COMP2012:  
Object-Oriented Programming and Data Structures

Lei Chen  
Department of Computer Science and Engineering
COMP 2012

- Lecture time: Mon 1:30 PM-2:50pm, Fri 9:00am-10:20am
- Place: Room 3008
- Web site: http://course.cse.ust.hk/comp2012/csd_only/leichen/
  - Lecture and lab materials
  - Assignments and solutions
  - Responsible TAs for questions
  - Peek at the materials advanced of classes
Instructor

- Lei Chen
- Office: Rm 3546
- Research area: Databases and Data Mining
- Homepage: [http://www.cse.ust.hk/~leichen/](http://www.cse.ust.hk/~leichen/)
Grading

- 3 individual programming assignments (25%)
  - PA1: 5%
  - PA2: 10%
  - PA3: 10%
- 1 written assignments (10%)
- Lab exercises (10%)
  - 1% each, the best 10 labs
- Midterm (20%)
- Final (35%)
## Tutorials and Labs

<table>
<thead>
<tr>
<th>Section</th>
<th>Day</th>
<th>Time</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td>LA1-LAB</td>
<td>Monday</td>
<td>2:00pm - 2:50pm</td>
<td>4214</td>
</tr>
<tr>
<td>LA2-LAB</td>
<td>Friday</td>
<td>9:00am - 10:50am</td>
<td>4214</td>
</tr>
<tr>
<td>LA3-LAB</td>
<td>Tuesday</td>
<td>10:30am - 12:20pm</td>
<td>4214</td>
</tr>
<tr>
<td>LA4-LAB</td>
<td>Monday</td>
<td>6:00pm - 7:50pm</td>
<td>4214</td>
</tr>
<tr>
<td>LA5-LAB</td>
<td>Tuesday</td>
<td>5:00pm - 6:50pm</td>
<td>4214</td>
</tr>
<tr>
<td>LA6-LAB</td>
<td>Tuesday</td>
<td>1:00pm - 2:50pm</td>
<td>4214</td>
</tr>
<tr>
<td>LA7-LAB</td>
<td>Friday</td>
<td>5:00pm - 6:50pm</td>
<td>4214</td>
</tr>
<tr>
<td>LA8-LAB</td>
<td>Monday</td>
<td>11:30am - 1:20pm</td>
<td>4214</td>
</tr>
<tr>
<td>T1-TUT</td>
<td>Monday</td>
<td>12:00pm - 12:50pm</td>
<td>3598</td>
</tr>
<tr>
<td>T2-TUT</td>
<td>Monday</td>
<td>1:00pm - 1:50pm</td>
<td>3598</td>
</tr>
<tr>
<td>T3-TUT</td>
<td>Friday</td>
<td>2:30pm - 3:20pm</td>
<td>4503</td>
</tr>
<tr>
<td>T4-TUT</td>
<td>Tuesday</td>
<td>6:00pm - 6:50pm</td>
<td>2463</td>
</tr>
<tr>
<td>T5-TUT</td>
<td>Friday</td>
<td>9:30am - 10:20am</td>
<td>3598</td>
</tr>
<tr>
<td>T6-TUT</td>
<td>Thursday</td>
<td>1:30pm - 2:20pm</td>
<td>1511</td>
</tr>
<tr>
<td>T7-TUT</td>
<td>Friday</td>
<td>10:30am - 11:20am</td>
<td>1511</td>
</tr>
<tr>
<td>T8-TUT</td>
<td>Wednesday</td>
<td>9:30am - 10:20am</td>
<td>2463</td>
</tr>
</tbody>
</table>
## TAs

<table>
<thead>
<tr>
<th>TA</th>
<th>Name</th>
<th>Email</th>
<th>Room</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>TA Eddie Chan</td>
<td><a href="mailto:csclchan@cse.ust.hk">csclchan@cse.ust.hk</a></td>
<td>4214A</td>
<td>Monday, 10:30am - 11:30am</td>
<td></td>
</tr>
<tr>
<td>TA Peter Chung</td>
<td><a href="mailto:cspeter@cse.ust.hk">cspeter@cse.ust.hk</a></td>
<td>4214A</td>
<td>Tuesday, 3:00pm - 4:00pm</td>
<td></td>
</tr>
<tr>
<td>TA Mak Wai Ho</td>
<td><a href="mailto:wallacem@cse.ust.hk">wallacem@cse.ust.hk</a></td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>TA Cao Chen</td>
<td><a href="mailto:caochen@cse.ust.hk">caochen@cse.ust.hk</a></td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>TA Li Mengyu</td>
<td><a href="mailto:mliaa@cse.ust.hk">mliaa@cse.ust.hk</a></td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>TA Wang Jinglu</td>
<td><a href="mailto:jwangae@cse.ust.hk">jwangae@cse.ust.hk</a></td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>TA Zhang Chen</td>
<td><a href="mailto:czhangad@cse.ust.hk">czhangad@cse.ust.hk</a></td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>TA Zhang Honghui</td>
<td><a href="mailto:honghui@cse.ust.hk">honghui@cse.ust.hk</a></td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
<tr>
<td>TA Zhang Bo</td>
<td>zhangbocse.ust.hk</td>
<td>TBA</td>
<td>TBA</td>
<td></td>
</tr>
</tbody>
</table>
COMP2012: A Very Rigorous Course

