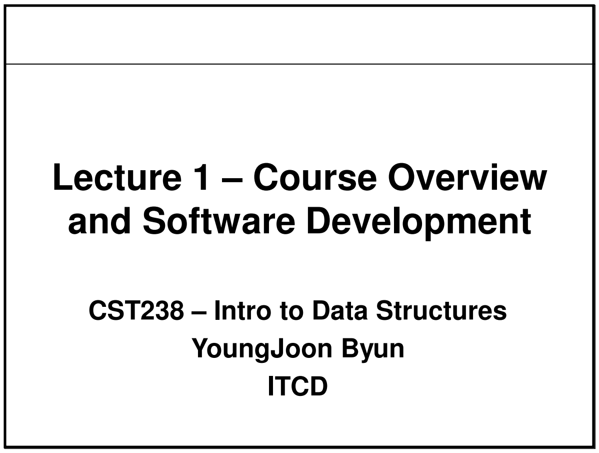
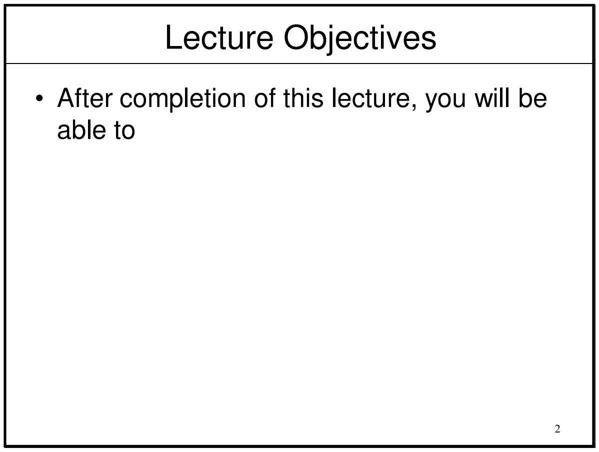


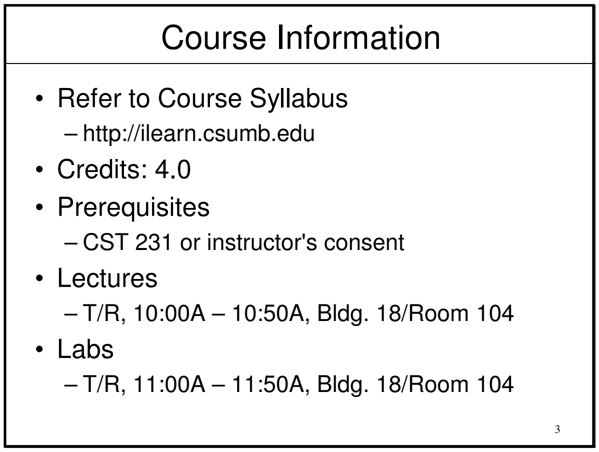
Lecture 1 – Course Overview and Software Development

CST238 – Intro to Data Structures
YoungJoon Byun
ITCD



Lecture Objectives

- After completion of this lecture, you will be able to



Course Information

- Refer to Course Syllabus
  - http://ilearn.csumb.edu
- Credits: 4.0
- Prerequisites
  - CST 231 or instructor's consent
- Lectures
  - T/R, 10:00A – 10:50A, Bldg. 18/Room 104
- Labs
  - T/R, 11:00A – 11:50A, Bldg. 18/Room 104

Instructor

- YoungJoon Byun, Ph.D.
- Email: ybyun@csumb.edu
- Phone: (831)582-3602
- Office: Buld.18/Rm.134
- Office hours:
  - Monday 2:00P – 3:00P
  - Tuesday 1:00P – 2:00P
  - Wednesday 2:00P – 3:00P
  - Or by appointment

Email Contact

- Email is always welcome.
  - Put **[CST238]** at the beginning of subject line
- Sample

To: ybyun@csumb.edu

Subject: [CST238] Homework1 solution?

Content:

Dr. Byun,

At the class, you said that you posted the solution of homework 1 on the iLearn. But I can't find it. Where is the solution?

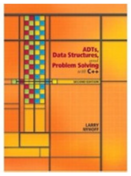
[Your name]

Course Web Site

- http://ilearn.csumb.edu/
  - Additional course information and announcements will be posted here.
  - Check frequently.

