

Лабораторная работа No10.

Понятие подпрограммы. Отладчик GDB.

Рассолова Маргарита Сергеевна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Самостоятельная работа	15
5	Выводы	17
	Список литературы	18

Список иллюстраций

3.1	Создание каталога и файла	7
3.2	Ввод текста программы	7
3.3	Создание исполняемого файла и проверка его работы	8
3.4	Создание исполняемого файла и проверка его работы	8
3.5	Создание второго файла	9
3.6	Ввод текста программы	9
3.7	Получение исполняемого файла и его загрузка в отладчик	9
3.8	Проверка	10
3.9	Проверка	10
3.10	Дисассимилированный код программы	10
3.11	Дисассимилированный код программы	11
3.12	Режим псевдографики	11
3.13	Проверка установки	11
3.14	Установка точки останова	11
3.15	Просмотр информации об установках	12
3.16	Просмотр информации об установках	12
3.17	Просмотр информации об установках	12
3.18	Вывод в различных форматах значение регистра edx	13
3.19	Изменение значение регистра ebx	13
3.20	Изменение значение регистра ebx	13
3.21	Создание исполняемого файла и его загрузка в отладчик	14
3.22	Адрес вершины стека	14
3.23	Установка точки останова и ее запуск	14
4.1	Установка точки останова и ее запуск	15
4.2	Создание исполняемого файла и проверка его работы	16
4.3	Исправление ошибки	16
4.4	Создание исполняемого файла и проверка его работы	16

Список таблиц

1 Цель работы

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

2 Теоретическое введение

Подпрограмма - это программа, которую можно выполнять в разных местах в составе одной или нескольких программ. Отладчик — это узкоспециализированное средство разработки, которое присоединяется к работающему приложению и позволяет проверять код.

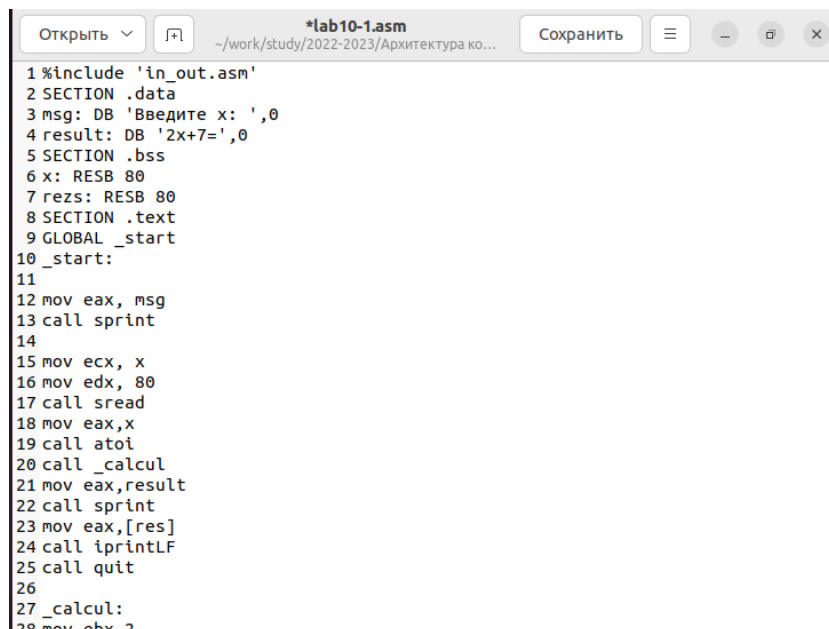
3 Выполнение лабораторной работы

1. Создала каталог для 10 лабораторной и файл к ней. (рис. 3.1)

```
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a
/arch-pc$ mkdir lab10
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc$ cd lab10
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ touch lab10-1.asm
```

Рис. 3.1: Создание каталога и файла

2. Ввела в созданный файл текст программы из Листинга 1. (рис. 3.2)



The screenshot shows a text editor window titled '*lab10-1.asm' with the following assembly code:

```
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 rezs: RESB 80
8 SECTION .text
9 GLOBAL _start
10 _start:
11
12 mov eax, msg
13 call sprint
14
15 mov ecx, x
16 mov edx, 80
17 call sread
18 mov eax, x
19 call atoi
20 call _calcul
21 mov eax, result
22 call sprint
23 mov eax, [rezs]
24 call iprintLF
25 call quit
26
27 _calcul:
28 mov ebx, 2
```

