МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
имени Н.Э. БАУМАНА

Учебное пособие

А.Е.Аверьянихин

Курс семинаров

«Системное программирование»

МГТУ имени Н.Э. Баумана

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
имени Н.Э. БАУМАНА

А.Е.Аверьянихин

Курс семинаров

«Системное программирование»

Москва
МГТУ имени Н.Э. Баумана

2012

УДК 681.3.06(075.8)
ББК 32.973-018
И201

А.Е.Аверьянихин
Курс семинаров «Системы искусственного интеллекта» / под ред. И.М.Холина - М.: МГТУ им. Н.Э. Баумана, 2012. – 26 с.: ил.

В курсе семинаров рассмотрены основные этапы практической части курса «Системное программирование».

Ил. 39. Табл. 5. Библиогр. 7 назв.

УДК 681.3.06(075.8)

АННОТАЦИЯ

В курсе семинаров будут рассмотрены основные темы практической части курса «Системное программирования» такие как: представление данных в вычислительных системах, основы работы с ОС Linux теории алгоритмов и программирования на языке С.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данный конспект семинаров составлен на основе семинарского курса, читаемого в МГТУ им. Н.Э. Баумана на кафедре ИУ4 преподавателем Аверьянихиным А.Е. Курс семинаров рекомендован к выполнению текущих аттестационных мероприятий и подготовки к зачету по предмету «Системное программирование».

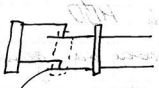
Семинары

Семинар №1 | 2012-09-05

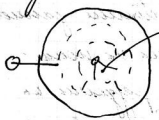
Обзор процедуры загрузки ОС

Системы адресации CHS (8×8×8)

цилиндр, головка, сектор

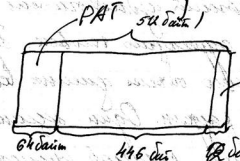


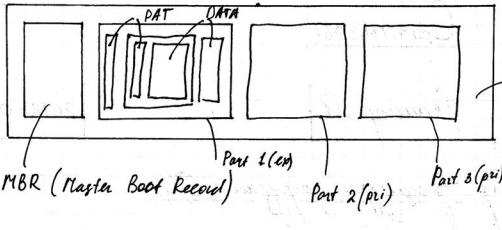
цилиндр



нулевой сектор (служебная информация)

```
Struct Partition {
    char active // признак активности
    char type // тип ФС
    char begin[3] // начало
    char end[3] // конец
    long start // начало
    long length // длина
```





В IBM совместимых машинах под ОС располагаемая на устройствах долговременного хранения (УДХ). Обычно УДХ представляет собой HDD.

Жесткий диск состоит из: вращающихся на одной оси магнитных дисков и синхронно перемещающихся в радиальном направлении головок чтения-записи. Поверхность магнитного диска разделена на концентрические дорожки, каждая из которых разделена на определенное число секторов (длина 512 байт, зависит от форматирования). Совокупность одновременно дорожек в данное время головками называется цилиндром. Наиболее простая система адресации CHS (cilinder, head, sector). Используется по 8 бит для нумерации. Система имеет ограничение: макс. объем адресуемых данных 2,1 Гб. Система содержит информацию в виде блоков, отводимых на адресацию головок. При необходимости адресации большего объема данных весь объем разделяется на логические разделы. Один экземпляр описывает один раздел. Описание разделов происходит в MBR - главной загрузочной записи -, которая находится в 0 секторе и занимает 512 байт

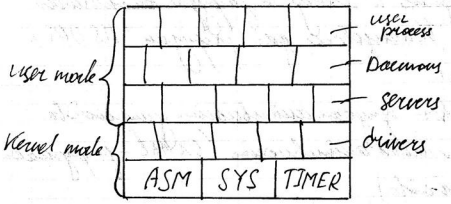
MBR состоит из:

1) PAT - Partitional Allocation Table (содержимое и характеристика структуры Partition)
2) Первичный загрузчик.
   Функции сводятся к поиску и загрузке вторичного загрузчика. Использовался для загрузки MS-DOS и Windows 3.1.
3) Magic number - признак активности устройства. Если стоит метка активности (0xAAZ - загружаем ОС с устройства).

