CS 310 Introduction

Class 1

Administrative

- class web site `http://borax.truman.edu/310`
- syllabus
- Blackboard for grades, occasionally content
- C++ resource online `http://cplusplus.com`
  - make sure you look at C++11 stuff
  - online references are for low-level things, not for solutions

Academic Honesty

- you may
  - discuss the mechanics of editing, compiling, and running a program
  - discuss the mechanics of a built-in C++ function, e.g., stoul
  - discuss the mechanics of document preparation using LaTeX
  - discuss the mechanics of using the shell, printing, etc.
  - use the instructor's code, original or modified, without attribution
- you may not
  - look at any portion of another student's code or writeup
  - show any portion of your code or writeup to another person other than the instructor
  - discuss the content of any assignment, in person or electronically, with anyone other than the instructor
  - copy code or solutions from the internet or a published source
  - collaborate in any way on tests
- if you're unsure, ask!

Software Environment

- you must use an environment that provides *all* of the following
  1. the Bash shell which provides:
     - redirection of standard input and output
     - command-line control of your own programs
     - command-line control of the compiler (see below)
  2. clang and llvm version 7.0 or later
     gcc is not acceptable
     MSVC++ is not acceptable
  3. a code editor such as Emacs, vim, or VScode
  4. a plotting system that can produce 2D plots of x,y points from a file and of defined functions and export the resulting images
     e.g., gnuplot, R's ggplot2
- the department's Linux server sand provides all of the above

Platform

- I most strongly recommend that you use Linux for this course, and Linux on your own computer is easier than using sand
- mac OS is almost as good, although you'll have to install some software
- it is possible to use Windows, but it will be up to you to install the correct software and I can't offer any technical help if things don't work correctly; I do not recommend Windows for this course
- whatever system you use, make backups of your files — sooner or later, all disk drives fail (sand makes automatic daily backups every night)

Console Scripting Shell

- the Bourne-Again Shell (bash) on Linux or mac os is the command line interpreter for the course
- note that Terminal.app in mac os has switched to zsh by default, but bash is still there; google "mac os change terminal to bash"
- bash was written by programmers for programmers
- an incredibly powerful productivity tool, a selling point trying to get an internship or job
- use a cheat sheet and learn the basic bash commands
- if you insist on using Windows, the best setup is to run full Linux in a virtual machine
- second best is to use the Linux Subsystem for Windows, which includes bash

- the primary types of bash commands needed in this course are
  - input and output redirection
  - create and manage directories
  - create, view, edit, rename, delete files

Assignments

- homework assignments will be
  - PDF generated from LaTeX source
  - C++ source
- tests will be completed in LaTeX

Code Editor

- Linux
  - Emacs (this is what I use)
  - vim
  - geany
  - VScode
- Mac
  - TextMate
  - VScode
  - Emacs
  - vim
- Windows
  - Emacs
  - vim
  - VScode
  - notepad++

Plotting System

- best plotting systems:
  - gnuplot (this is what I use)
  - ggplot2 in RStudio

- need to be able to plot standard functions
  e.g., $f(x) = x^2 - 3x$
  as well as (x, y) data contained in a file, e.g.,

```
1  3
2  6
3  9
4  12
```

- need to be able to control and format:
  - axis labels
  - axis ranges
  - legend or key
  - size or aspect ratio
  - export to graphics: jpeg, png, or pdf

Keys to Success

1. attend class
   - by 10:30, you need to be in your seat, notebook out, ready to take notes
   - slides of each day's class on the course calendar will be available after we are finished with that unit
2. participate
   - if you have a question, don't sit there quietly — interrupt me and ask your question
   - don't sit there confused — interrupt me and ask your question
   - interaction is part of your participation grade

Keys to Success

3. program!
   - computer science is so much more than programming
   - but you cannot be a computer scientist without programming
   - this course requires strong programming skills
   - you must have completed both CS181 and CS260 with C or better