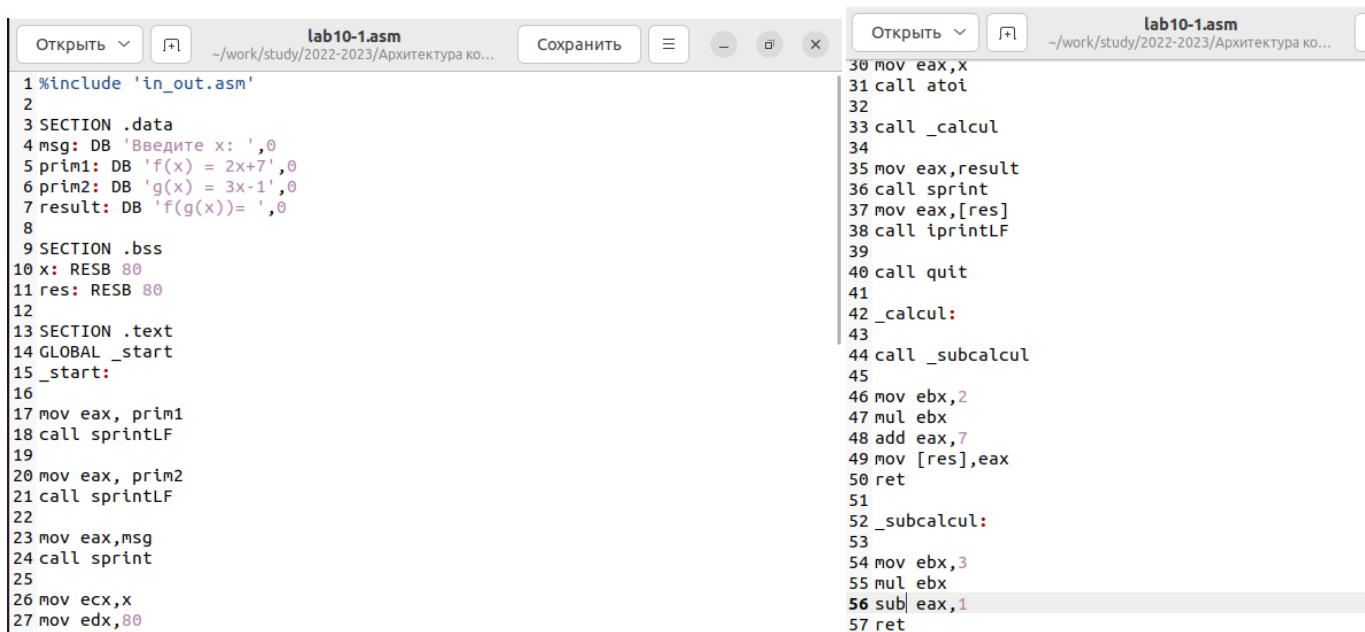
Рис. 3.2: Ввод текста программы

3. Создала исполняемый файл и проверила его работу. (рис. 3.3)

```
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ nasm -f elf lab10-1.asm
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ld -m elf_i386 -o lab10-1 lab10-1.o
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ./lab10-1
Введите x: 6
2x+7=19
```

Рис. 3.3: Создание исполняемого файла и проверка его работы

4. Изменила текст программы в соответствии с инструкцией. (рис. ??, рис. ??)



5. Создала исполняемый файл и проверила его работу. (рис. 3.4)

```
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ nasm -f elf lab10-1.asm
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ld -m elf_i386 -o lab10-1 lab10-1.o
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ./lab10-1
f(x) = 2x+7
g(x) = 3x-1
Введите x: 3
f(g(x))= 23
```

Рис. 3.4: Создание исполняемого файла и проверка его работы

6. Создала файл 2. (рис. 3.5)

