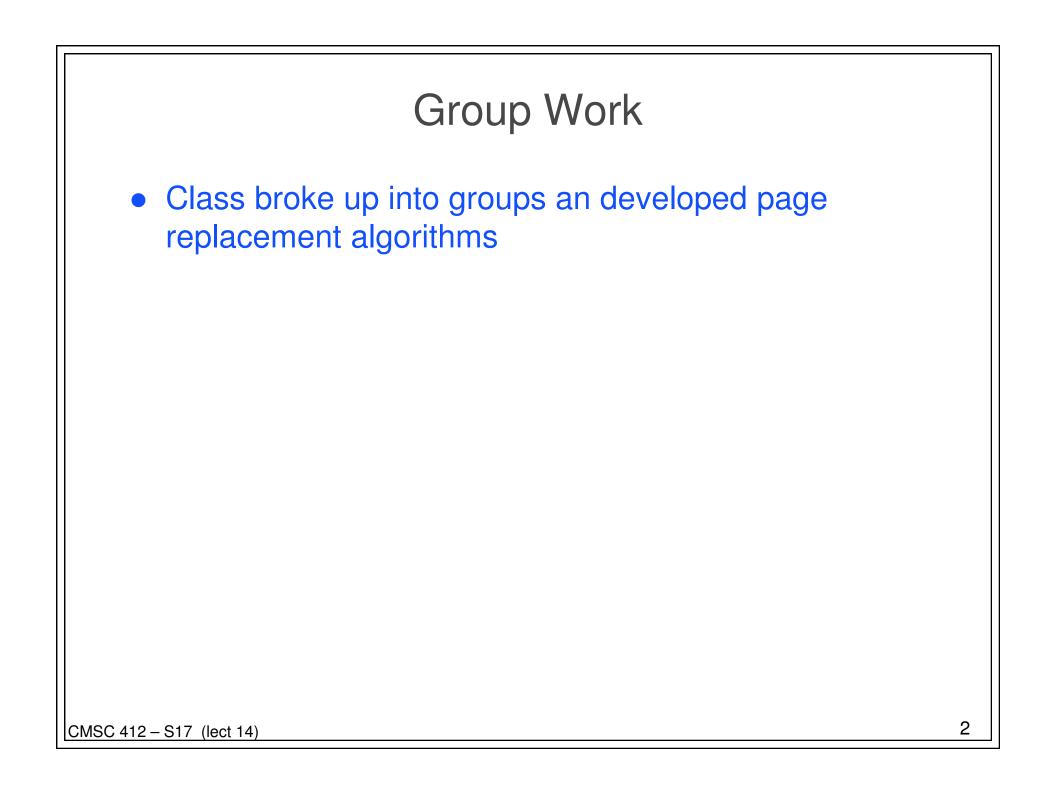# Announcements

- Project #4 teams emailed out

- Git groups setup

- Reminder about re-grade deadline
    - Thursday March 30th (11:00 AM)

- Reading Chapter 10 (in 8th Ed)

Group Work

- Class broke up into groups an developed page replacement algorithms

What happens when we fault and there are no more physical pages?

- **Need to remove a page from main memory**
  - if it is "dirty" we must store it to disk first.
    - dirty pages have been modified since they were last stored on disk.

- **How to we pick a page?**
  - Need to choose an appropriate algorithm
    - should it be global?
    - should it be local (one owned by the faulting process)

Page Replacement Algorithms

- FIFO
  - Replace the page that was brought in longest ago
  - However
    - old pages may be great pages (frequently used)
    - number of page faults may increase when one increases number of page frames (discouraging!)
      - called belady's anomaly
      - 1,2,3,4,1,2,5,1,2,3,4,5 (consider 3 vs. 4 frames)
- Optimal
  - Replace the page that will be used furthest in the future
  - Good algorithm(!) but requires knowledge of the future
  - With good compiler assistance, knowledge of the future is sometimes possible

Page Replacement Algorithms

- ## LRU
  - Replace the page that was actually used longest ago
  - Implementation of LRU can be a bit expensive
    - e.g. maintain a stack of nodes representing pages and put page on top of stack when the page is accessed
    - maintain a time stamp associated with each page

- ## Approximate LRU algorithms
  - maintain reference bit(s) which are set whenever a page is used
  - at the end of a given time period, reference bits are cleared