- Covering C++ object-oriented programming and how to use it to implement some important data structures
  - These data structures make algorithms easy to be implemented and particularly efficient (COMP3711)
  - List, queue, stack, hash, (binary) trees, etc.
- Fast-paced
  - Keep up with the course schedule
- Programming-intensive and thinking-intensive sequence
  - You need to think smart and program smart in order to keep up
How Hard Should I Work?

- **Input time**
  - Normally, a 3-unit course takes about 8-9 hours/week outside classes. Since it is a 4-unit course, expect about 11 hours of work per week outside classes.
  - That means you need to make sacrifice on some activities.

- **Expect to absorb fast over the whole semester**
  - Must have some programming background and experience before.
  - Basic C++ knowledge at the level of COMP104.

- **This is a rigorous course, and you are mature students**
  - Don’t expect to be spoon-fed!
  - The lectures only provide you basic fundamentals.
  - The assignments will challenge you to apply the principles to reach the next level.
  - You are supposed to look up materials beyond the lecture notes.
  - You should cultivate an independent learning habit.
Tips on Effective Learning

- **Effective programming**
  - Planning (30-40% of time): algorithm design, refinements, modularity, roadmap
  - Coding (60-70% of time): syntax, logic, debugging, focusing more on implementation details
  - Strive to minimize your debugging time by testing your program one step at a time!

- **Effective learning**
  - Prestudy (15-30 minutes): initial ideas on what each lecture would be covering
  - Pay attention in class: ask questions if you don’t understand; try to follow as closely as possible
  - Post-study (3 hours): notes organization, book reading
Course Prerequisite

- Have programming/software experience before at the level of COMP104
  - Basic C++ syntax, decision loops, functions, I/O, pointers, etc.
  - Speedy review on that in the first week. If the materials are too difficult, you are not ready for the course yet.
- Basic mathematical skills equivalent to F.6 standard
  - Doing sums, GP, etc.
Major Course Topics

1. Basic C++ review (syntax, functions, pointers, I/O, etc.) and development of large software projects [1 week]
2. OOP concept and classes [2 weeks]
   - Object creation, destruction, member variables and functions
3. Basic data structure as ADT (Abstract Data Type) [2 weeks]
   - List, stack, queue and their applications
4. Generic programming [2 weeks]
   - Function and operator overloading; function and class templates
5. Standard template library (STL) [1 week]
   - List, queue, stack, string, vector, etc.
6. Inheritance, polymorphism and virtual function for code reuse [2 weeks]
7. Binary tree, binary search tree, data structures and operations [2 weeks]
8. Hash search [1 week]
Course Format

- Lectures
  - Slides and transparencies
  - Illustrative examples on board to supplement the slides and transparencies
  - Feel free to interrupt to ask questions
  - Not compulsory, but it is your responsibility to catch up with your missed lectures with your friends

- Tutorial
  - Supplement the lectures with more materials and examples
  - Discussion of programming and written assignments

- Lab
  - Programming exercise to show your basic understanding of lecture materials
  - Exercises will be graded
  - No lab hopping – you must attend the lab you sign up; otherwise you will get no grade for the lab

- Individual programming and written assignments
  - More rigorous problems to consolidate your knowledge
Labs

- Each lab is 3 marks (1 mark for attendance; 2 marks for quality)
- Attendance is strongly encouraged (you can show up at 10am)
- You can submit your lab assignment by Saturday night via CASS if you miss the lab
- All the lab materials will be available on Monday
Activate your CSD Computer Accounts


- By default, every UG doing CS courses will get a CSD PC account. For those UGs doing CS courses which require UNIX will also be assigned a UNIX account. Both accounts will have the same username as your ITSC account.