Required Textbook

- Author: Larry Nyhoff
- Title: ADTs, Data Structures, and Problem Solving with C++, 2nd Edition
- Publisher: Prentice-Hall, 2005
- ISBN: 0-13-140909-3

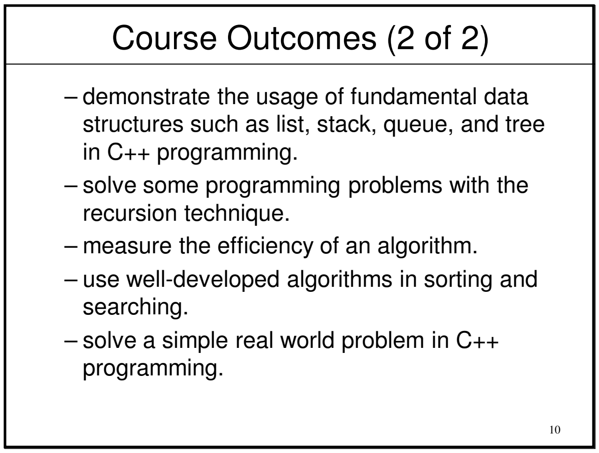


Main Objectives

- In this class, you will acquire **intermediate-level programming skills** to write clear, efficient, and high-quality code.
  - You will learn **basic data structures and algorithms**: array, list, stack, queue, tree, searching, sorting, recursion, algorithm analysis.
  - You will have **in-depth knowledge of C++** programming language.

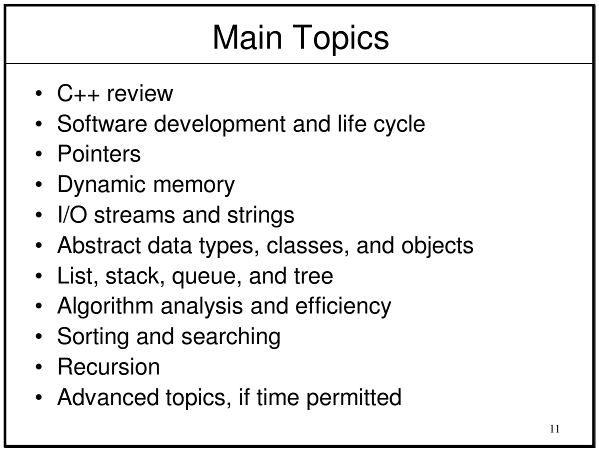
Course Outcomes (1 of 2)

- After finishing this course, you should be able to
  - explain the software development life cycle and software development models.
  - define and use pointers in C++ program.
  - process data using dynamic memory allocation.
  - demonstrate the usage of I/O streams and strings in C++ program.
  - define and use programmer-defined classes.



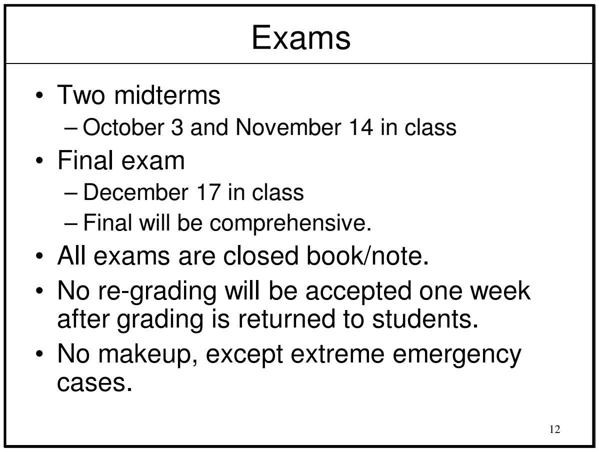
Course Outcomes (2 of 2)

- demonstrate the usage of fundamental data structures such as list, stack, queue, and tree in C++ programming.
- solve some programming problems with the recursion technique.
- measure the efficiency of an algorithm.
- use well-developed algorithms in sorting and searching.
- solve a simple real world problem in C++ programming.