   - program every day
   - type in, run, and experiment with
     - my code examples
     - code you find online

A C++ Example

- use an editor to create program source code hello_world.cpp
- use the command line to compile the source code to executable
- the standard compiler command we will use:

```
$ clang++ -pedantic-errors -Wno-c++98-compat
    -Weverything -std=c++11 -o hello_world
    hello_world.cpp
```

  `-pedantic-errors` enforce ANSI standards
  `-Wno-c++98-compat` no backwards compatibility
  `-Weverything` turn on all warnings
  `-std=c++11` use C++ version 11
  `-o hello_world` specify executable file name
  `hello_world.cpp` the source code file containing main

- use the command line to run the program:

```
$ ./hello_world
```

Markup Language

- HTML stands for hypertext markup language
- a markup language is a system for embedding tags or codes into a plain text document
- these tags tell a renderer how to display the document's contents
- for example, the HTML on the left might be rendered by a browser to appear as seen on the right

```
<h1>A Heading</h1>

<p>This is a paragraph.</p>
```

Markup Language

```
<h1>A Heading</h1>

<p>This is a paragraph.</p>
```

- the tags tell the renderer what kind of text it is
- the tags do not tell the renderer how to display the text
- the renderer (with hints from CSS) makes its own decisions on how to display

- the author provides the content
- the renderer controls the appearance

LaTeX

- HTML and CSS were invented for displaying documents in browsers
- they are the right tool for the job
- we need to use a tool for creating technical PDF documents in the computer science domain
- the right tool for that job is LaTeX

- LaTeX is pronounced with a long or short A in the first syllable, and the final syllable rhymes with “deck”
- the final letter is not the Roman character X, but rather the Greek letter Chi, the equivalent of Roman K

The WYSIWYG Problem

- WYSIWYG editors such as Word are menu systems
- the only commands you can give are the ones in the menus
- if there isn't a button for it, you can't do it

- a complete markup language, such as SGML or LaTeX, allows the user complete control down to the individual pixels (if desired)

Logical Design

- however, while you can micromanage your document down to the pixel
- you should not even think of doing that
- rather, you use LaTeX to specify the logical design of your document
- then let LaTeX make it look good

- by contrast Word makes it hard to specify the logical design
- and easy to mess around with local visual elements
- this results in a jumbled, disorganized document
- the reader thinks that the author's brain are similarly jumbled and disorganized
- LaTeX makes it easy to specify an organized, logical layout, freeing the author to concentrate on the content, not on the appearance

A LaTeX Example

- use an editor to create document source code hello_world.tex
- use the command line to typeset the source code to PDF:
  ```
  $ pdflatex hello_world
  $ pdflatex hello_world
  ```
- typically need to run the pdflatex command twice so that cross references and numbering get updated
- view the results:
  ```
  $ atril hello_world.pdf &
  ```

LaTeX Resources

- `https://www.latex-tutorial.com/tutorials/`
- read the sections in the left column below
- read the sections in the right column if you wish
- this class won't use the sections not listed

Required Sections	Optional Sections
01 Your first document	00 Installation
03 Packages	02 Document structure
04 Math	09 Tables
05 Adding pictures	13 Source code highlighting
12 Drawing graphs (tikz)	

Homework Assignment 113

- on course calendar
  `http://borax.truman.edu/310/CalendarCS310.html`
- due Thursday at noon
- full instructions are on the link on Thursday

Homework Assignment 113

- part 1: a simple program that illustrates some C++ features
  - Doxygen documentation for function headers
  - command line arguments
  - unsigned ints for values that can't be negative
  - typecasting using `static_cast`
  - editor setup
    - margins — line length maximum
    - indentation — 2 spaces, no tabs
    - backup files — for your protection
  - many things in the C++ style guide
- part 2: a document that illustrates some features of our homework style
  - document format for homework
  - typical packages
  - title, your name
  - includes the question as well as the answer
  - inline vs. display math