- New CSD accounts (PC and UNIX) are not activated by default. To activate your CSD account, go to an ITSC Computer Barn, login (with your ITSC login name and password), start a web browser (Microsoft Internet Explorer or Netscape), and access the following URL:

  https://password.cse.ust.hk:8443/pass.html
Midterm and Final

- There are a mid-term and a final
- Individually done
  - Unassisted by living objects
  - You have to sign “I did not cheat in this exam” before we grade your booklets
- Closed-book, closed-notes
- No calculator
- No early or late examination
  - Unless under very unusual circumstances with solid proofs
  - Reasons like sleeping over, society activities or traffic congestion are unacceptable excuses
  - You need to inform me beforehand for re-scheduling
Regrade Requests for Assignments

- Only be entertained within 1 week after the graded assignments are returned
- Please approach your TAs directly
Programming Assignments

- C++
- Individually done
- Run on Windows Eclipse
  - Your first few labs will cover the environment and how to write C++ programs
  - We will not entertain porting problem if you write in other platforms
- Submit the assignments by the deadline yourself
  - Course assignment submission system (CASS)
  - For more details on how to use it, please visit [http://cssystem.cse.ust.hk](http://cssystem.cse.ust.hk) → Undergraduates → CASS User Guide
- Writing programs is like learning a language or an instrument: you need to constantly practice it in order to write good codes minimizing bugs
I encourage you to discuss your assignments with your classmates or friends
- This is an effective way of learning

After learning, put everything in your own words
- Do NOT take short-cut by copying
- Copying is stealing (intellectual property), which is a crime
- You are too smart to copy

What if you are caught copying?
- Both the copier and the originator get 0
- 2\textsuperscript{nd} time: Both get 0 and one full downgrade
- Caught 3rd time: FAIL course grade
- If it is examination, an automatic FAIL
Cheating:
JPlag is very powerful to detect plagiarism

<table>
<thead>
<tr>
<th>Matches sorted by average similarity (What is this?):</th>
</tr>
</thead>
<tbody>
<tr>
<td>ckwongab(L2) -&gt; kytchan(L1) (92.2%)</td>
</tr>
<tr>
<td>kytchan(L1) -&gt; chwongac(L2) (89.8%)</td>
</tr>
<tr>
<td>ynyg(L2) -&gt; slchanaa(L2) (84.1%)</td>
</tr>
<tr>
<td>wcwongaa(L1) -&gt; kymak(L1) (83.9%)</td>
</tr>
<tr>
<td>cyhoab(L2) -&gt; chwan(L2) (82.0%)</td>
</tr>
<tr>
<td>wtwip(L1) -&gt; chsao(L1) (81.2%)</td>
</tr>
<tr>
<td>ph_lsx(L3) -&gt; ie_gyax(L2) (80.7%)</td>
</tr>
<tr>
<td>cmkwok(L3) -&gt; kylau(L2) (79.6%)</td>
</tr>
<tr>
<td>cs_hyxab(L2) -&gt; nlo(L3) (78.2%)</td>
</tr>
<tr>
<td>ma_wsmab(L1) -&gt; ma_sktaa(L1) (77.9%)</td>
</tr>
<tr>
<td>chemor(L2) -&gt; hmlleungab(L3) (77.6%)</td>
</tr>
</tbody>
</table>

Course introduction
// Binary Matrix multiplication operator
template<typename T>
Matrix<T> Matrix<T>::operator *(const Matrix<T> &m)
{
    //To implement
    Matrix<T> result;
    if (_colSize != m._rowSize)
    {
        return result;
    }
    else
    {
        result.element = new T* [_rowSize];
        if (result.element == NULL)
        {
            return result;
        }
        else
        {
            for (int i = 0; i < _rowSize; i++)
            {
                result.element[i] = new T [m._colSize];
                if (result.element[i] == NULL)
                {

                }
            }
        }
    }
    return result;
}

PPM.h(22-35) PPM.h(22-35) 15
Pixel.cpp(3-139) Pixel.cpp(3-139) 102
Matrix.h(8-479) Matrix.h(8-479) 570
Pixel.h(7-49) Pixel.h(7-49) 51
The following tricks do not work:
- Cut & Paste
- Add space
- Rename the variables
- Adjust order of code segment

The decision is made collectively by TAs and me. We are usually 100% sure.
Email Policy

- General course/lecture questions: to me
- Assignments: to the responsible TAs
- Labs and tutorials: to the responsible TAs
- Remember: our mailboxes are of limited size
- Use email unless it is necessary
  - Not effective to explain things
  - Visit my or TA’s office hours
- Please do not expect answers right away
- Please do not send us codes for debugging
  - We will not debug codes for you
Newsgroup

- There will be a newsgroup for the class
- Course-related and assignment-related information will be posted there
- Check it regularly
- Behave yourself in the newsgroup
Textbooks

- **Main books**
  - Larry Nyhoff, ADTs, Data Structures, and Problem Solving with C++, Prentice Hall

- **References**
  - Michael Main and Walter Savitch, Data Structures & Other Objects Using C++, Addison Wesley
    - We assume a C++ background at the level of Chapters 1-2, 4-8, 15, 17
    - We assume a C++ background at the level of Chapters 1-5, 7-8
  - Stroustrup, “The C++ Programming Language,” Addison Wesley
  - Ivor Horton, “Beginning C++: The Complete Language,” WROX
  - Savitch, “Problem Solving with C++: The Object of Programming,” Addison Wesley