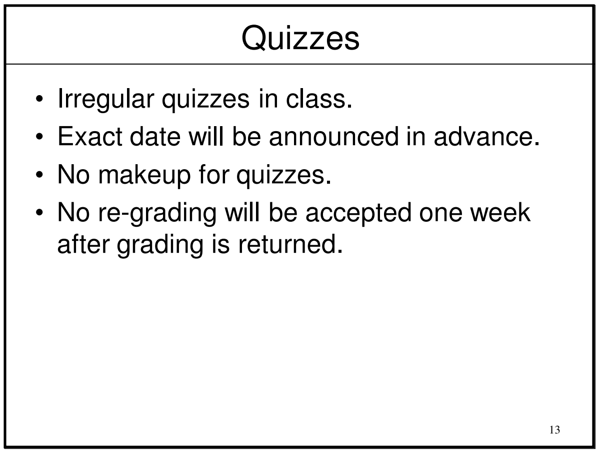
Main Topics

- C++ review
- Software development and life cycle
- Pointers
- Dynamic memory
- I/O streams and strings
- Abstract data types, classes, and objects
- List, stack, queue, and tree
- Algorithm analysis and efficiency
- Sorting and searching
- Recursion
- Advanced topics, if time permitted



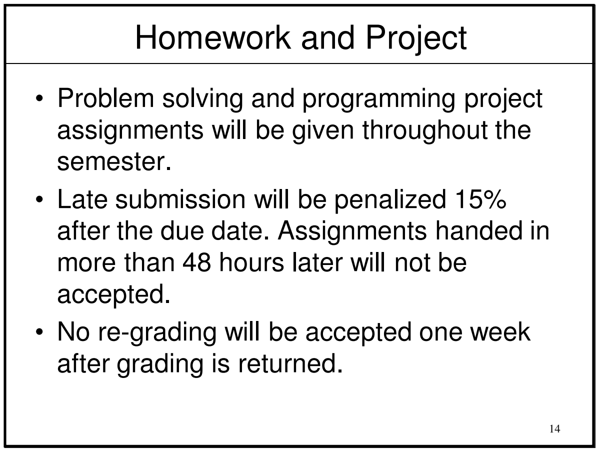
Exams

- Two midterms
  - October 3 and November 14 in class
- Final exam
  - December 17 in class
  - Final will be comprehensive.
- All exams are closed book/note.
- No re-grading will be accepted one week after grading is returned to students.
- No makeup, except extreme emergency cases.



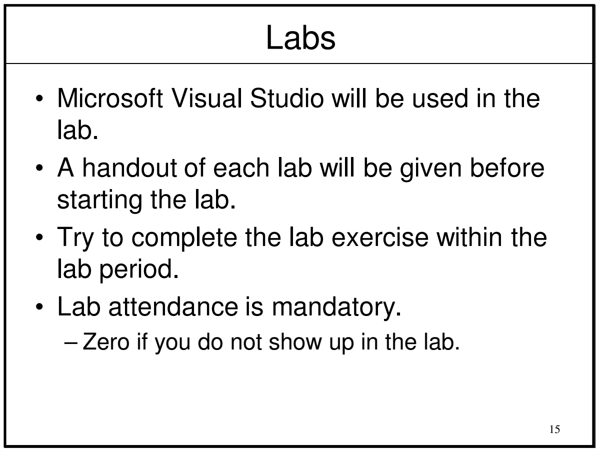
Quizzes

- Irregular quizzes in class.
- Exact date will be announced in advance.
- No makeup for quizzes.
- No re-grading will be accepted one week after grading is returned.



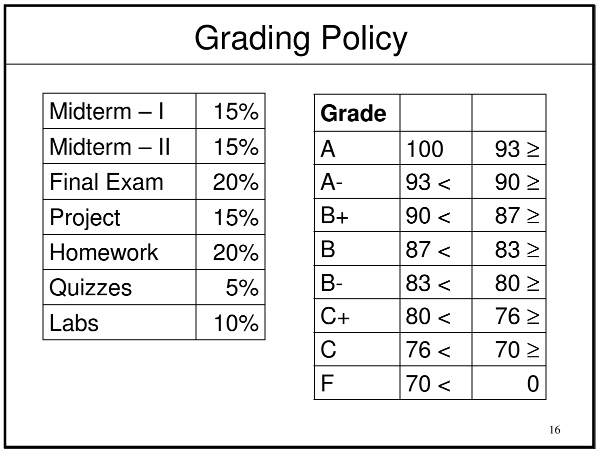
Homework and Project

- Problem solving and programming project assignments will be given throughout the semester.
- Late submission will be penalized 15% after the due date. Assignments handed in more than 48 hours later will not be accepted.
- No re-grading will be accepted one week after grading is returned.



