

Отчет по лабораторной работе №8

Дисциплина: архитектура компьютера

Желобицкая П.А.

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Выполнение заданий самостоятельной работы	13
5	Выводы	16
	Список литературы	17

Список иллюстраций

3.1	Создание каталога и файла	7
3.2	Текст программы	7
3.3	Создание файла	8
3.4	Внесение изменений	8
3.5	Проверка работы файла	8
3.6	Внесение изменений	9
3.7	Проверка работы файла	9
3.8	Создание файла	9
3.9	Ввод программы	10
3.10	Запуск файла	10
3.11	Создание файла	10
3.12	Ввод программы	11
3.13	Ввод программы	11
3.14	Изменения в программе	12
3.15	Проверка работы файла	12
4.1	Создание файл	13
4.2	Написание программы	14
4.3	Проверка работы программы	14
4.4	Название рисунка	15

Список таблиц

2.1	Описание некоторых каталогов файловой системы GNU Linux . .	6
-----	---	---

1 Цель работы

Приобрести навыки написания программ с использованием циклов и обработкой аргументов командной строки.

2 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы. Например, в табл. 2.1 приведено краткое описание стандартных каталогов Unix.

Таблица 2.1: Описание некоторых каталогов файловой системы GNU Linux

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую систему
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно об Unix см. в [1–6].

3 Выполнение лабораторной работы

Создаю каталог, перехожу в него и создаю в нем файл lab8-1.asm (рис. 4.4).

```
pazhelobickaya@dk3n62 ~ $ mkdir ~/work/arch-pc/lab08
pazhelobickaya@dk3n62 ~ $ cd ~/work/arch-pc/lab08
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ touch lab8-1.asm
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ mc
```

Рис. 3.1: Создание каталога и файла

Ввожу в файл lab8-1.asm текст программы из листинга 8.1 (рис. 3.2).

```
/afs/.dk.sci.pfu.edu.ru/home/p/a/pazhelobickaya/work/arch-pc/lab08/lab8-1.asm
;-----
; Программа вывода значений регистра 'ecx'
;-----
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ----- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ----- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, 'ecx=N'
label:
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit
```

Рис. 3.2: Текст программы

Создаю исполняемый файл и проверяю его работу (рис. 3.3).

```
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 2
2
1
```

Рис. 3.3: Создание файла

Вношу изменения в текст программы (рис. 3.4).

```
/afs/.dk.sci.pfu.edu.ru/home/p/a/pazhelobickaya/work/arch-pc/lab08/lab8-1.asm
;-----
; Программа вывода значений регистра 'ecx'
;-----
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ----- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ----- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, 'ecx=N'
label:
sub ecx,1 ; 'ecx=ecx-1'
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit
```

Рис. 3.4: Внесение изменений

Создаю исполняемый файл и проверяю его работу (рис. 3.5).

```
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 2
1
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $
```

Рис. 3.5: Проверка работы файла

Вношу изменения в текст программы (рис. 3.6).

```
/afs/.dk.sci.pfu.edu.ru/home/p/a/pazhelobickaya/work/arch-pc/lab08/lab8-1.asm
;-----
; Программа вывода значений регистра 'ecx'
;-----
#include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ----- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ----- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ----- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, 'ecx=N'
label:
push ecx ; добавление значения ecx в стек
sub ecx,1 ; 'ecx=ecx-1'
mov [N],ecx
mov eax,[N]
call iprintf ; Вывод значения 'N'
pop ecx ; извлечение значения ecx из стека
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit
```

Рис. 3.6: Внесение изменений

Создаю исполняемый файл и проверяю его работу (рис. 3.7).

```
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ./lab8-1
Введите N: 4
3
2
1
0
```

Рис. 3.7: Проверка работы файла

Создаю файл lab8-2.asm (рис. 3.8).

```
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ touch lab8-2.asm
```

