Quick Logistics
Project 1: Team formation

• Teams of three

• Enter at my.wpi.edu until Tuesday class time, 2pm.

• After that, we’ll assign partners to teams.
BS/MS Credit

- Do project1 on your own.

- Go to my.wpi.edu and enter yourself as one-person team (put a note re BS/MS)

- Use BS/MS Form (even if not yet enrolled).
Lecture #4 – part 1

Review

Chapter 2.3: Merge-Sort Algorithm
Using secondary storage effectively

- General Wisdom:
  - I/O costs dominate
  - Design algorithms to reduce I/O
Two-Way External Merge Sort

- Costs for pass: \(2 \times N\)
- \# of passes: height of tree
- total cost: product of above
External Merge Sort

- What if we had more buffer pages?
- How do we utilize them wisely?

The two main ideas are?
Phase 1

- Construct as large as possible starter lists.
Phase 2

- Compose as many sorted sublists into one long sorted list.
Example

• Buffer : with 5 buffer pages
• File to sort : 108 pages

  – Pass 0:
    • Size of each run?
    • Number of runs?

  – Pass 1:
    • Size of each run?
    • Number of runs?

  – Pass 2: ???
Example

- **Buffer**: with 5 buffer pages
- **File to sort**: 108 pages
  - Pass 0: \[\lceil 108 / 5 \rceil = 22\] sorted runs of 5 pages each (last run is only 3 pages)
  - Pass 1: \[\lceil 22 / 4 \rceil = 6\] sorted runs of 20 pages each (last run is only 8 pages)
  - Pass 2: 2 sorted runs, 80 pages and 28 pages
  - Pass 3: Sorted file of 108 pages

- **Total I/O costs**: ?
Example

• Buffer : with 5 buffer pages
• File to sort : 108 pages
  – Pass 0: \( \lceil \frac{108}{5} \rceil = 22 \) sorted runs of 5 pages each (last run is only 3 pages)
  – Pass 1: \( \lceil \frac{22}{4} \rceil = 6 \) sorted runs of 20 pages each (last run is only 8 pages)
  – Pass 2: 2 sorted runs, 80 pages and 28 pages
  – Pass 3: Sorted file of 108 pages

• Total I/O costs: \( 2 \times N (4) \)
Double Buffering (Useful here)

- To reduce wait time for I/O request to complete, can *prefetch* into `shadow block`.
  - Potentially, more passes; in practice, most files *still* sorted in 2 or at most 3 passes.
Lecture #4 – part 2

Pointer to Chapter 2.4: Improving Access Times of Secondary Storage: *Five Disk Optimizations*
Up to now we assume:

- One single disk
- Blocks are chosen randomly from disk to be read in
Five Optimizations (in controller or O.S.)

• Group blocks accessed together on same Cylinder
• One big disk → several smaller disks
• Mirror disks → multiple copies of same data
• Prefetch blocks into main memory → (e.g., double-buffering.)
• Disk Scheduling Algorithms → to select order in which several blocks will be read or written (e.g., elevator algorithm)
Assessment of Five Optimizations

• Effect for “regular predictable tasks”,
  – like one long dedicated process with sequential read
  – e.g., a database SORT (2-phase-multi-way-sort)

• Effect for many “unpredictable irregular tasks”
  – like many short processes in parallel
  – e.g., airline reservations

• Or, some mixture in workload ...
Five Optimizations : Useful or Not ?

- Group blocks together on same Cylinder
- One big disk → several smaller disks
- Mirror disks → multiple copies of same data
- Prefetch blocks → e.g., double-buffering.
- Disk Scheduling → e.g., elevator algo
Assessment of Five Optimizations

• Book has in-depth answer to this assessment.
• So read the book (chapter 2.4).
• Also, question 1 in homework 1 has you study this question.
Lecture #4 – part 3

Storage Layout:
How to lay out data on disk.
( chapter 3)
Overview

Data Items

- Records
- Blocks
- Files
- Memory
What are the data items we want to store?

- a salary
- a name
- a picture

What we have available: Bytes

\[\text{8 bits}\]
To represent:

- Integer (short): 2 bytes
  e.g., 35 is

  00000000  00100011
To represent:

- Boolean
  
e.g., TRUE  1111 1111
  FALSE  0000 0000

- Application specific
  
e.g., RED → 1  GREEN → 3
  BLUE → 2  YELLOW → 4  ...

⇔ Can we use less than 1 byte/code?

Yes, but only if desperate...
To represent:

- Characters
  
  → various coding schemes suggested (ascii)

Example:

A: 1000001
a: 1100001
5: 0110101
LF: 0001010
To represent: String of characters

- Null terminated
  e.g.,
  
- Length given
  e.g.,
  
- Fixed length
  e.g., in Oracle define the string length.
Key Points

• Fixed length items

• Variable length items
  - usually length given at beginning

• Type of an item: Tells us how to interpret (plus size if fixed)
Overview

Data Items
  ↓
  Records
  ↓
  Blocks
  ↓
  Files
  ↓
  Memory
Record - Collection of related data items (called FIELDS)

E.g.: Employee record:
    name field,  
    salary field, 
    date-of-hire field, 
    ...
Types of records:

• Main choices:
  – FIXED vs VARIABLE FORMAT
  – FIXED vs VARIABLE LENGTH
A SCHEMA contains information such as:
- # fields (attributes)
- type of each field (length)
- order of attributes in record
- meaning of each field (domain)
- constraints (primary key, etc).

Not associated with each record.
Example: fixed format and length

Employee record
(1) E#, 2 byte integer
(2) E.name, 10 char.
(3) Dept, 2 byte code

We can simply concatenate fields.
More on Record Layout next time
Reminder:
Homework 1 out
Topic: all about chapter 2.
Due: Next Friday (in class)

Reminder:
Read chapter 3 in textbook