Announcements

- Project #2 is available on the web
  - Atoi.c has been posted
  - Reference solution to project #1, delete loop in Print_String function inside libuser.c
- Midterm #1 was returned
  - Solution on web
  - Must submit requests for re-grades **in writing** by 3/19/02
  - Grade breakdown (stddev 13.2)

	1	2	3	4	5	6	Total
Avg	12	8	5	13	9	9	56.2
Min	3	0	0	4	0	4	30
Max	20	10	20	15	20	15	84

Sharing Memory

- Pages can be shared
  - several processes may share the same code or data
  - several pages can be associated with the same page frame
  - given read-only data, sharing is always safe
- when writes occur, decide if processes share data
  - operating systems often implement “copy on write” - pages are shared until a process carries out a write
    - when a shared page is written, a new page frame is allocated
    - writing process owns the modified page
    - all other sharing processes own the original page
  - page could be shared
    - processes use semaphores or other means to coordinate access

What Happens when a virtual address has no physical address?

- called a *page fault*
  - a trap into the operating system from the hardware
- caused by: the first use of a page
  - called *demand paging*
  - the operating system allocates a physical page and the process continues
  - read code from disk or init data page to zero
- caused by: a reference to an address that is not valid
  - program is terminated with a "segmentation violation"
- caused by: a page that is currently on disk
  - read page from disk and load it into a physical page, and continue the program
- causde by: a copy on write page

OS Protection attributes (Win32)

- NOACCESS: attempts to read, write or execute will cause an access violation
- READONLY: attempts to write or execute memory in this region cause an access violation
- READWRITE: attempts to execute memory in this region cause an access violation
- EXECUTE: Attempts to read or write memory in this region cause an access violation
- EXECUTE_READ: Attempts to write to memory in this region cause an access violation
- EXECUTE_READ_WRITE: Do anything to this page
- WRITE_COPY: Attempts to write will cause the system to give a process its own copy of the page. Attempts to execute cause access violation
- EXECUTE_WRITE_COPY: Attempts to write will cause the system to give a process its own copy of a page. Can’t cause an access violation

Handling a page fault

1) Check if the reference is valid
   – if not, terminate the process
2) Find a page frame to allocate for the new process
   – for now we assume there is a free page frame.
3) Schedule a read operation to load the page from disk
   – we can run other processes while waiting for this to complete
4) Modify the page table entry to the page
5) Restart the faulting instruction
   – hardware normally will abort the instruction so we just return from the trap to the correct location.