Структурный состав ОС

Современная ОС состоит из набора упорядоченных в иерархическом порядке в зависимости от важности программ. После успешной загрузки ОС эти программы должны находиться в оперативной памяти. ОС состоит из:

1) машиннозависимого ассемблерного кода;
2) набора базовых служебных программ (на С);
3) драйверов - служебных программ обеспечивающих взаимодействие с периферией;
4) серверов - программ, ожидающих запросов от других программ и отвечающих на них.:
   а) Memory Manager (ОП)
   б) Process Manager (менеджер процессов);
   в) File Manager
5) демонов - самовыполняющихся процессов;
6) пользовательских процессов (имеют интерактивный интерфейс).


user process
Daemons
servers
drivers
user mode
Kernel mode
ASM
SYS
TIMER

Обычно системные процессы и драйверы выполнялись в режиме ядра (доступны все ограничения памяти) и составляют непосредственно ОС. Серверы, демоны и непривилегированные процессы выполняются в пользовательском режиме.

Последовательность загрузки ОС.

1) POST
2) загрузка в MBR в ограниченной памяти
3) поиск вторичного загрузчика (ОС сектор раздела).
4) поиск и загрузка в ограниченную систему модуля ОС. Загрузочный модуль состоит из набора программ, набора базовых драйверов, набора базовых серверов и демонов.
5) последовательная инициализация tty.
6) инициализация серверов и демонов
7) загрузка непривилегированных процессов, начиная из набора оболочек пользователя (bash, sh, csh)

Д/з. установить текстовую Linux на виртуальную машину

Основные команды bash

man - справочное руководство

<имя_команды> --help - справка по команде.

which
whereis } нахождение команды.

pwd - путь к папке текущего положения

ls - содержимое каталога.

-al - закрытые файлы и св-ва.

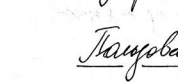
cp <откуда> <куда> - скопировать.

rm <путь+файл> - удалить.

клавиша Tab - позволяет дописывать команду.

cat - вывод файла на экран.

Пользователи Linux



/etc/passwd - файл пользователей.

Пользователи имеют свои числовые идентификаторы uid - 32х битное число. Uid является одним из свойств процессов, что является частью механизма привилегий.

Между Uid и мнемоническими именами существует взаимосвязь. Файл является базой данных.

пользователь : пароль : uid : Guid : опис : домашний каталог : командный интерпретатор.

Содержимое файла passwd используется следующим образом:

1) программа запуска терминала (getty) выводит на экран приглашение к вводу имени пользователя и пароля;
2) программа шифрует пароль, находит строчку описывающую введенного пользователя, сравнивает результаты шифрования паролей, если совпадает, то запускает интерпретатор, присваивает ему Uid и Gid
3) переходит в указанный домашний каталог
4) после этого можно работать с системой

useradd - создать пользователя
user-del - удалить пользователя
passwd - создание/изменение пароля пользователя.

Создание нового пользователя:

1) создание дом-й директории;
2) модификация файла /etc/passwd
3) создание пароля
4) модификация прав доступа к дом-й папке

Для удобства администрирования пользователи могут объединяться в группы. Пользователь может быть в нескольких группах. Инф-я о группах хранится в файле /etc/group.

