4. Reading and writing Disk blocks

Reading a disk block

```
Struct buf *
bread(uint dev, uint sector)
{
  struct buf *b;
  b=bget(dev.sector);
  if(!(b->flags & B_valid))
    iderw(b);
}
```

Writing a disk block

```
void bwrite(struct buf *b)
{ if(b->flags & B_BUSY)==0)
    panic("bwrite");
    b->flags |= B_DRTY;
    iderw(b);
 }
```

```
$ rm a
log-write 29 write i
log-write 4 iupdate
log-write 28 btree
log-write 4 iupdate
log-write 4 iupdate
```

algorithm read

```
get file table entry from user file descriptor;
check file accessibility ;
set parameters in u area for user address,
byte count, I/O to user;
get node from file table;
lock node;
set byte offset in u area from file table offset;
while (count not satisfied)
{ convert file offset to disk block (algorithm bmap)
  Calculate offset into block, no of bytes to read;
  if ( no of bytes to read is 0)
      break;
  read block (algorithm breada if with read ahead,
              algorithm bread otherwise);
  Copy data from system buffer to user
  address;
  update u area fields for file byte offset,
  read count, address to write into user space;
  release buffer;
}
unlock node;
update file table offset for next read;
return ( total no of bytes read);
```