Рис. 3.8: Создание файла

Ввожу в этот файл текст программы из листинга 8.2 (рис. 3.9).

```
/afs/.dk.sci.pfu.edu.ru/home/p/a/pazhelobickaya/work/arch-pc/lab08/lab8-2.asm
%include 'in_out.asm'
SECTION .text
global _start
_start:
    pop ecx ; Извлекаем из стека в 'ecx' количество
             ; аргументов (первое значение в стеке)
    pop edx ; Извлекаем из стека в 'edx' имя программы
             ; (второе значение в стеке)
    sub ecx, 1 ; Уменьшаем 'ecx' на 1 (количество
             ; аргументов без названия программы)
    next:
    cmp ecx, 0 ; проверяем, есть ли еще аргументы
    jz _end ; если аргументов нет выходим из цикла
             ; (переход на метку '_end')
    pop eax ; иначе извлекаем аргумент из стека
    call sprintf ; вызываем функцию печати
    loop next ; переход к обработке следующего
             ; аргумента (переход на метку 'next')
_end:
    call quit
```

Рис. 3.9: Ввод программы

Создаю исполняемый файл и запускаю его (рис. 3.10).

```
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-2.asm
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ./lab8-2 аргумент1 аргумент 2 'аргумент 3'
аргумент1
аргумент
2
аргумент 3
```

Рис. 3.10: Запуск файла

Создаю файл lab8-3.asm (рис. 3.11).

```
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ touch lab8-3.asm
```

Рис. 3.11: Создание файла

Ввожу в этот файл текст программы из листинга 8.3 (рис. 3.12).

```

/afs/.dk.sci.pfu.edu.ru/home/p/a/pazhelobickaya/work/arch-pc/lab08/lab8-3.asm
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx ; Извлекаем из стека в 'ecx' количество
; аргументов (первое значение в стеке)
pop edx ; Извлекаем из стека в 'edx' имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем 'ecx' на 1 (количество
; аргументов без названия программы)
mov esi, 0 ; Используем 'esi' для хранения
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку '_end')
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
add esi,eax ; добавляем к промежуточной сумме
; след. аргумент 'esi=esi+eax'
loop next ; переход к обработке следующего аргумента
_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint
mov eax, esi ; записываем сумму в регистр 'eax'
call iprintLF ; печать результата
call quit ; завершение программы

```

Рис. 3.12: Ввод программы

Создаю исполняемый файл и запускаю его (рис. 3.13).

```

pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 10 5
Результат: 47

```

Рис. 3.13: Ввод программы

Вношу изменения в текст программы из листинга 8.3 для вычисления произведения аргументов командной строки (рис. 3.14).

```

/afs/.dk.sci.pfu.edu.ru/home/p/a/pazhelobickaya/work/arch-pc/lab08/lab8-3.asm
#include 'in_out.asm'
SECTION .data
msg db "Результат: ",0
SECTION .text
global _start
_start:
pop ecx ; Извлекаем из стека в 'ecx' количество
; аргументов (первое значение в стеке)
pop edx ; Извлекаем из стека в 'edx' имя программы
; (второе значение в стеке)
sub ecx,1 ; Уменьшаем 'ecx' на 1 (количество
; аргументов без названия программы)
mov esi, 1 ; Используем 'esi' для хранения
; промежуточных сумм
next:
cmp ecx,0h ; проверяем, есть ли еще аргументы
jz _end ; если аргументов нет выходим из цикла
; (переход на метку '_end')
pop eax ; иначе извлекаем следующий аргумент из стека
call atoi ; преобразуем символ в число
mul esi
mov esi,eax
; след. аргумент 'esi=esi*eax'
loop next ; переход к обработке следующего аргумента
_end:
mov eax, msg ; вывод сообщения "Результат: "
call sprint
mov eax, esi ; записываем сумму в регистр 'eax'
call iprintLF ; печать результата
call quit ; завершение программы

```

Рис. 3.14: Изменения в программе

Создаю исполняемый файл, запускаю его и проверяю его работу (рис. 3.15).

```

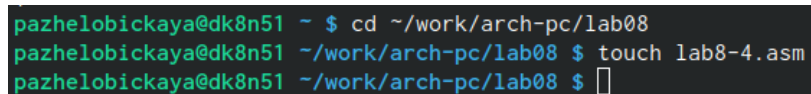
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ nasm -f elf lab8-3.asm
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-3 lab8-3.o
pazhelobickaya@dk3n62 ~/work/arch-pc/lab08 $ ./lab8-3 12 13 7 10 5
Результат: 54600

```

Рис. 3.15: Проверка работы файла

4 Выполнение заданий самостоятельной работы

Создаю файл lab8-4.asm (рис. 4.1).

