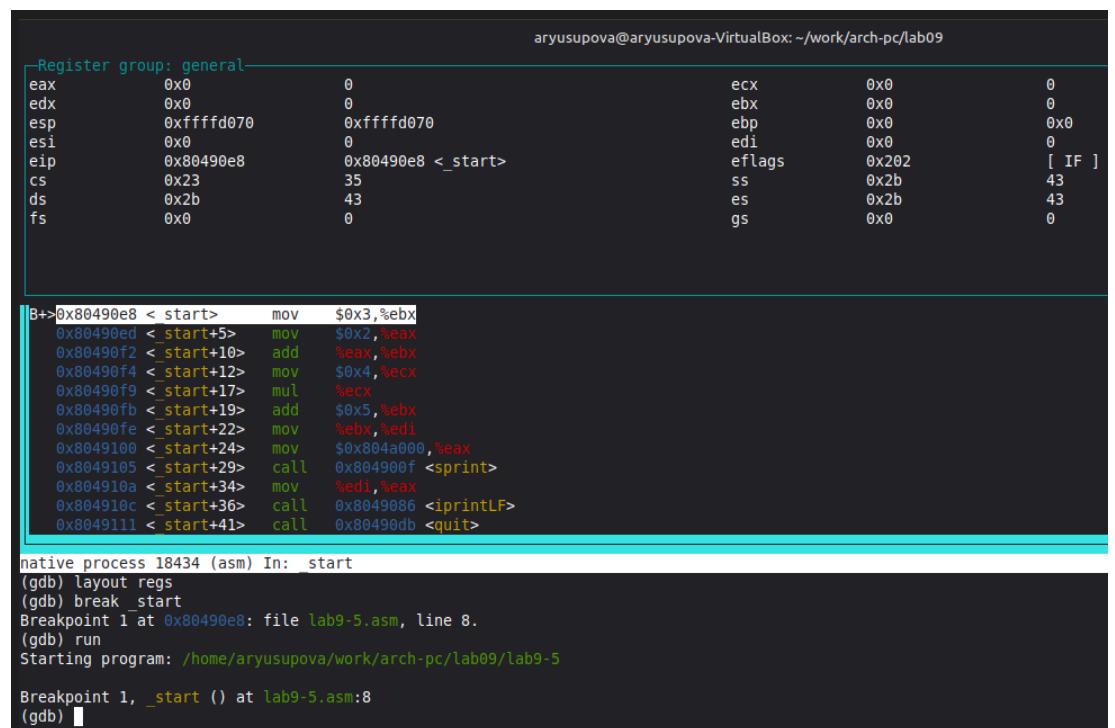
Включаю режим псевдографики для более удобного анализа программы (рис. 31).



The screenshot shows a terminal window with GDB running. The assembly code is displayed in a light blue background with syntax highlighting. The code starts at address 0x80490e8 and includes instructions like mov, add, mul, and calls to functions like sprint and iprintLF. Below the assembly code, the command 'layout regs' is entered, which changes the interface to show registers.

Рисунок 31: Режим псевдографики

Для более подробного анализа программы установила брейкпоинт на метку \_start, с которой начинается выполнение любой ассемблерной программы, и запустила её (рис. 32).



The screenshot shows the GDB interface with a register dump and assembly code. The register dump shows the state of general-purpose registers (eax, edx, esp, etc.) and flags. The assembly code is identical to Figure 31. Below the assembly code, the command 'break \_start' is entered, setting a breakpoint at the start of the program. The command 'run' is then entered to start the program. The output shows the program starting and reaching the breakpoint at address 0x80490e8.

Рисунок 32: Запуск программы и установка метки

Устанавливаем отображение регистров по шагам:

Шаг 1: Устанавливаем отображение регистров (mov ebx,3) (рис. 33)

Шаг 2: Проверка ошибки. Обошлось без них (mov eax,2) (рис. 34)

Шаг 3: Снова проверка ошибки (add ebx,eax) (рис. 35)

Шаг 4: Ещё одна проверка ошибки (mov ecx,4)(рис. 36).

Шаг 5: Ошибка найдена (рис. 37). Команда mul ecx умножает регистр eax на ecx, но в eax осталось значение 2 из предыдущей операции, а не результат сложения (5).

