

Лабораторная работа №9

Дисциплина: Архитектура компьютера

Ким Ангелина Павловна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	22
	Список литературы	23

Список иллюстраций

3.1	Создание файла lab9-1.asm	7
3.2	Текст программы из листинга 9.1	8
3.3	Создание исполняемого файла	9
3.4	Измененная программа (1)	10
3.5	Создание исполняемого файла	11
3.6	Измененная программа (2)	12
3.7	Создание исполняемого файла	13
3.8	Текст программы из листинга 9.2	14
3.9	Создание исполняемого файла	14
3.10	Текст программы из листинга 9.3	16
3.11	Создание исполняемого файла	17
3.12	Измененная программа из листинга 9.3	18
3.13	Создание исполняемого файла	19
3.14	Текст программы вариант-20	20
3.15	Создание исполняемого файла	21

Список таблиц

1 Цель работы

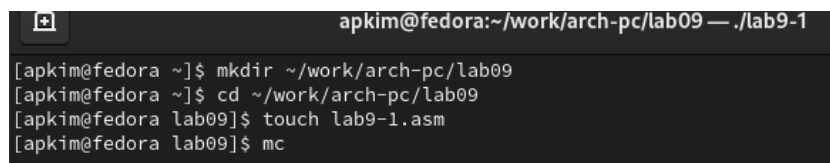
В ходе данной лабораторной работы мне нужно приобрести навыки написания программ с использованием циклов и обработкой аргументов командной строки.

2 Задание

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

3 Выполнение лабораторной работы

Создаем каталог для программ лабораторной работы №9, переходим в него и создаем файл lab9-1.asm (рис. 3.1)

A terminal window with a dark background. The title bar shows 'apkim@fedora:~/work/arch-pc/lab09 — ./lab9-1'. The terminal contains the following commands and their outputs:

```
[apkim@fedora ~]$ mkdir ~/work/arch-pc/lab09
[apkim@fedora ~]$ cd ~/work/arch-pc/lab09
[apkim@fedora lab09]$ touch lab9-1.asm
[apkim@fedora lab09]$ mc
```

Рис. 3.1: Создание файла lab9-1.asm

Введем в файл lab9-1.asm текст программы из листинга 9.1. (рис. 3.2)

```
GNU nano 5.8 /l
#include 'in_out.asm'
SECTION .data
    msg1 db 'Введите N: ',0h

SECTION .bss
    N:      resb 10

SECTION .text
    global _start
_start:

    mov eax,msg1
    call sprint

    mov ecx, N
    mov edx, 10
    call sread

    mov eax,N
    call atoi
    mov [N],eax

    mov ecx,[N]
label:
    mov [N],ecx
    mov eax,[N]
    call iprintLF

    loop label
```

Рис. 3.2: Текст программы из листинга 9.1

Создаем исполняемый файл и проверяем его работу. (рис. 3.3)

```
[apkim@fedora lab09]$ mc  
  
[apkim@fedora lab09]$ nasm -f elf lab9-1.asm  
[apkim@fedora lab09]$ ld -m elf_i386 -o lab9-1 lab9-1.o  
[apkim@fedora lab09]$ ./lab9-1  
Введите N: 11  
11  
10  
9  
8  
7  
6  
5  
4  
3  
2  
1  
Ошибка сегментирования (стек памяти сброшен на диск)  
[apkim@fedora lab09]$
```

Рис. 3.3: Создание исполняемого файла

Меняем текст программы, добавив изменение значения регистра есх в цикле
(рис. 3.4)

```

GNU nano 5.8 /home/apkim/w
%include 'in_out.asm'
SECTION .data
    msg1 db 'Введите N: ',0h

SECTION .bss
    N:      resb 10

SECTION .text
    global _start
_start:

    mov eax,msg1
    call sprint

    mov ecx, N
    mov edx, 10
    call sread

    mov eax,N
    call atoi
    mov [N],eax

    mov ecx,[N]
label:
    sub ecx,1
    mov [N],ecx
    mov eax,[N]
    call iprintLF

    loop label

```

Рис. 3.4: Измененная программа (1)

Создаем исполняемый файл и проверяем его работу. Данный пример показывает, что использование регистра ecx в теле цикла может привести к некорректной работе программы. Число проходов не соответствует значению N введенному с клавиатуры, программа зациклилась. (рис. 3.5)

```
[apkim@fedora lab09]$ nasm -f elf lab9-1.asm
lab9-1.asm:1: error: unable to open include file 'in_out.asm': No such file or directory
[apkim@fedora lab09]$ nasm -f elf lab9-1.asm
[apkim@fedora lab09]$ ld -m elf_i386 -o lab9-1 lab9-1.o
[apkim@fedora lab09]$ ./lab9-1
Введите N: █
```

Рис. 3.5: Создание исполняемого файла

Вносим изменения в текст программы, добавив команды `push` и `pop` (добавление в стек и извлечения из стека) для сохранения значения счетчика цикла. (рис. 3.6)

```

GNU nano 5.8 /home/apkim/
%include 'in_out.asm'
SECTION .data
    msg1 db 'Введите N: ',0h

SECTION .bss
    N:      resb 10

SECTION .text
    global _start
_start:

    mov eax,msg1
    call sprint

    mov ecx, N
    mov edx, 10
    call sread

    mov eax,N
    call atoi
    mov [N],eax

    mov ecx,[N]
label:
    push ecx
    sub ecx,1
    mov [N],ecx
    mov eax,[N]
    call iprintLF
    pop ecx

    loop label