A terminal window with a dark background and light green text. It shows three lines of commands and their outputs. The first line is 'cd ~/work/arch-pc/lab08', the second is 'touch lab8-4.asm', and the third is an empty prompt. The prompt for each line is 'pazhelobickaya@dk8n51 ~\$'.

```
pazhelobickaya@dk8n51 ~ $ cd ~/work/arch-pc/lab08
pazhelobickaya@dk8n51 ~/work/arch-pc/lab08 $ touch lab8-4.asm
pazhelobickaya@dk8n51 ~/work/arch-pc/lab08 $
```

Рис. 4.1: Создание файл

Пишу программу, которая выполняет условия (рис. 4.2).

```

/afs/.dk.sci.pfu.edu.ru/home/p/a/pazhelobickaya/work/arch-pc/lab08/lab8-4.asm
%include 'in_out.asm'

SECTION .data
f_x db 'Функция: 5(2+x)', 0h
msg db 'Результат: ', 0h

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 eax, 2
mov ebx, 5
mul ebx
add esi, eax

loop next

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

call quit

```

Рис. 4.2: Написание программы

Создаю исполняемый файл, запускаю его и проверяю его работу (рис. 4.3).

```

pazhelobickaya@dk8n51 ~/work/arch-pc/lab08 $ nasm -f elf lab8-4.asm
pazhelobickaya@dk8n51 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-4 lab8-4.o
pazhelobickaya@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3
Функция: 5(2+x)Результат: 60
pazhelobickaya@dk8n51 ~/work/arch-pc/lab08 $ ./lab8-4 1 2 3 4 5
Функция: 5(2+x)Результат: 125

```

Рис. 4.3: Проверка работы программы

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрацию (рис. 4.4).

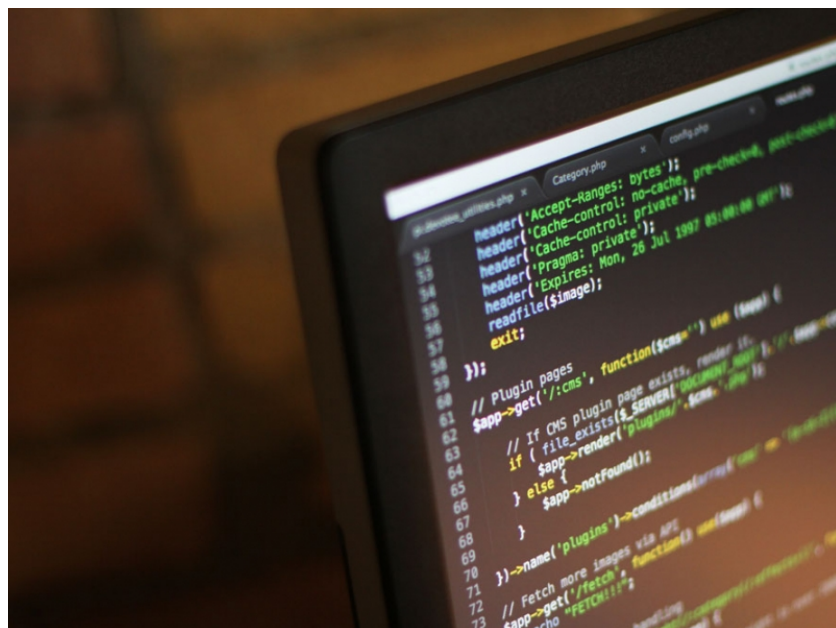


Рис. 4.4: Название рисунка

5 Выводы

Я получила навыки по организации циклов и работе со стеком на языке NASM.

Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016.
URL: <https://www.gnu.org/software/bash/manual/>.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.