Stuyvesant High School Computer Science
MKS65C - Systems Level Programming

teacher:
email:
office: Room 301
available periods:

course website: stuyvesant-cs.github.io/courses/mks65/

Course Description:
Systems Level Programming covers the C programming language and development in a linux environment. Over the course of the semester students will learn how to control the main components of a computer (memory, storage, input/output, processor) at a low level. Over the course of the semester students will write a terminal shell as well as network server and client programs.

AP Computer Science is a prerequisite for Systems Level Programming.

Required & Recommended Tools:
• Notebook/Section in binder.
  • All students are required to take physical (pen & paper) notes for this class (barring any required accommodations).
• Github account: https://github.com
• Google Group Q & A forum.
  • Invitation links will be sent out after the first week of classes, you must accept.
• Recommended
  • Access to a linux computer outside of class.
  • If you cannot do this at home, you may always ssh into school computers.
  • See course website (resources section) for help on this.
  • Don't forget about the CS Dojo (307, M-Th, 3:45 - 5)
• Your notes will be your primary resource, and the most up-to-date information on C and linux will be found through online resources so there is no textbook. If interested, The C Programming Language, by Brian Kerninghan and Dennis Ritchie is the definitive book on C. There are copies at Stuyvesant available to look at.

Course Requirements:
• Treat each other with respect.
• Come to class on time.
  • Absences and latenesses must be accompanied by a note.
• Participate in class discussions, including the online Q & A forum.
• Submit work on time.
  • **There are no exams** in this course, your grade will be primarily made up from your submitted work.

Grade Breakdown:
• Participation: 10% (this includes in-class discussions, Q&A participation and group work).
• Work assignments: 70%
  • There will be approximately 2 assignments a week, involving a mix of individual and pair work.
  • These assignments will be posted on the class website.
  • Work grades will be based on how well the required tasks were performed.
  • Late assignments will be accepted up to a week after the deadline with a penalty.

- Projects: 20%
  - Each marking period will conclude with a larger scale project.
  - Projects may be done individually or in small groups (depending on project).
  - There will be some class time devoted to projects, but they will require outside time as well.
- Work and project submission must be done using github.

Course Outline:
- **Unit I: The C programming Language & Memory Management**
  - Syntax and variable types
  - `printf()`
  - Arrays
  - Pointers
  - Strings
  - Separate compilation
  - Dynamic/manual memory allocation
  - `struct, union & typedef`
  - Preprocessor instructions
  - **Project**
- **Unit II: Storage & File Systems**
  - File permissions
  - Hierarchical file systems
  - Program file table & `open()`
  - `read() & write()`
  - `umask() & lseek()`
  - File metadata & stat()
  - Navigating directories
  - Program input
  - **Project**
- **Unit III: Process Management**
  - Signals & Signal handling
  - Processes
  - `fork() & wait()`
  - Executing external programs
  - Parsing strings
  - File redirection
- **Unit IV: Interprocess Communication**
  - Shared memory
  - Semaphores
  - Unnamed pipes
  - Named pipes
  - Server-Client design patterns
- **Unit V: Network Communication**
  - IP Addressing
  - Network Ports
  - Network Protocols
  - Using Sockets
  - OSI Network model
- **Final Project**