```

Рис. 3.6: Измененная программа (2)

Создаем исполняемый файл и проверяем его работу. В данном случае число проходов цикла значению N введенному с клавиатуры. (рис. 3.7)

```
[apkim@fedora lab09]$ nasm -f elf lab9-1.asm
[apkim@fedora lab09]$ ld -m elf_i386 -o lab9-1 lab9-1.o
[apkim@fedora lab09]$ ./lab9-1
Введите N: 11
10
9
8
7
6
5
4
3
2
1
0
Ошибка сегментирования (стек памяти сброшен на диск)
[apkim@fedora lab09]$
```

Рис. 3.7: Создание исполняемого файла

Создаем файл lab9-2.asm и вводим в него текст программы из листинга 9.2 (рис. 3.8)

```

GNU nano 3.0 /10
#include 'in_out.asm'
SECTION .text
global _start
_start:
    pop ecx

    pop edx

    sub ecx, 1

next:
    cmp ecx, 0
    jz _end

    pop eax
    call sprintf
    loop next
_end:
    call quit

```

Рис. 3.8: Текст программы из листинга 9.2

Создаем исполняемый файл и проверяем его работу. Программой было обработано 4 аргумента (рис. 3.9)

```

[apkim@fedora lab09]$ nasm -f elf lab9-2.asm
[apkim@fedora lab09]$ ld -m elf_i386 -o lab9-2 lab9-2.o
[apkim@fedora lab09]$ ./lab9-2
[apkim@fedora lab09]$ ./lab9-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
[apkim@fedora lab09]$

```

Рис. 3.9: Создание исполняемого файла

Создаем файл lab9-3.asm и вводим туда текст программы из листинга 9.3 (рис. 3.10)

```

GNU nano 5.8
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
    pop ecx
    pop edx

    sub ecx,1

    mov esi, 0

next:
    cmp ecx,0h
    jz _end

    pop eax
    call atoi
    add esi,eax

    loop next

_end:
    mov eax,msg
    call sprint
    mov eax,esi
    call iprintLF
    call quit

```

Рис. 3.10: Текст программы из листинга 9.3

Создаем исполняемый файл и проверяем его работу, указав аргументы.(рис. 3.11)

```
[apkim@fedora lab09]$ touch lab9-3.asm
[apkim@fedora lab09]$ mc

[apkim@fedora lab09]$ nasm -f elf lab9-3.asm
[apkim@fedora lab09]$ ld -m elf_i386 -o lab9-3 lab9-3.o
[apkim@fedora lab09]$ ./lab9-2 12 13 7 10 5
12
13
7
10
5
[apkim@fedora lab09]$ ./lab9-3 12 13 7 10 5
Результат: 47
[apkim@fedora lab09]$
```

Рис. 3.11: Создание исполняемого файла

Изменяем текст программы из листинга 9.3 для вычисления произведения аргументов командной строки. (рис. 3.12)

```

GNU nano 5.8
%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
    pop ecx
    pop edx

    sub ecx,1

    mov esi, 1

next:
    cmp ecx,0h
    jz _end

    pop eax
    call atoi
    mul esi
    mov esi,eax

    loop next

_end:
    mov eax,msg
    call sprint
    mov eax,esi
    call iprintLF
    call quit

```

Рис. 3.12: Измененная программа из листинга 9.3

Создаем исполняемый файл и проверяем его работу. (рис. 3.13)

```
[arkim@fedora lab09]$ nasm -f elf lab9-3.asm
[arkim@fedora lab09]$ ld -m elf_i386 -o lab9-3 lab9-3.o
[arkim@fedora lab09]$ ./lab9-3 12 13 7 10 5
Результат: 54600
[arkim@fedora lab09]$ ./lab9-3 12 2
Результат: 24
[arkim@fedora lab09]$
```

Рис. 3.13: Создание исполняемого файла

Задание для самостоятельной работы. У меня вариант 20. Текст программы, которая находит сумму значений функции. (рис. 3.14)

```

%include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
    pop ecx
    pop edx

    sub ecx,1

    mov ebx,3

    mov esi,0

next:
    cmp ecx,0h
    jz _end

    pop eax
    call atoi
    add eax, 10
    mul ebx
    add esi,eax

    loop next

_end:
    mov eax,msg
    call sprint
    mov eax,esi
    call iprintLF
    call quit

```

Рис. 3.14: Текст программы вариант-20

Создаем исполняемый файл и проверяем его работу. (рис. 3.15)

```
[arkim@fedora lab09]$ nasm -f elf lab9-3.asm
[arkim@fedora lab09]$ ld -m elf_i386 -o lab9-3 lab9-3.o
[arkim@fedora lab09]$ ./lab9-3
Результат: 0
[arkim@fedora lab09]$ ./lab9-3 1 2
Результат: 69
[arkim@fedora lab09]$ ./lab9-3 1 2 3
Результат: 108
[arkim@fedora lab09]$
```

Рис. 3.15: Создание исполняемого файла

4 Выводы

Исходя из этой работы, я приобрела навыки написания программ с использованием циклов и обработкой аргументов командной строки.

Список литературы