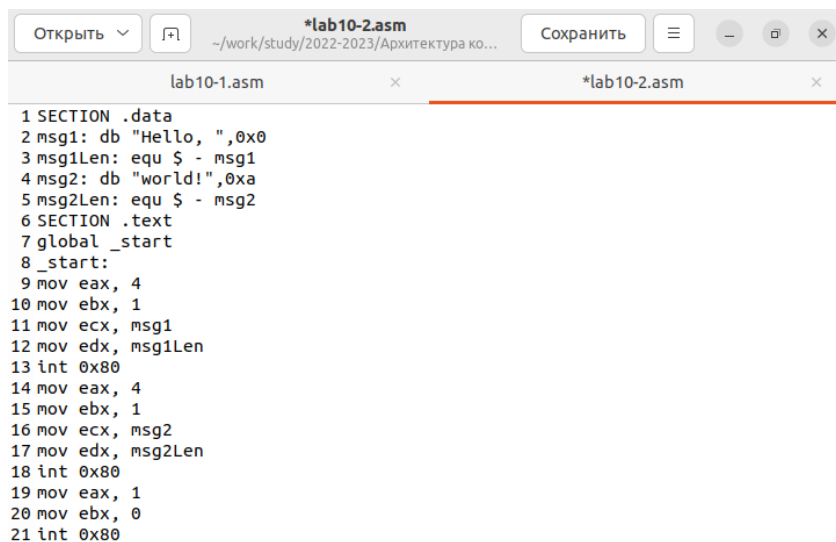

```

mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ touch lab10-2.asm
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ls
in_out.asm lab10-1 lab10-1.asm lab10-1.o lab10-2.asm

```

Рис. 3.5: Создание второго файла

7. Ввела в созданный файл текст программы из Листинга 2. (рис. 3.6)



```

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

```

Рис. 3.6: Ввод текста программы

8. Получила исполняемый файл и загрузила его в отладчик gdb. (рис. 3.7)

```

mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ nasm -f elf -g -l lab10-2.lst lab10-2.asm
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ld -m elf_i386 -o lab10-2 lab10-2.o
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ gdb lab10-2
GNU gdb (Ubuntu 12.0.90-0ubuntu1) 12.0.90
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-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 lab10-2...
(gdb)

```

Рис. 3.7: Получение исполняемого файла и его загрузка в отладчик

9. Проверила работу программы. (рис. 3.8)

```
(gdb) run
Starting program: /home/mikotseruba/work/study/2022-2023/Архитектура компьютера
/arch-pc/lab10/lab10-2
Hello, world!
[Inferior 1 (process 3512) exited normally]
(gdb)
```

Рис. 3.8: Проверка

10. Установила брейкпоинт на метку `_start`. Запустила ее. (рис. 3.9)

```
(gdb) break _start
Breakpoint 1 at 0x08049000: file lab10-2.asm, line 9.
(gdb) run
Starting program: /home/mikotseruba/work/study/2022-2023/Архитектура компьютера
/arch-pc/lab10/lab10-2

Breakpoint 1, _start () at lab10-2.asm:9
9      mov eax, 4
(gdb)
```

Рис. 3.9: Проверка

11. Посмотрела дисассимилированный код программы. (рис. 3.10)

```
(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. 3.10: Дисассимилированный код программы

12. Подключилась на отображение команд с Intel'овским синтаксисом. Различия: В коде Интел отсутствуют суффиксы обозначения размера, код Интел опускает символ “%” перед именами регистров, имеет другой способ описания местоположений в памяти. (рис. 3.11)

```

(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.

```

Рис. 3.11: Дисассимилированный код программы

13. Включила режим псевдографики. (рис. 3.12)

```

[ Register Values Unavailable ]

B+> 0x08049000 <_start>    mov     eax,0x4
      0x08049005 <_start+5>  mov     ebx,0x1
      0x0804900a <_start+10> mov     ecx,0x804a000
      0x0804900f <_start+15> mov     edx,0x8
      0x08049014 <_start+20> int     0x80
      0x08049016 <_start+22> mov     eax,0x4
      0x0804901b <_start+27> mov     ebx,0x1
      0x08049020 <_start+32> mov     ecx,0x804a008
native process 3532 In: _start L9 PC: 0x08049000
(gdb) layout regs

```

Рис. 3.12: Режим псевдографики

14. Проверила установку точки останова. (рис. 3.13)

```

(gdb) info breakpoints
Num   Type             Disp Enb Address      What
1      breakpoint       keep y 0x08049000 lab10-2.asm:9
       breakpoint already hit 1 time

```

Рис. 3.13: Проверка установки

15. Установила точку останова по адресу инструкции. (рис. 3.14)

```

(gdb) b *0x08049000
Note: breakpoint 1 also set at pc 0x08049000.
Breakpoint 2 at 0x08049000: file lab10-2.asm, line 9.

```

Рис. 3.14: Установка точки останова

16. Посмотрела информацию о всех установленных точках останова. (рис. 3.15)

```
(gdb) i b
Num      Type           Disp Enb Address      What
1        breakpoint     keep y   0x08049000 lab10-2.asm:9
          breakpoint already hit 1 time
2        breakpoint     keep y   0x08049000 lab10-2.asm:9
```

Рис. 3.15: Просмотр информации об установках

17. Посмотрела содержимое регистров. (рис. ??, рис. ??)