Файловая система Linux

Виды файлов:

1) обычный файл —
2) каталог d
3) блоко-ориент. устройства b
4) символьно-ориент. устройства c
5) ссылки l
6) мягкие ссылки s
7) именованные каналы p

touch - создать обычный файл.
mkdir - создать каталог
mknod создать устройство
ln - создать ссылку (-s - мягкая ссылка).
pipe - создать канал
unlink - удалить канал.

```
d rwx|-w-|rw-     x x x  → биты смены UID и GID (описание работы см. лекцию №3)
      U   G  Others  SUID SGID Sticky
111  111 111  1u     111
```

16 битовая маска доступа к файла.

chmod – изменение прав доступа

-S- sort - сортировка

Каждый файл имеет определенный набор свойств (ls -l). Эти свойства представляются в битовой маске доступа. БМД состоит из 15 битов, разделенных на 3:

1) тип файла (создается при создании файла)
2) права доступа владельца файла } chmod
3) права доступа членов группы
4) права доступа всех остальных
5) набор случайных битов.

cd /

bin boot etc dev home tmp var

root

Путь к файлу может быть абсолютным или относительным.

. – ссылка на саму себя

.. – ссылка на родителя

Файловая система Linux реализована в стандарте POSIX

/ - корневой каталог и содержит каталоги 1 уровня.

1) bin - исполняемые модули системных программ
2) boot - загрузочный загрузчик
3) etc - общий файлы конфигурации
4) dev - файлы устройств
5) home - домашние папки юзеров
6) tmp - временные файлы
7) var - каталоги программного окружения

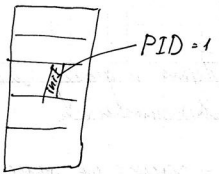
Семинар №3

2012-09-19

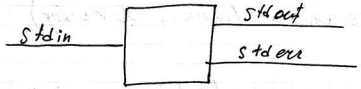
Процессы.

Процесс - программа в момент выполнения.
С процессом связаны следующие идентификаторы:

PID - process id
PPID - parent process id
UID - user id
GID - group id
EUID - effective user id.
EGID - effective group id.



При старте процесса он получает свой процесс ID (PID). PPID обычно равен id предка (вводили команду в bash, то PPID = PID bash). В случае, если род-й процесс завершается, родителем его потомков становится init (PID = 1). При старте процесса UID и GID, а также EUID и EGID становятся равными соответствующим id пользователя, который запустил процесс. В случае если в битовой маске файла процесса установлены биты suid и sgid, то идентификаторы EUid и EGID становятся равными соответствующим ID владельца файла.



PS - средство отображения процессов

ps -aux (-lax)

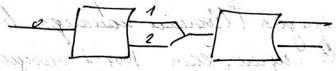
| - создает канал между процессами

запуск в фоне: команда &

less (more) - программа постраничного просмотра

bg - в фон

fg - на передний план.



С процессом связано 3 потока - поток ввода, вывода, стандартный поток ошибок. Если процесс ожидает ввода команд bash : 0 - на клавиатуру, 1,2 - на экран. Такой процесс называется выполняемым на переднем плане. Все остальные процессы называются выполняемыми в фоне.

Потоки процессов можно перенаправлять:

< - перенаправление потока ввода (less < filename)

> - перенаправление потока вывода (ps > filename)

2> - перенаправить ошибки (man ... 2> errlog).

ps - bx - перенаправляет поток ввода другому процессу.

Сигналы

Сигнал - средство управления процессами.
Реализуется через прерывание процесса и вызов обработчика. Обработчики могут быть заданы программистом для обрабатываемых или для необрабатываемых сигналов. Вызываются стандартной системой. Сигнал могут быть посланы:

1) системой;
2) другими процессами;
3) пользователем. (KILL – средство направления сигналов процессов)

Сигналы бывают:

HUP (1) – сброс процесса
INT (2) – прерывание процесса
QUIT (3) – запрос на выход
KILL (9) – принудительное завершение – необрабатываемый
TERM (15) – запрос на останов } ~
STOP (19) – запрос на останов }
TSTP (20) – -//-
CONT (18) – продолжение остановленного процесса.

Драйвер терминала может принимать нажатия клавиш и генерировать некоторые сигналы.