Labs

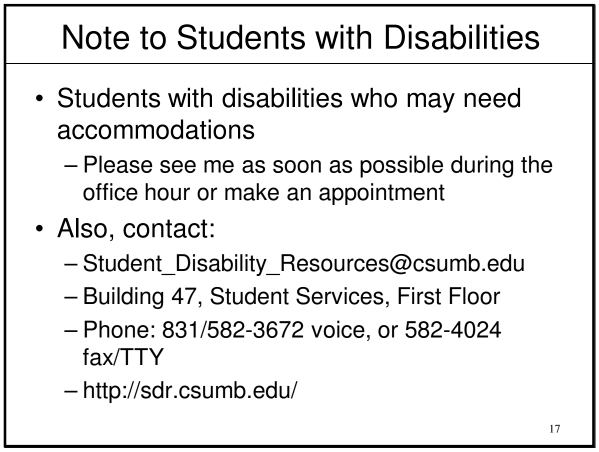
- Microsoft Visual Studio will be used in the lab.
- A handout of each lab will be given before starting the lab.
- Try to complete the lab exercise within the lab period.
- Lab attendance is mandatory.
  - Zero if you do not show up in the lab.



Grading Policy

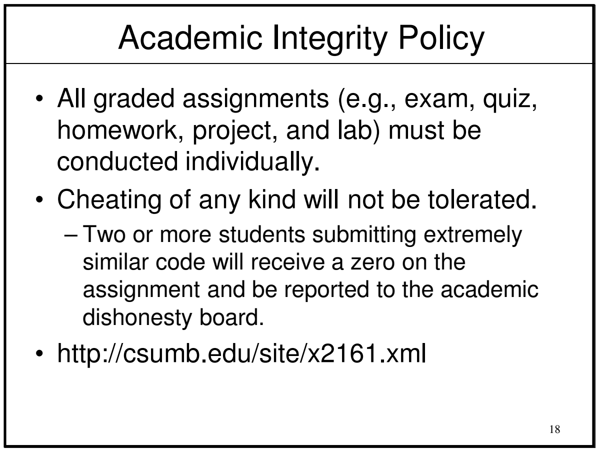
Midterm – I	15%
Midterm – II	15%
Final Exam	20%
Project	15%
Homework	20%
Quizzes	5%
Labs	10%

Grade		
A	100	93 ≥
A-	93 <	90 ≥
B+	90 <	87 ≥
B	87 <	83 ≥
B-	83 <	80 ≥
C+	80 <	76 ≥
C	76 <	70 ≥
F	70 <	0



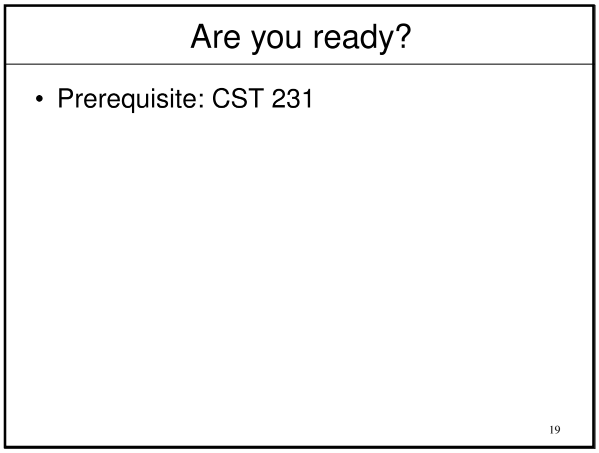
Note to Students with Disabilities

- Students with disabilities who may need accommodations
  - Please see me as soon as possible during the office hour or make an appointment
- Also, contact:
  - Student_Disability_Resources@csumb.edu
  - Building 47, Student Services, First Floor
  - Phone: 831/582-3672 voice, or 582-4024 fax/TTY
  - http://sdr.csumb.edu/



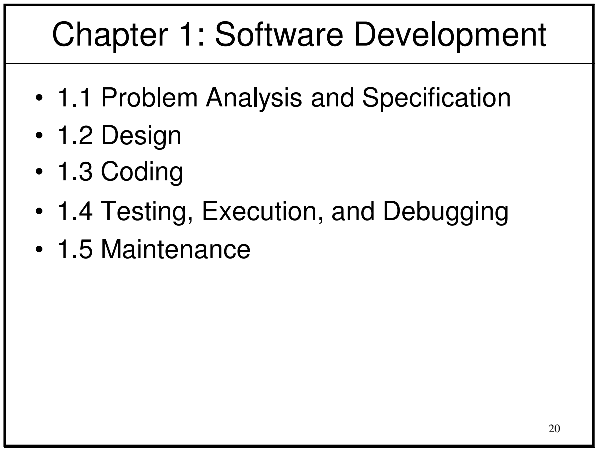
Academic Integrity Policy

- All graded assignments (e.g., exam, quiz, homework, project, and lab) must be conducted individually.
- Cheating of any kind will not be tolerated.
  - Two or more students submitting extremely similar code will receive a zero on the assignment and be reported to the academic dishonesty board.
- http://csumb.edu/site/x2161.xml