Программа вычисляет:

$$(2 * 4) + 5 = 13 \text{ вместо } (3 + 2) * 4 + 5 = 25$$

The screenshot shows a GDB session with the following details:

- Registers:** A table showing general registers (eax, edx, esp, esi, eip, cs, ds, fs) and control registers (ecx, ebx, ebp, edi, eflags, ss, es, gs). The values for eax, ebx, and ecx are highlighted.
- Assembly:** The assembly code for the program. The first few instructions are:

```
B+ 0x80490e8 < start>    mov    $0x3,%ebx
>0x80490ed < start+5>    mov    $0x2,%eax
0x80490f2 < start+10>     add    %eax,%ebx
0x80490f4 < start+12>     mov    $0x4,%ecx
0x80490f9 < start+17>     mul    %ecx
0x80490fb < start+19>     add    $0x5,%ebx
0x80490fe < start+22>     mov    %ebx,%edi
0x8049100 < start+24>     mov    $0x804a000,%eax
0x8049105 < start+29>     call   0x804900f <sprint>
0x804910a < start+34>     mov    %edi,%eax
0x804910c < start+36>     call   0x8049986 <iprintf>
0x8049111 < start+41>     call   0x80499db <quit>
```
- Stack Dump:** A native process dump showing the stack contents from address 18434 to 18438. It shows the assembly code for the start function and the current state of the registers.

Рисунок 33: Шаг 1 (Установление отображения регистров)

```

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x2          2          ecx      0x0          0
edx      0x0          0          ebx      0x3          3
esp      0xfffffd070    0xfffffd070  ebp      0x0          0x0
esi      0x0          0          edi      0x0          0
eip      0x80490f2    0x80490f2 < start+10>  eflags   0x10202    [ IF RF ]
cs       0x23         35         ss       0x2b         43
ds       0x2b         43         es       0x2b         43
fs       0x0          0          gs       0x0          0

B+ 0x80490e8 <_start>    mov    $0x3,%ebx
0x80490ed <start+5>    mov    $0x2,%eax
>0x80490f2 <start+10>  add    %eax,%ebx
0x80490f4 <start+12>    mov    $0x4,%ecx
0x80490f9 <start+17>    mul    %ecx
0x80490fb <start+19>    add    $0x5,%ebx
0x80490fe <start+22>    mov    %ebx,%edi
0x8049100 <start+24>    mov    $0x804a000,%eax
0x8049105 <start+29>    call   0x804900f <sprint>
0x804910a <start+34>    mov    %edi,%eax
0x804910c <start+36>    call   0x8049086 <iprintfLF>
0x8049111 <start+41>    call   0x80490db <quit>

native process 18434 (asm) In: start
1: /d $eax = 0
(gdb) display /d $ebx
2: /d $ebx = 3
(gdb) display /d $ecx
3: /d $ecx = 0
(gdb) display /d $edx
4: /d $edx = 0
(gdb) stepi
1: /d $eax = 2
2: /d $ebx = 3
3: /d $ecx = 0
4: /d $edx = 0
(gdb)

```

Рисунок 34: Шаг 2 проверки

```

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x2          2          ecx      0x0          0
edx      0x0          0          ebx      0x5          5
esp      0xfffffd070    0xfffffd070  ebp      0x0          0x0
esi      0x0          0          edi      0x0          0
eip      0x80490f4    0x80490f4 < start+12>  eflags   0x10206    [ PF IF RF ]
cs       0x23         35         ss       0x2b         43
ds       0x2b         43         es       0x2b         43
fs       0x0          0          gs       0x0          0

B+ 0x80490e8 <_start>    mov    $0x3,%ebx
0x80490ed <start+5>    mov    $0x2,%eax
>0x80490f2 <start+10>  add    %eax,%ebx
0x80490f4 <start+12>    mov    $0x4,%ecx
0x80490f9 <start+17>    mul    %ecx
0x80490fb <start+19>    add    $0x5,%ebx
0x80490fe <start+22>    mov    %ebx,%edi
0x8049100 <start+24>    mov    $0x804a000,%eax
0x8049105 <start+29>    call   0x804900f <sprint>
0x804910a <start+34>    mov    %edi,%eax
0x804910c <start+36>    call   0x8049086 <iprintfLF>
0x8049111 <start+41>    call   0x80490db <quit>

native process 18434 (asm) In: start
(gdb) display /d $edx
1: /d $edx = 0
(gdb) stepi
1: /d $eax = 2
2: /d $ebx = 3
3: /d $ecx = 0
4: /d $edx = 0
(gdb) stepi
1: /d $eax = 2
2: /d $ebx = 5
3: /d $ecx = 0
4: /d $edx = 0
(gdb)