Семинар №4

2012-09-26

Фактор уступчивости – определяет приоритетность процесса.

nice – запустить процесс и назначить приоритетность nice. Старшая защита фр уступи.

renice – перезапустить

Приоритеты процессов.

Процессы в системе выполняются циклически. Каждому процессу выделяется определенный квант процессорного времени.

Чем больше приоритет, тем больше времени выделяется ему.

Чем больше фактор уступчивости, тем меньше приоритет.

Пользователь не может создавать более приоритетные процессы, чем его экземпляр командного интерпретатора.

Для управления приоритетами существуют команды nice; renice.

nice позволяет задавать приоритет при старте процесса

renice позволяет изменять приоритет выполняемого процесса

Командный интерпретатор bash

echo - копирует свой стандартный вход на стандартный выход.

cat - копирует стандартный вход на стандартный выход (позволяет работать с файлами);

cat /etc/inittab.

bash воспринимает:

.

␣ разделение слов

& перевод процесса в фон

| соединение потока ввода 1 с потоком вывода 2.

>, >>, < } перенаправление потоков

`...` - спец. символ, который всегда используется в паре; необходим для вложенных команд

Например: echo `ls`.

'...', "..." } используется для обозначения строк; символы, заключенные в них теряют специальное значение.

*, ? } шаблонные символы;
* - ∀ набор символов
? - ∀ одиночный символ.

( ) - используется для программирования арифметических выражений и обозначает вложенные команды.

{ } - перечисление
например: touch file{1,2,3}

[ ] - аналог ( )

/ - обозначает корневую директорию или разделитель в пути файлов

\ - снимает действие специального значения с следующего символа.

; - разделение команд

~ - домашняя директория пользователя.

$ - позволяет обращаться к переменным.

&& - логическое „и"

|| - логическое „или"

Переменные bash

bash позволяет создавать переменные 3х видов:
1) пользовательские переменные;
2) специальные переменные;
3) позиционные переменные.

Пользовательские переменные создаются путем инициализации произвольного набора символов.

```
var=5
echo $var
```

удаление переменной: unset var.

Различные программы, например bash, используют пользовательские переменные

set - просмотреть полный набор пользовательских переменных.

Наиболее часто используемые локальные переменные:

PATH - строка поиска; содержит имена каталогов, разделенные :

EDITOR - используется по умолчанию как редактор файлов.

HOME - содержит значение домашней директории пользователя.

pwd - хранит значение текущего каталога

SHEL - содержит строку запуска домашнего интерпретатора пользователя

Создаваемые в bash переменные подчиняются в локальную область видимости.
Для обеспечения глобальной видимости переменной нужен ее экспорт.

Специальные переменные:

? - содержит статус выхода предыдущего процесса.

- содержит десятичное значение количества позиционных параметров

Позиционные переменные используются для передачи пар-в в скрипт

Файлы сценариев

bash позволяет программировать несложную логику. Составляющие команды могут быть записаны в одну строчку или текстовой файл, которому необходимо присвоить права на выполнение. Такой файл называется файлом сценария или скриптом.

- .profile
- .bashrc

Файл скрипта должен начинаться с директивы
#: /bin/bash, которой указывает адрес обработчика.
Внутри скрипта ";" ~ переход на новую строчку.
#! /bin/bash - директива

Порядок обработки командной строки

1) раскрытие скобок { };
2) замена имени ~;
3) подстановка значений переменных;
4) постановка команд; ` ` $( )
5) вычисление арифметики; $(( )); $[ ]
6) разделение слов;
7) раскрытие шаблонов

имя файла
chmod ### filename — права на выполнение.

Семинар №5 | 2012-10-03

Логические операторы bash

test - программа проверки.

test -f filename

[-f filename]

echo $?

