**CS 2060: Programming with C - Introduction**

**Introduction**

* Let's talk about me!
  + Bachelors and masters in Computer Science - both from UCCS
  + After bachelors - worked at Texas Instruments for 4 years
  + Returned to UCCS fulltime to work on masters
  + While working on masters I taught CS classes - that was in the 90's!
  + After masters - Digital, couple of startups, Compaq, HP, HPE
  + Left HPE to pursue teaching full time
  + 30 years of industry experience
    - Software engineer 20+ years
    - User experience engineer 10+ years (last job at HPE)
  + Standalone applications, enterprise applications, web applications, mobile, UX design
  + My team at HPE was a development team with 8 other developers
  + We were part of a large project - probably 50 developers in US - another 20+ in India
  + Our team did both GUI development and server (backend) development
  + I did the user experience design so I worked generally with the GUI developers and other UX engineers
  + We did what is called Agile development – followed scrum practices
  + GUI developers wrote in Java Script, HTML, CSS
  + Backend developers use C++ and Java
  + Married and have one son who is attending UCCS - Mechanical Engineering
  + Used to race motocross - big part of my life for many many years- now I mountain bike
* Let's talk about you! My view of the class:
  + Expect people with a wide range of knowledge and skills
  + We will start with the basics - but we will move pretty fast
  + I’m assuming you’ve had one of the classes listed on the prerequisites but if you’re struggling with anything as we get started come see me so we can make sure you don't get into the weeds right away
  + This class is not just about learning C it is also about problem solving
  + When working as a developer problem solving is a huge part of your job
  + The goal is to teach you how to start becoming good at solving problems
  + Please do not hesitate to ask questions
* Syllabus
* Computer systems
  + You may use PC’s in any of the laboratories or your own systems
  + If you want to use a personal computer, you need to setup your environment
  + I will provide instructions for how to setup a C development environment
  + Notes with these instructions will be placed on Blackboard

**Chapter 1: Introduction to Computers, the Internet, and the Web - Lecture 1**

**Announcements**

Reading:

* Chapter 1
  + Read sections 1 – 6 and 12-14
  + DO NOT worry about reading the following sections
    - 7 (C++ and Other C-Based Languages)
    - 8 (Object Technology)
    - 9 (Typical C Program Development Environment) – based on a Linux-based C system – most people I believe will be using Windows or Mac and we will be using Eclipse which is an IDE
    - 10 (Test-Driving a C Application in Windows, Linus, and MAC OS X)
    - 11 (Operating Systems)
* Chapter 2

Assignments:

* 1st programming assignment will be handed out in next class

**Terminology**

**Programming is full of terminology!**

* Terminology will be presented before each class.
* I will NOT be testing you on terminology **definitions** on the quizzes or exams but you need to **know the meaning** of the terminology we cover.
* A good study technique for learning terminology is using notecards – start creating a stack of terminology cards now!

**Terminology**

* Software terms
  + Algorithm
    - The set of steps to solve a problem
    - For example, the steps to sort a list of numbers from ascending to descending
  + Pseudocode
    - English mixed with programming code
    - You represent an algorithm in pseudocode
    - Pseudocode is not a programming language
    - Could be implemented in any programming language
  + Program (software)
    - Instructions for a particular task written in a language like C
  + Source code
    - Another name for a program - collection of computer instructions
    - When writing C programs, source code will be saved in .c files
  + Complier errors
    - Errors that occur when you violate the syntax rules of a language
  + Logic errors
    - Error that causes your program to produce the wrong result.
    - Maybe you multiplied when you were supposed to add, etc.
  + Runtime errors
    - Errors that occur when running your code - things that cause program to crash
    - Memory leaks, division by zero, etc
    - Also, called bugs
* Hardware Terms
  + CPU
    - Central processing unit - brain of your computer
    - Executes the statements in your code
    - Most computers have multiple processors
      * Dual-core (two CPU’s), quad-core (4 CPU’s)
  + Random Access Memory (RAM)
    - Program is placed into RAM when it is executed
    - Most computers come with 2 to 16GB (gigabytes) of RAM
  + Bit
    - A binary digit - one or zero
  + Bytes
    - A byte is 8 bits
  + Kilobyte (KB)
    - 1024 bytes is 1 KB
  + Megabytes (MG)
    - 1 million bytes or 1024 KB is 1 MB
  + Gigabytes (GB)
    - 1 billion bytes or 1024 MB is 1 GB
  + Terabyte (TB)
    - 1 trillion bytes or 1024 GB is 1 TB
  + Petabyte (PB)
    - 1024 Terabytes is 1 PB
  + Hard drive
    - Persistent storage device for applications, documents, etc.
    - Most computers come with hard drives that are 500 GB or 1 TB in size
* Programs and Programming Languages
  + Machine Language
    - The language computers understand - one's and zero's
    - All other languages must be converted into machine language before a computer can understand it.
    - Of course, cumbersome for humans to read/understand.
  + Low Level Languages
    - A language that is above machine language but still hard to write code in
    - Assembly language is one such language
    - Use program called an assembler to translate assembly code into machine language
  + High Level Languages
    - A language that is easy for humans to read like C, Java, C++, etc.
    - Use program called a compiler to translate high-level language into machine language
* Programming Environment
  + Compiler
    - A program that translates high-level language into machine code
  + Operating System
    - Software program that runs on your computer - like Windows
    - The OS manages the computer's basic functions - running programs, controlling devices, etc
  + IDE
    - Integrated Development Environment
    - Contains all the tools to write software - editor, complier, etc
    - Eclipse is an IDE - different versions have different names

**C Programming Language**

**The C Language**

* C is an old language!
* Implemented in 1972 – 45 years ago!
* Great language when performance is important such as with
  + Operating systems
    - Linux Kernel
    - Windows: C, C++, C# (C-Sharp)
    - Java Virtual Machine (JVM): C
    - IOS and OSX: C and Objective-C
  + Embedded systems
    - Microprocessors embedded in devices that serve a dedicated function
    - Cell phones, printers, home appliances, etc.
  + Realtime systems
    - Air traffic control systems
  + Communication systems

**C11 Standard**

* The latest C standard
* Approved in 2011
* What the book uses

**C Standard Library**

**Library**

* Collection of C built-in functions, constants and header files that makes your life easier
* It is important to learn what is available in the library
* Reuse existing code instead of writing it yourself
* You know the code in the library works so reduces debugging
* Increases code portability

Website

* The **C Standard Library** or libc is the **standard library** for the **C** programming language, as specified in the **ANSI C standard**.
* Documentation for the ANSI standard C library can be found at
  + <http://www.csse.uwa.edu.au/programming/ansic-library.html>

**Programming Assignment Requirements 10 min**

**Expectations**

* The course is for students who are proficient in some other high-level language such as Java.
* Programming assignments must follow the polices stated in *Programming Assignments Policy*

**Policies**

* Please see the *Programming Assignments Policy* document
* Late programming assignments will not be accepted
* Even if you cannot get the assignment to work you can still get 50% of the grade but ONLY if the assignment meets the other 3 criteria (documentation, quality, specification). The point is - hand in what you have – show me that you tried to figure it out!