```
native process 3020 In: _start L9 PC: 0x8049000
eax      0x0      0
ecx      0x0      0
edx      0x0      0
ebx      0x0      0
esp      0xffffd110 0xffffd110
ebp      0x0      0x0
esi      0x0      0
edi      0x0      0
eip      0x8049000 0x8049000 <_start>
--Type <RET> for more, q to quit, c to continue without paging--

native process 3020 In: _start 0x8049000 <_start>
eip      0x8049000 0x8049000 <_start>
--Type <RET> for more, q to quit, c to continue without paging--
[ IF ]
cs      0x23      35
ss      0x2b      43
ds      0x2b      43
es      0x2b      43
fs      0x0      0
gs      0x0      0
(gdb)
```

18. Посмотрела значения переменных.(рис. ??, рис. ??)

```
(gdb) x/1sb &msg1
0x804a000 <msg1>: "Hello, "

(gdb) x/1sb 0x804a008
0x804a008 <msg2>: "world!\n\034"
```

19. Изменила первый символ переменной msg1. (рис. 3.16)

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

Рис. 3.16: Просмотр информации об установках

20. Заменяла символ в msg2. (рис. 3.17)

```
(gdb) set {char}&msg2='k'
(gdb) x/1sb &msg2
0x804a008 <msg2>: "korld!\n\034"
```

Рис. 3.17: Просмотр информации об установках

21. Вывела в шестнадцатиричном, двоичном форматах и символьном виде значение регистра edx. (рис. 3.18)

```
(gdb) p/x $edx
$1 = 0x0
(gdb) p/t $edx
$2 = 0
(gdb) p/s $edx
$3 = 0
```

Рис. 3.18: Вывод в различных форматах значение регистра edx

22. Изменила значение регистра ebx. (рис. 3.19)

```
(gdb) set $ebx='2'
(gdb) p/s $ebx
$4 = 50
(gdb) set $ebx=2
(gdb) p/s $ebx
$5 = 2
```

Рис. 3.19: Изменение значение регистра ebx

23. Скопировала файл 9 лабораторной в третий файл десятой лабораторной. (рис. 3.20)

```
mikotseruba@mikotseruba-VirtualBox:~/work/study/2022-2023/Архитектура_компьютер
a/arch-pc/lab10$ cp ~/work/study/2022-2023/Архитектура_компьютера/arch-pc/lab0
9/lab9-2.asm ~/work/study/2022-2023/Архитектура_компьютера/arch-pc/lab10/lab10
-3.asm
```

Рис. 3.20: Изменение значение регистра ebx

24. Создала исполняемый файл и загрузила его в отладчик, указав аргументы. (рис. 3.21)

```

Reading symbols from lab10-3...
(gdb) b _start
Breakpoint 1 at 0x80490e8: file lab10-3.asm, line 7.
(gdb) run
Starting program: /home/mikotseruba/work/study/2022-2023/Архитектура компьютера
/arch-pc/lab10/lab10-3 аргумент 1 аргумент 2 аргумент\ 3

Breakpoint 1, _start () at lab10-3.asm:7
7      pop ecx

```

Рис. 3.21: Создание исполняемого файла и его загрузка в отладчик

25. Посмотрела адрес вершины стека. (рис. 3.22)

```

(gdb) x/x $esp
0xffffd170:      0x00000006

```

Рис. 3.22: Адрес вершины стека

25. Установила точку останова и запустила ее. (рис. 3.23)