Условные сокращения

list - набор команд
name - пользовательская переменная
word - интерпретируемое значение переменной

if list then list [elif list then list]
else list fi - оператор условного выбора.

case word in [(] pattern) list ;;] esac - оператор множественного выбора.

select name [in word] do list done - оператор интерактивного взаимодействия

for name [in word] do list done - счетный цикл

while list 1 do list 2 done - предусловие
until list 1 do list 2 done - постусловие

В цикле for не происходит автоматической модификации счетчика.

Семинар №6 | 2011-10-10

1) Создать скрипт, переносящий принимать 2 пар-ра и выводящий их сумму;
2) Создать скрипт калькулятор. На вход 2 операнда, предлагает выбрать операцию: +, -, *, :. Выводит ответ;
3) Создать скрипт генерации log-файлов. Создается log архив директории. После создания 4 файла архив. 1 архив удаляется.

Семинар №7 | 2012-10-17

РК

Вариант №2

Семинар №8 | 2012-10-24

Создание секретного ключа

openssl genrsa [-out file] [des/-tes/idea] -rand file] bits.

Создание публичного ключа на основе частичного.

openssl rsa -in filename [out filename] [-des/des3/idea] -pubout.

Шифрование и дешифрование файла симметр. ассиметрии

openssl des3 -in file out file des3
openssl des3 -d -in file des3 -out file

Вычисление симметрической цифры:

openssl mods -e file

Подпись файла секретным ключом:
openssl mdi -sign private.key -out signature -hexsig file [S].

Проверка подписи файла

openssl md5 - signature -verify public.key file [s].

Шифрование файла public

open ssl rsautl in filename -out file.cr -key in public.key -public encrypt.

Шифрование файла секретным ключом.

openssl rsautl in file.cr. -out file -key in privatekey -decrypt.

Язык программирования Си

gcc -o program source.c

Программа на языке C состоит из функций и переменных

Для компиляции исходного кода используется компилятор gcc

Для обеспечения сборки программ, составляемых из исходных кодов (файлов) служ-т утилита автоматизированной компиляции.

Процедура make будучи запущенной без параметров производит поиск в текущей директории файла с именем „makefile", в котором описываются действия необходимые для компиляции программ по частям.

Makefile состоит из целевых сводок. Каждая сводка описывает атомарную последовательность действий.

Формат целевых сводок:

Имя : source |→
→ команда

Пр: all : source.c |→
→ gcc -o pr source.

clean |→
→ - rm *.o *.bak } для удаления необязательных файлов.

Язык С преобразует перенос, пустые операторы (всего 3х)
; - обозначает конец оператора.

Переменные языка С

int - целое число;
char - символ (8 б);
float - вещественные числа одинарной точности;
double - вещественные двойной точности;
void - пустой тип данных.

Модификаторы

long - регистрает в 2 раза область допустимых значений;
short - уменьшает в 2 раза базовый тип;
extent - внешние переменная;
static - размещение переменной в обл. памяти;
register - обращение компилятору, что пер-ую в регистр

const - объявляет неизменяемую пер-ую
volatile - запрещает оптимизацию переменной вызов функции
unsigned - беззнаковый тип данных.

Виды переменных C:

1) локальные - область видимости ограничена {}
2) глобальные - доступны из всех областей программы.
3) формальные - параметры используемые для передачи значений в функцию.

Пример программы:

```
1) <# include. cstdio.h >
2) int main (void)
3) {
4)     printf ("Hello, world \n")
5)     return 0;
   }
```

1) библиотека, находящаяся в директории libraru и содержит разные функции, наборы расширеных возможностей.
2) объявление функции;
3) тело функции ограничено операторными скобками;
4) при вызове в функцию передаёт объявленные формальные параметры;
5) оператор возврата значения функции.

Семинар №9 | 2012-11-07

Объявление константы

"A" - буква
'A' - число : код - по порядку.

Специальные константы

EOF - конец файла
EOL - конец строки.

Константы, символы.

