

Отчёт по лабораторной работе №9

Архитектура компьютера

Исаханян Армен Артурович

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выполнение самостоятельной работы	12
6	Выводы	16
	Список литературы	17

Список иллюстраций

4.1	Создание каталога	8
4.2	Создание файла	8
4.3	Ввод программы	9
4.4	Запуск файла	9
4.5	Создание файла	10
4.6	Ввод программы	10
4.7	Получение файла	11
4.8	Копирование файла	11
4.9	Создание исполняемого файла и его загрузка	11
5.1	Копирование файла	12
5.2	Редактирование программы	13
5.3	Проверка	14
5.4	Редактирование программы	14
5.5	Проверка	15

Список таблиц

1 Цель работы

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

2 Задание

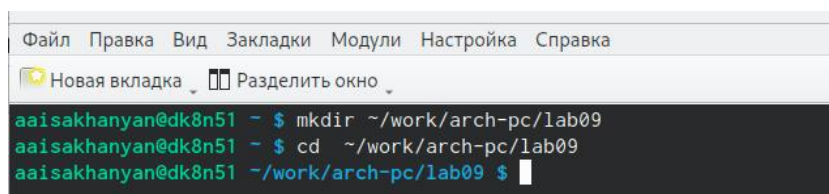
Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

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 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

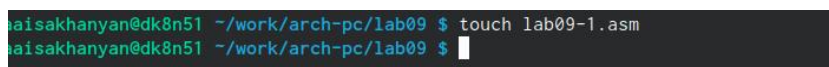
Создание каталога lab09 и переход в него (рис. 4.1).



```
Файл  Правка  Вид  Закладки  Модули  Настройка  Справка
Новая вкладка  Разделить окно
aaisakhanyan@dk8n51 ~ $ mkdir ~/work/arch-pc/lab09
aaisakhanyan@dk8n51 ~ $ cd ~/work/arch-pc/lab09
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $
```

Рис. 4.1: Создание каталога

Создание файла lab09-1.asm (рис. 4.2).



```
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ touch lab09-1.asm
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $
```

Рис. 4.2: Создание файла

Ввел программу в файл lab09-1.asm (рис. 4.3).


```
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 res: RESB 80
8 SECTION .text
9 GLOBAL _start
10 _start:
11 ;-----
12 ; Основная программа
13 ;-----
14 mov eax, msg
15 call sprint
16 mov ecx, x
17 mov edx, 80
18 call sread
19 mov eax, x
20 call atoi
21 call _calcul ; Вызов подпрограммы _calcul
22 mov eax, result
23 call sprint
24 mov eax, [res]
25 call iprintLF
26 call quit
27 ;-----
28 ; Подпрограмма вычисления
```

Рис. 4.3: Ввод программы

Запустил исполняемый файл со значением 5 (рис. 4.4).

```
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ nasm -f elf lab09-1.asm
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-1 lab09-1.o
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ ./lab09-1
Введите x: 5
2x+7=17
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $
```

