Stuyvesant High School Computer Science
MKS66C - Computer Graphics

teacher: Mr. JonAlf Dyrland-Weaver
email: dw@stuy.edu
office: Room 301
available periods: 3, 6, 7, 8

course website: stuyvesant-cs.github.io/courses/mks66/dw

Course Description:

Over the course of the semester, students in Computer Graphics will write their own graphics engine, starting with line encoded and ending with lighting, shading and animation. The course is built around devising, writing and using the algorithms that are key to computer graphics. The class is not taught in any specific language because it is focused on the algorithms that students can choose to implement in a language of their choosing. Resources will be provided in C and Python.

AP Computer Science is a prerequisite for Computer Graphics.

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 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)

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: 15% (this includes in-class discussions, Q&A participation and group work).
- Work assignments: 85%
  - The class focuses on building a graphics engine. Each assignment will add new features to the engine.
  - There will be approximately 12 assignments to make up the graphics engine.
  - Since a working version of the previous assignment will be needed to continue, working solutions in C and Python will be provided by the time the next assignment is posted.
  - Once working solutions have been posted, late assignments will not be accepted.
  - These assignments will be posted on the class website.
  - Work grades will be based on how well the required tasks were performed.

- Most assignments will require a picture submission demonstrating the new features in addition to working code. Picture submissions will be made through a gallery website.
- Code submission must be done using github.

Course Outline:

- **Unit I: Graphics Engine Foundations**
  - Digital representation of images.
  - Image file formats.
  - Creating PPM images.
  - Bresenham's Line Algorithm
- **Unit II: Representing Images as Points**
  - Matrix math for Computer Graphics.
  - Using edge matrices to store points.
  - Applying transformations to images.
    - Translations, dilations, and rotations
- **Unit III: Advanced Shapes**
  - Parametric equations.
  - Generating circles using parametric equations.
  - Generting Hermite and Bezier splines.
- **Unit IV: 3-Dimensional Shapes**
  - Generating points on the surfaces of rectangular prisms, spheres and tori.
  - Rendering surfaces instead of edges.
  - Using polygon (triangle) point matrices.
  - Generating polygon meshes of 3-D shapes.
- **Unit V: Coordinate Systems**
  - Global coordinate systems.
  - Relative coordinate systems.
  - Keeping track of relative coordinate systems in code.
- **Unit VI: Lighting & Shading**
  - Scanline conversion.
  - The Phong Reflection Model.
  - Flat Shading using the Phong Reflection Model
- **Unit VII: Computer Language Design & Animation**
  - Parts of a compiler.
  - Creating a compiler for our Motion Description Language (MDL).
  - Generating animation in MDL using linear interpolation.
- **Unit VIII: Advanced Topics**
  - Gouraud Shading.
  - Phong Shading.
  - Importing polygon meshes.
  - Creating advanced 3-D shapes.
  - Generating animation using non-linear interpolation.