\b bell
\t tab
\n new line
\h hop
\0 oct (восьмеричное счисление)

c = 'i'm a string';
char c[50] = "I'm a string";

Операторы

1) Математические:

* / + - %

2) Логические

> < == != >= <=

&& - И
|| - ИЛИ

3) присваивание

$=$ $\quad -=$ $\quad +=$ $\quad \ldots$

$i = i+1 \equiv i{+}{+}$

$i = i-1 \equiv i{-}{-}$

Операторы инкремента и декремента имеют префиксную и постфиксную форм.

Управляющие конструкции.

1) if ( )

else

2) switch ( )

case :

case :

—//—

default:

3) счётный цикл

for (начало ; проверка ; мод.)

...

4) while (выражение)
instruction.

Семинар №10 | 2012 - 11 - 14

Функции.

Функции - как и любые ресурсы должны быть описаны до использования - объявить функцию.

тип_возвр._знач. имя_функции (типы форм. пар-в)

| Ресурсы | | main | | польз. не функции |

В случае если, польз. не функции определены до main, приводить их объявление в поле ресурсов не нужно.

Объявление функции имеет цель: согласовать тип возвр. пар-ра, типы и кол-во входных пар-в. Имена формальных пар-в не согласуются.

Определение функции - описание логики её работы.

тип_возвр.-знач. имя (типы пар-в) {
алгоритм
}

Функции не могут влиять на переменные, переданные им в кач-ве формальных пар-в, т.к. вызовы локальные в функции.

Для того чтобы иметь возможность влиять на глоб. переменные, то передавать указатели на переменные или массивы.

Рекурсия

Язык "си" позволяет вызывать функцию из функции, в том числе и саму себя.

Рекурсия – это многократный вызов функцией самой себя.

сортировка "быстрая"

Семинар №11	2012-11-21

Указатели

Это переменная, которая содержит в себе адрес другой переменной.

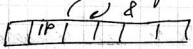
Объявление:

```
int *ip, i;
```

указатель — переменная

Указатели запрещается инициализировать каким-либо числом.

Операции с указателями



i = 10

ip = &i

С указателями разрешается производить арифметические операции.

Увеличение указателя на 1 => перемещение указателя на след. ячейку.

```
printf ("%d", i);
printf ("%d", *ip);
```

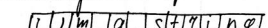
* - переход по значению;

& - взятие адреса.

При работе с указателями мы можем работать с 2-мя операторами: * и &.

В языке "С" массивы имеют одно очень важное св-во: они располагаются в памяти строго последовательно.

```
char * mesp, mes;
mes = "i'm a string";
mesp = * mes;
mesp = &mesp +1.
```



a[i] <=> * (ip +1) ____ адреса арифметика.

Для работы с матрицами можно использовать комбинирование методов.

Операторы & и * имеют приоритет, они вычисляются в первую очередь.

```
int strlen (char *);
```

Пр.

```
#include <stdio.h>
int strlen (char *s)
```

```
{
  int a=0;
  for (a=0 ; *s != '\0' ; s++)
      a++;
  return a;
}
void main()
{ int l;
  l = strlen("123");
printf("%d", l);
}
gcc -o prog s1.c
./prog.
```

Семинар № 12 | 2012-11-28

include

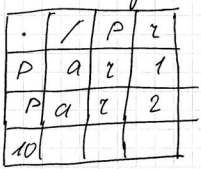
main

func.

(int) main (int argc, char **argv

argument counter — argument vector

Пр.




```
int mas[ ];
   mas[i] <=> *(mas fi)
   argv [i][j]
   *(argv+j)[i]
   *(*(argv+j)+i)
```

argument counter. содержит количество элементов

Доступ к файлам

Для работы с файлами необходимо объявить файловую переменную. Для открытия файла необходимо создать привязку.

```
FILE *fp;
FILE *fopen (char *name; char *mode),
fp = fopen ("file test", "r");
fgetc (FILE*)
fputc (FILE*, char)
fprintf (FILE*, char* format, variable)
fscanf (FILE*, char* format, varep)
Удаление файлов unlink
```