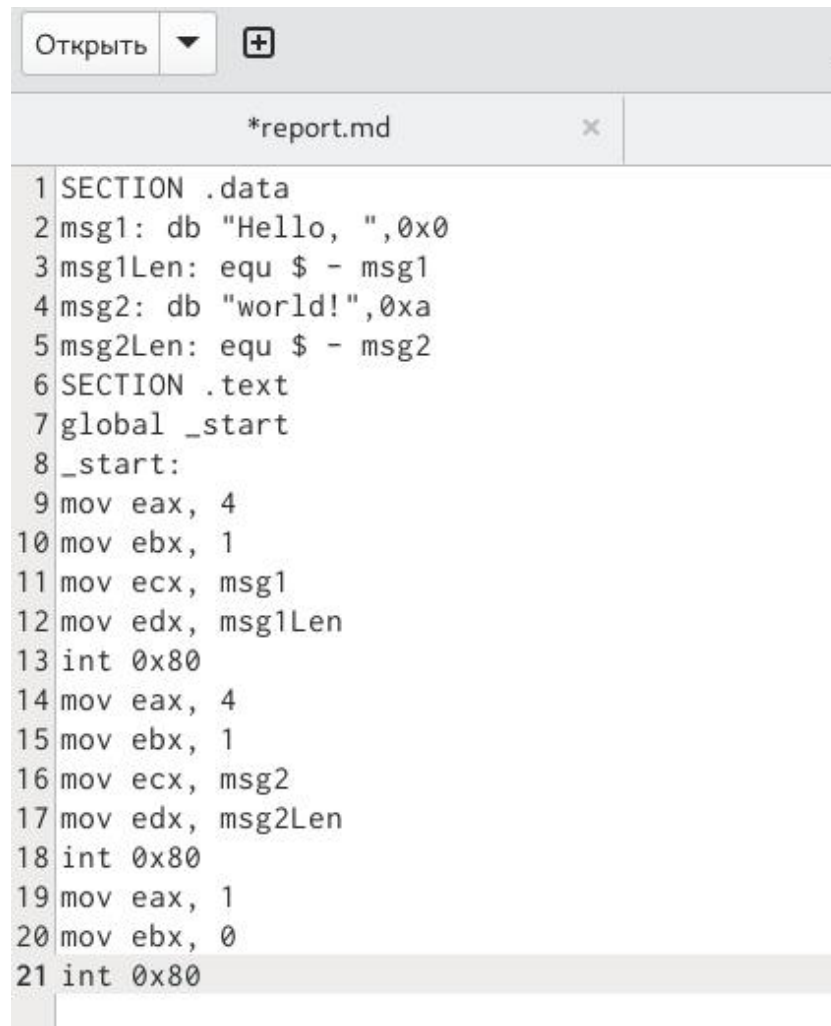
Рис. 4.4: Запуск файла

Создал файл lab09-1.asm в том же каталоге (рис. 4.5).

```
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ touch lab09-2.asm  
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $
```

Рис. 4.5: Создание файла

Ввел программы в файл (рис. 4.6).



The screenshot shows a text editor window with a title bar containing 'Открыть', a dropdown arrow, and a '+' icon. Below the title bar is a tab labeled '*report.md' with a close button 'x'. The main area of the window contains assembly code with line numbers from 1 to 21. The code defines two data sections, .data and .text, and implements a program that prints two messages: 'Hello, ' and 'world!'. It uses standard x86 assembly instructions for memory movement, register manipulation, and system calls (int 0x80).

```
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. 4.6: Ввод программы

Получил исполняемый файл с ключом -g (рис. 4.7).

```

aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ nasm -f elf -g -l lab09-2.lst lab09-2.asm
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-2 lab09-2.o
aaisakhanyan@dk8n51 ~/work/arch-pc/lab09 $ gdb lab09-2
GNU gdb (Gentoo 12.1 vanilla) 12.1
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-pc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://bugs.gentoo.org/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from lab09-2...
(gdb)

```

Рис. 4.7: Получение файла

Скопировал файл lab8-2.asm в файл с именем lab09-3.asm (рис. 4.8).

```

aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ cp ~/work/arch-pc/lab08/lab8-2.asm ~/work/arch-pc/lab09/lab09-3.asm
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $

```

Рис. 4.8: Копирование файла

Создал исполняемый файл, используя ключ `--args` для загрузки в `gdb`, загрузил исполняемый файл в отладчик, указав аргументы (рис. 4.9).