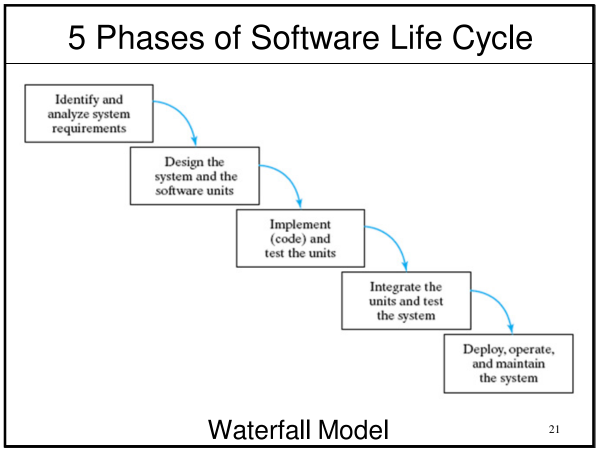
Are you ready?

- Prerequisite: CST 231



Chapter 1: Software Development

- 1.1 Problem Analysis and Specification
- 1.2 Design
- 1.3 Coding
- 1.4 Testing, Execution, and Debugging
- 1.5 Maintenance



5 Phases of Software Life Cycle

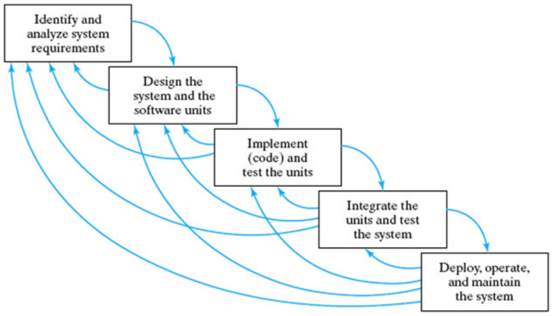
Identify and analyze system requirements
Design the system and the software units
Implement (code) and test the units
Integrate the units and test the system
Deploy, operate, and maintain the system

Waterfall Model

Typical Software Life Cycle in Your Project

1. Problem Analysis and Specification
   - Read project description very carefully.
2. Design
   - Pencil and paper work.
3. Coding
   - Programming
4. Testing, Execution, and Debugging
   - Your instructor may provide some test cases.
   - You may also make your own test cases.
5. Maintenance

Realistic Waterfall Model



Identify and analyze system requirements

Design the system and the software units

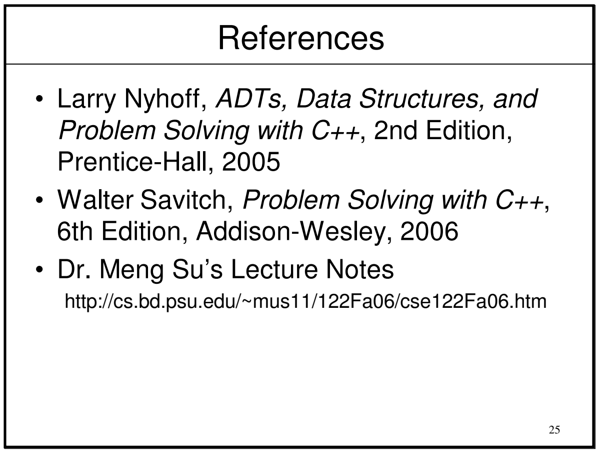
Implement (code) and test the units

Integrate the units and test the system

Deploy, operate, and maintain the system

Summary

- Course Introduction
- Software Development (Chap. 1)
  - Waterfall model and 5 phases of software development.
- Next Lecture
  - Introduction to abstract data types (Chap. 2)



References

- Larry Nyhoff, *ADTs, Data Structures, and Problem Solving with C++*, 2nd Edition, Prentice-Hall, 2005
- Walter Savitch, *Problem Solving with C++*, 6th Edition, Addison-Wesley, 2006
- Dr. Meng Su's Lecture Notes
  http://cs.bd.psu.edu/~mus11/122Fa06/cse122Fa06.htm