```

Рисунок 35: Шаг 3 проверки

```

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x2          2
edx      0x0          0
esp     0xfffffd070    0xfffffd070
esi      0x0          0
eip     0x80490f9    0x80490f9 < start+17>
cs       0x23         35
ds       0x2b         43
fs       0x0          0
ecx      0x4          4
ebx      0x5          5
ebp      0x0          0x0
edi      0x0          0
eflags   0x10206    [ PF IF RF ]
ss       0x2b         43
es       0x2b         43
gs       0x0          0

B+ 0x80490e8 <_start>    mov    $0x3,%ebx
0x80490ed <_start+5>    mov    $0x2,%eax
0x80490f2 <_start+10>   add    %eax,%ebx
0x80490f4 <_start+12>   mov    $0x4,%ecx
>0x80490f9 <_start+17>  mul    %ecx
0x80490fb <_start+19>   add    $0x5,%ebx
0x80490fe <_start+22>   mov    %ebx,%edi
0x8049100 <_start+24>   mov    $0x804a000,%eax
0x8049105 <_start+29>   call   0x804900f <sprint>
0x804910a <_start+34>   mov    %edi,%eax
0x804910c <_start+36>   call   0x8049086 <iprintLF>
0x8049111 <_start+41>   call   0x80490db <quit>

native process 18434 (asm) In: _start
3: /d $ecx = 0
4: /d $edx = 0
(gdb) stepi
1: /d $eax = 2
2: /d $ebx = 5
3: /d $ecx = 0
4: /d $edx = 0
(gdb) stepi
1: /d $eax = 2
2: /d $ebx = 5
3: /d $ecx = 4
4: /d $edx = 0
(gdb)