```

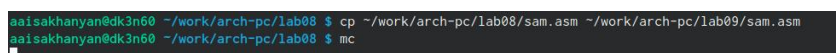
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ nasm -f elf -g -l lab09-3.lst lab09-3.asm
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o lab09-3 lab09-3.o
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ gdb --args lab09-3 аргумент1 аргумент 2 'аргумент 3

```

Рис. 4.9: Создание исполняемого файла и его загрузка

5 Выполнение самостоятельной работы

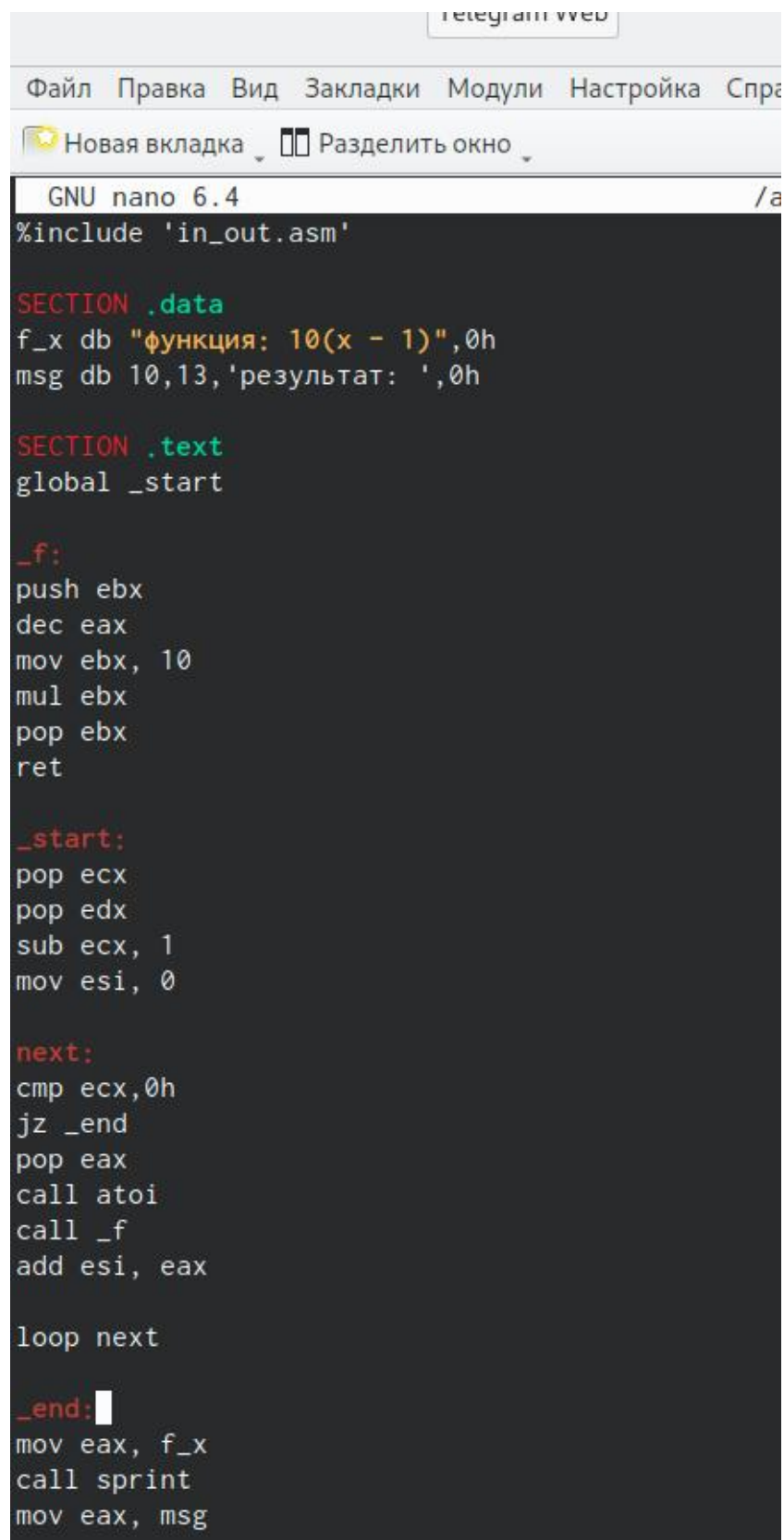
Скопировал файл из 8-ой лабораторной работы для выполнения самостоятельной работы в каталог `work/arch-pc/lab09` с названием `sam.asm` (рис. 5.1).

A terminal window showing two commands being executed. The first command is `cp ~/work/arch-pc/lab08/sam.asm ~/work/arch-pc/lab09/sam.asm` and the second is `mc`. The prompt is `aaishakhanyan@dk3n60`.