```

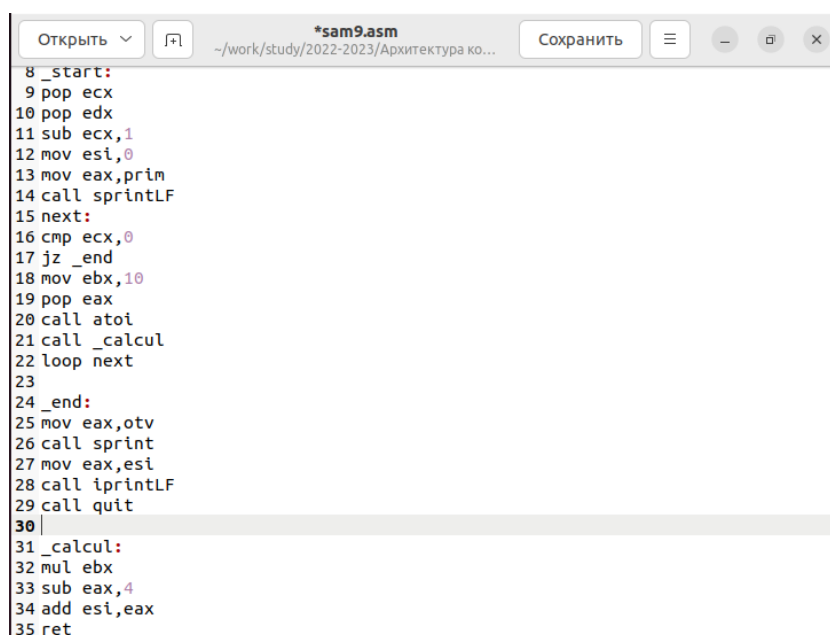
(gdb) x/s *(void**)(esp + 4)
0xffffd321:      "/home/mikotseruba/work/study/2022-2023/Архитектура компьютера/
arch-pc/lab10/lab10-3"
(gdb) Quit
(gdb) x/s *(void**)(esp + 8)
0xffffd38a:      "аргумент"
(gdb) x/s *(void**)(esp + 12)
0xffffd39b:      "1"
(gdb) x/s *(void**)(esp + 16)
0xffffd39d:      "аргумент"
(gdb) x/s *(void**)(esp + 20)
0xffffd3ae:      "2"
(gdb) x/s *(void**)(esp + 24)
0xffffd3b0:      "аргумент 3"

```

Рис. 3.23: Установка точки останова и ее запуск

4 Самостоятельная работа

26. Преобразовала программу из девятой лабораторной. (рис. 4.1)



```
8 _start:
9 pop ecx
10 pop edx
11 sub ecx, 1
12 mov esi, 0
13 mov eax, prim
14 call sprintfLF
15 next:
16 cmp ecx, 0
17 jz _end
18 mov ebx, 10
19 pop eax
20 call atoi
21 call _calcul
22 loop next
23
24 _end:
25 mov eax, otv
26 call sprint
27 mov eax, esi
28 call iprintfLF
29 call quit
30
31 _calcul:
32 mul ebx
33 sub eax, 4
34 add esi, eax
35 ret
```

Рис. 4.1: Установка точки останова и ее запуск

27. Создала исполняемый файл и проверила его работу. (рис. 4.2)

```

mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a
/arch-pc/lab10$ nasm -f elf sam10-1.asm
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ld -m elf_i386 -o lab10-4 lab10-4.o
ld: невозможно найти lab10-4.o: Нет такого файла или каталога
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ nasm -f elf sam10-1.asm
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ld -m elf_i386 -o sam10-1 sam10-1.o
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ./sam10-1 1 2 3
f(x)=10x-4
Результат: 48
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ./sam10-1 1 2 3 4
f(x)=10x-4
Результат: 84

```

Рис. 4.2: Создание исполняемого файла и проверка его работы

28. С помощью отладчика определила и исправила ошибку в листинге 3. (рис. 4.3)



```

Открыть  ~work/study/2022-2023/Архитектура ко...  Сохранить
*lab10-4.asm
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov ebx,3
8 mov eax,2
9 add eax,ebx
10 mov ecx,4
11 mul ecx
12 add eax,5
13 mov edi,eax
14 mov eax,div
15 call sprint
16 mov eax,edi
17 call iprintLF
18 call quit

```

Рис. 4.3: Исправление ошибки

29. Создала исполняемый файл и проверила его работу. (рис. 4.4)

```

mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ touch lab10-4.asm
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ nasm -f elf lab10-4.asm
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ld -m elf_i386 -o lab10-4 lab10-4.o
mikotseruba@mikotseruba-VirtualBox: ~/work/study/2022-2023/Архитектура компьютер
a/arch-pc/lab10$ ./lab10-4
Результат: 25

```

Рис. 4.4: Создание исполняемого файла и проверка его работы

5 Выводы

Приобрела навыки написания программ с использованием подпрограмм. Ознакомилась с методами отладки при помощи GDB и его основными возможностями.

Список литературы