```

Рисунок 36: Шаг 4 проверки

```

aryusupova@aryusupova-VirtualBox: ~/work/arch-pc/lab09

Register group: general
eax      0x8          8
edx      0x0          0
esp     0xfffffd070    0xfffffd070
esi      0x0          0
eip     0x80490fb    0x80490fb < start+19>
cs       0x23         35
ds       0x2b         43
fs       0x0          0
ecx      0x4          4
ebx      0x5          5
ebp      0x0          0x0
edi      0x0          0
eflags   0x10202    [ IF RF ]
ss       0x2b         43
es       0x2b         43
gs       0x0          0

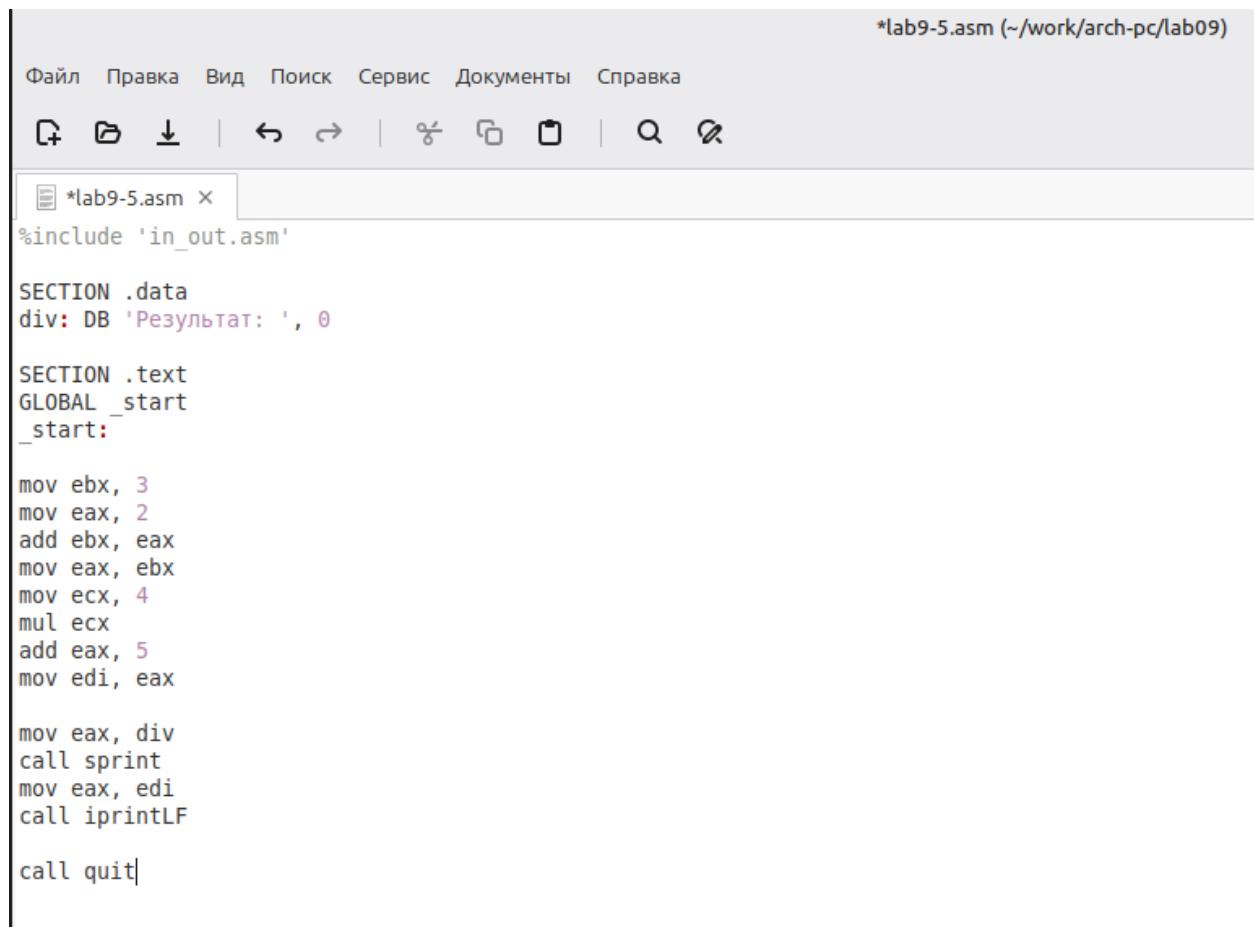
B+ 0x80490e8 <_start>    mov    $0x3,%ebx
0x80490ed <_start+5>    mov    $0x2,%eax
0x80490f2 <_start+10>   add    %eax,%ebx
0x80490f4 <_start+12>   mov    $0x4,%ecx
0x80490f9 <_start+17>   mul    %ecx
>0x80490fb <_start+19>  add    $0x5,%ebx
0x80490fe <_start+22>   mov    %ebx,%edi
0x8049100 <_start+24>   mov    $0x804a000,%eax
0x8049105 <_start+29>   call   0x804900f <sprint>
0x804910a <_start+34>   mov    %edi,%eax
0x804910c <_start+36>   call   0x8049086 <iprintLF>
0x8049111 <_start+41>   call   0x80490db <quit>

native process 18434 (asm) In: _start
3: /d $ecx = 0
4: /d $edx = 0
(gdb) stepi
1: /d $eax = 2
2: /d $ebx = 5
3: /d $ecx = 4
4: /d $edx = 0
(gdb) stepi
1: /d $eax = 8
2: /d $ebx = 5
3: /d $ecx = 4
4: /d $edx = 0
(gdb)

```

Рисунок 37: Шаг 5 (Найденная ошибка)

Исправим ошибки кода в файле lab9-5.asm (рис. 38).



The screenshot shows a text editor window with the title bar reading '\*lab9-5.asm (~/work/arch-pc/lab09)'. The menu bar includes 'Файл', 'Правка', 'Вид', 'Поиск', 'Сервис', 'Документы', and 'Справка'. Below the menu is a toolbar with icons for file operations. The main editor area contains the following assembly code:

```
%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
```

Рисунок 38: Изменение программы lab9-5.asm

Запускаю изменённую программу и вижу, что она работает корректно (рис. 39).

```
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ nasm -f elf lab9-5.asm
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ld -m elf_i386 -o lab9-5 lab9-5.o
aryusupova@aryusupova-VirtualBox:~/work/arch-pc/lab09$ ./lab9-5
Результат: 25
```

Рисунок 39: Запуск изменённой программы lab9-5.asm

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
```

## 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.

## 6 Список литературы

1. <https://esystem.rudn.ru/mod/resource/view.php?id=1030457>
2. <https://esystem.rudn.ru/mod/resource/view.php?id=1030557>
3. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).