```
aaishakhanyan@dk3n60 ~/work/arch-pc/lab08 $ cp ~/work/arch-pc/lab08/sam.asm ~/work/arch-pc/lab09/sam.asm
aaishakhanyan@dk3n60 ~/work/arch-pc/lab08 $ mc
```

Рис. 5.1: Копирование файла

Программа для с/р 8-ой лабораторной работы, но с использованием подпрограмм (рис. 5.2).



```
GNU nano 6.4 /a
#include 'in_out.asm'

SECTION .data
f_x db "функция: 10(x - 1)",0h
msg db 10,13,'результат: ',0h

SECTION .text
global _start

_f:
push ebx
dec eax
mov ebx, 10
mul ebx
pop ebx
ret

_start:
pop ecx
pop edx
sub ecx, 1
mov esi, 0

next:
cmp ecx,0h
jz _end
pop eax
call atoi
call _f
add esi, eax

loop next

_end:
mov eax, f_x
call sprint
mov eax, msg
```

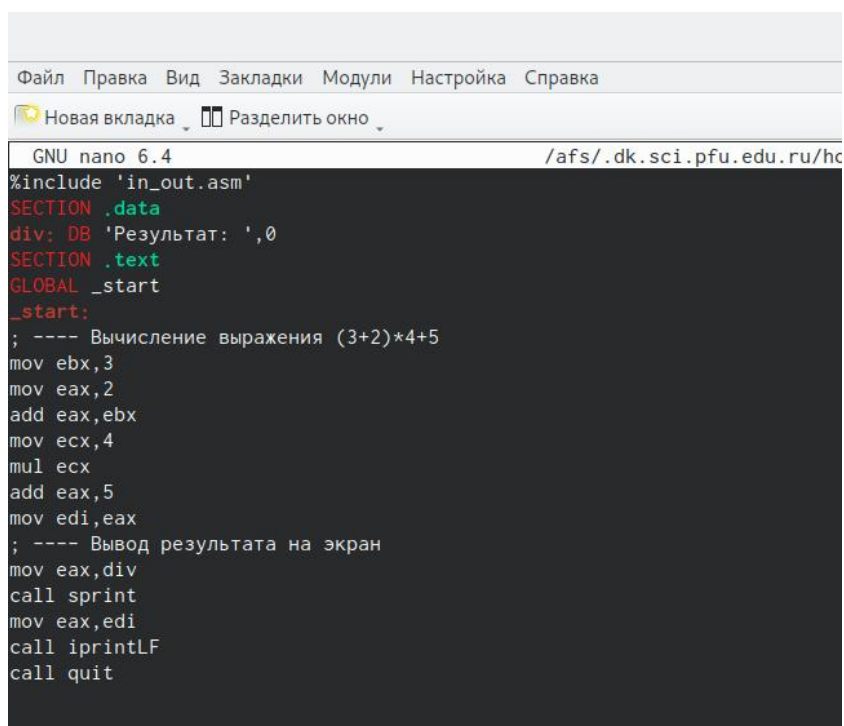
Рис. 5.2: Редактирование программы

Проверка работы программы (рис. 5.3).

```
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ nasm -f elf sam.asm
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o sam sam.o
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ ./sam
функция: 10(x - 1)
результат: 0
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ ./sam 1 2 3
функция: 10(x - 1)
результат: 30
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $
```

Рис. 5.3: Проверка

Редактировал программу из листинга 9.3 (рис. 5.4).



```
GNU nano 6.4 /afs/.dk.sci.pfu.edu.ru/hc
#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 eax,ebx
mov ecx,4
mul ecx
add eax,5
mov edi,eax
; ---- Вывод результата на экран
mov eax,div
call sprint
mov eax,edi
call iprintLF
call quit
```

Рис. 5.4: Редактирование программы

Проверил правильность редактирования программы (рис. 5.5).

```
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ nasm -f elf sam2.asm
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ ld -m elf_i386 -o sam2 sam2.o
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $ ./sam2
Результат: 25
aaisakhanyan@dk3n60 ~/work/arch-pc/lab09 $
```

Рис. 5.5: Проверка

6 Выводы

Я приобрел навыки написания программ с использованием подпрограмм. Познакомился с методами отладки при помощи GDB и его основными возможностями.

Список литературы