UNIX System Programming

Lecture 10: Data Management

• Outline
  • Managing Memory
  • File Locking
  • The gdbm Database Routines

• Reference
  • BLP: Chapter 7
  • man pages: malloc, calloc, fcntl, flock, lockf, lslocks, gdbm

Lecture 10: Data Management

Managing Memory

• The malloc() routine is similar to malloc() but initializes the allocated memory to _____.

• realloc() can resize the memory block from an earlier call to malloc().

• The container classes from the STL can be resized and are recommended for use whenever possible. They are simpler to use than malloc(), calloc(), and realloc().

Lecture 10: Data Management

Managing Memory

• C uses malloc() and free() to allocate and free memory from the _________. While these routines can be used in C++, you should use new and delete instead:

```c
// C dynamic memory allocation
float *farray = (float *) malloc(10000*sizeof(float));
free(farray);

// C++ dynamic memory allocation
float *farray = new float[10000];
delete[] farray;
```

• Memory pointers returned by new, malloc(), etc are pointers to virtual memory and are not _________ memory addresses.

• UNIX provides process memory protection. Memory allocated to one process can not be read/written by another process. This is enforced in hardware on most CPU architectures.

Lecture 10: Data Management

Exclusive Access

• Lock files can be used as _________ to ensure exclusive access to a file:

```c
while (fd = open("/tmp/stu_rec.sck",
    O_RDWR | O_CREAT | O_EXCL, 0644)) == -1)
    sleep (10); // lock held by other process
close(fd); // ok, we've got the lock
// Access the data
datafd = open("studentrecords.dat", O_RDWR);
// DO SOMETHING WITH DATA HERE
// close(datafd);
unlink("/tmp/stu_rec.sck"); // release lock
```

• The O_CREATE | O_EXCL flags ensure that the open() call fails if the LOCK file exists.

• This same method can be used to guarantee exclusive access to hardware. It is often used for exclusive access to the serial ports. (Lock files are in /var/lock - see the FSSTND for naming conventions)

• Note that this is a cooperative arrangement only (aka _________ locking).
Lecture 10: Data Management
Exclusive Access

- Lock files are fine for exclusive access to hardware or to small files (for short periods of time). For large, __________ oriented (random access or database) files that may be accessed simultaneously by many processes we need another method.
- fcntl() allows locking of only a section of a file. (fcntl() has a number of uses, we will discuss only file locking here.)

Lecture 10: Data Management
Exclusive Access

There may be multiple read (shared) locks, but only one write (exclusive) lock on an area. You can not get a write lock if there is a read lock on a section or vice versa.

- The OS read() and write() routines should be used with fcntl() locking. (avoid C or C++ routines which ____________ I/O.)
- Advisory locking is the default. Use Islocks to list system file locks.

Lecture 10: Data Management
Exclusive Access

```c
struct flock region;
struct datarec active;
int recsize = sizeof(active);

fd = open("database.dat", O_RDWR);

// Request write lock on a record
region.l_type = F_WRLCK;
region.l_whence = SEEK_SET;
region.l_start = recnum*recsize;
region.l_len = recsize;
val = fcntl(fd, F_SETLK, &region);
// MUST CHECK VAL TO SEE IF LOCK SUCCESSFUL
// CONTINUED ON NEXT SLIDE
```

Lecture 10: Data Management
Exclusive Access

```c
// Read record into memory
nb = lseek(fd, recnum*recsize, SEEK_SET);
nr = read(fd, &active, recsize);

// Process and modify record here

// Write modified record back
nb = lseek(fd, recnum*recsize, SEEK_SET);
nw = write(fd, &active, recsize);

// Release the write lock
region.l_type = F_UNLCK;
val = fcntl(fd, F_SETLK, &region);
```

Lecture 10: Data Management
gdbm Database Routines

- You can use F_SETLK/W instead of F_SETLK in fcntl() to ______ until a lock is available.
- The lockf() routine provides a simpler interface to fcntl() for file locking. It locks a region starting at the current file pointer:
  lockf(fd, F_LOCK, recsize);
- Most UNIX OSes (Linux) also provide flock() which can be used to lock an entire file.

Lecture 10: Data Management
gdbm Database Routines

- We have seen that files can be used for data storage, so why use a database?
- Databases allow storage of data records of different ________
- Databases use an index to store and retrieve data. The index does not have to be a simple number, but may be an arbitrary string.
Lecture 10: Data Management
gdbm Database Routines

- We will use the _______ database routines. The gdbm library also provides routines for compatibility with dbm and ndbm databases. (The text discusses ndbm.)
- To use the gdbm routines, include the gdbm.h header file in your C/C++ source code and link with gdbm library (pass the -lgdbm option to gcc/g++).

Lecture 10: Data Management
gdbm Database Routines

- The database routines use a datum struct for both _______ and content. The datum struct contains members:
  char *dptr;
  size_t dsize;
- There are four main routines: gdbm_open(), gdbm_store(), gdbm_fetch(), and gdbm_close().

Lecture 10: Data Management
gdbm Database Routines

- gdbm_open() returns a GDBM_FILE type that is passed as an _______ to the other routines. Only a single writer can open a gdbm file. Multiple readers may open it.
- gdbm_store() is used to enter data into the database.
- See the dbmwrite.cpp program.

Lecture 10: Data Management
gdbm Database Routines

- gdbm_fetch() is used to _________ data from the database.
- gdbm_firstkey() and gdbm_nextkey() can be used to scan the entire database.
- Refer to man gdbm and info gdbm pages for complete documentation.
- See the dbmscan.cpp program.

Lecture 10: Data Management
gdbm Database Routines

- Download, compile and run dbmwrite.cpp and dbmscan.cpp.
- Write a dbmread.cpp program to read and display the employee and salary information for the employee with id 43210.