CS 102 – Lecture 1

Syllabus, history, intro
Contact Info

- Instructor: Hannah Aizenman
- Email: haizenm00@ccny.cuny.edu
- Office: NAC 7/311
- Office Hours:
  - After class
  - by appointment
  - 2:00-4:00 T/Th
- Recitation Instructor: Andrew Nagel
Grading

- %10 – Attendance Quizzes
- %40 – Quizzes
- %20 – Midterm
- %30 - Final
## Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Topics</th>
<th>Chapter Readings</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fundamentals of C++ Programming</td>
<td>Chapter 1</td>
</tr>
<tr>
<td></td>
<td>Problem Solving Using C++</td>
<td>Chapter 2</td>
</tr>
<tr>
<td>2</td>
<td>Assignment, Formatting, and Interactive Input</td>
<td>Chapter 3</td>
</tr>
<tr>
<td>3</td>
<td>Selection Statement</td>
<td>Chapter 4</td>
</tr>
<tr>
<td>4</td>
<td>Repetition Statements</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>5</td>
<td>Modularity Using Functions</td>
<td>Chapter 6</td>
</tr>
<tr>
<td>6-7</td>
<td>Arrays</td>
<td>Chapter 7</td>
</tr>
<tr>
<td>8</td>
<td>Mid-term Exam</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>I/O Streams and Data Files</td>
<td>Chapter 8</td>
</tr>
<tr>
<td>10</td>
<td>Completing the Basics</td>
<td>Chapter 9</td>
</tr>
<tr>
<td>11</td>
<td>Introduction to Classes</td>
<td>Chapter 10</td>
</tr>
<tr>
<td>12</td>
<td>Class Functions and Conversions</td>
<td>Chapter 11</td>
</tr>
<tr>
<td>13</td>
<td>Pointers</td>
<td>Chapter 12</td>
</tr>
<tr>
<td>14</td>
<td>Final Exam</td>
<td></td>
</tr>
</tbody>
</table>
Plagiarism

- Plagiarism is copying solutions from pretty much anywhere, including friends, solution manuals, and random websites.

- Plagiarism can result in various academic repercussions as described in the CCNY integrity policy: http://www1.ccny.cuny.edu/current/integrity.cfm

- Any work suspected of being plagiarized will be dealt with per CCNY guidelines
Textbook

Software: Compiler

- Official/Supported Compiler: g++
- Windows: MinGW or Cigwin
- Mac: Xcode and Macports
- Linux: g++ is built in (but you might need to install the dev libraries)
Software: Terminal

- Windows: cmd or terminal
  - start->accessories-> command prompt
- Mac: terminal
  - applications->utilities->terminal
- Linux: Konsole or gterm
- Note: use Bash as the shell
Software: Editor

- Use something simple that supports:
  - Code highlighting
  - Code folding
- Any platform: vim, emacs, jed, nano
- Windows: notepad++, scite
- Mac: aquamacs, smultron
- Linux: kate, gedit, scite
History

1805

Jacquard

1837

Jacquard

Babbage and Lovelace

1890

Hollerith

Turing

WWII
Software Architecture Layers
Computer Data

- Digital Signal: low high
- Bit: 0 or 1
- Byte: Group of 8 bits
- Character: Byte (or group of Bytes) representing a letter or symbol
  - ASCII, unicode
Languages

- Machine language
  - 0s and 1s
  - transistor level
- Assembly language
  - Basic instructions (Boolean Logic, arithmetic)
  - CPU specific
  - Non portable
- Compiled Language
  - All types of instructions
  - Translated into machine specific assembly/machine code
Language Families

- Imperative:
  - a series of instructions
  - C/C++, Java, Fortran, Python

- Functional:
  - Lambda calculus
  - Scheme, lisp, haskell, JavaScript (to some extent)

- Logical:
  - Predicate logic
  - Prolog, SQL

- Markup (not real languages)
  - Describe document layout
  - HTML, XML, ReST, LaTeX
Why C++?

- Object Oriented – good for abstraction
- Statically compiled-great for speed
- provides low level access
- widely used in industry and open source
- C and C++ used almost interchangeably since C++ is essentially C with objects
- ++ means increment by 1
Translation Hierarchy

C program → Compiler → Assembly language program → Assembler

Assembler

Object: Machine language module → Linker

Object: Library routine (machine language) → Linker

Linker

Executable: Machine language program → Loader

Loader → Memory
Source code: hello.cc

```cpp
#include<iostream>
using namespace std;

int main()
{
    cout<<"Hello World!"<<endl;
    return 0;
}
```
Compile code

g++ -Wall hello.cc -o hello
output

Hello World!
Make an executable

Debug

- Use the -Wall switch to get debug statements
- Read the errors and look them up on google
- Write your code piecemeal
  - Test that each part works before moving on
  - Use print statements to check results
- Create solid test cases before you write the code
Quiz 1

What are the stages in translating code from hello.cc to hello.exe?

A) Compiler, Assembler, Linker
B) Assembler, Compiler, Linker
C) Compiler, Linker, Assembler
D) Linker, Compiler, Assembler
Sources

- Some material and slides came from Michael Grossberg and Wai Khoo
- Pictures: