Computer Programming — 2021-2022 (Spring) — Fred Kral — The Marin School

Teacher: Fred Kral, Ph.D.
Email me: fkral
Call school: 415-339-9336
Get info: http://teach.kralsite.com

Students of this computer science elective have an interest in working with algorithms, data, logic, algebra, geometry, discrete mathematics, processor-based devices, and bugs. While a large part of the course is spent programming, the focus is on problem-solving techniques commonly used in computer science and on the larger issues of computer science. Students gain more and more understanding of the big ideas of computer science: creative development, data and information, algorithms and programming (including abstraction), computer systems and networks (including the Internet), and the impact of computing. The impact of computing addresses topics such as privacy, security, equity, copyright, artificial intelligence, and commerce. Students learn about computer architecture, such as the internal components of a computer, how computers deal with data and instructions, and the underlying computer logic.

During the first half of the course, students build a foundation with the building blocks of procedural programming (basic control structures, strings, file input/output, and arrays) and they get a taste of object-oriented programming. Students learn interactively and get immediate feedback from graphics. During the second half of the course, students continue to learn procedural programming skills while increasingly learning to use object-oriented programming concepts and principles. Students add and modify code for hands-on simulations, games, and animations on computers, microprocessors, and mobile phones. The course is project-driven and students are required to complete substantial programming projects. Students will learn at least one common contemporary language in depth. Object-oriented languages used in the past include Java and Python. **Co-requisite:** Algebra 2/Trigonometry, more math recommended. **Offered to grades 10-12.**

Text and Materials

Python:

• Matthes, Eric. *Python Crash Course: A Hands-on, Project-based Introduction to Programming*. San Francisco: No Starch Press. First or Second Edition. Highly recommended.
• *Python Crash Course* Cheat Sheets, https://ehmatthes.github.io/pcc_2e/. Required.
• Swaroop, C.H. (a.k.a. Swaroop Chitlur). *A Byte of Python*. Online textbook, http://python.swaroopch.com/. Recommended.
• Grok Learning, https://groklearning.com/.
• Python.org, https://www.python.org.

Micro Python:

• Microbit.org http://microbit.org/

Blockly:

• Code.org, http://www.code.org/. Hour of Code materials written in Blockly. Begin with *Classic Maze* and/or *Minecraft Hour of Code*, online courses. Strongly recommended.

General:

• Stackoverflow website, http://stackoverflow.com. High quality website with help.

Course materials:

• Laptop computer: a dependable, relatively recent, and reasonably fast computer. 8GB RAM or more is preferred. Should run Windows, OS X, or Linux. Required.
• USB flash memory stick. Required.

Expected School-wide Learning Results (ESLRs):

Students at TMS have the opportunity to become: 1. Self-reliant learners. 2. Self-directed individuals. 3. Critical and creative thinkers. 4. Effective communicators. 5. Responsible members of society. Each of the components of this course prepare students to obtain results 1-5.

Assessment

Progress in work ("HW"): completing assignments. Assessed relative to each student's goal.	5 points per assignment
Quantity of work: taking on work that goes further and deeper. Assessed relative to the full curriculum.	50 points per semester
End-Semester Assessment: Final Project.	20% of course grade
Participation: positive energy and interest level during in-class work including listening to others, contributing to discussion, completing assignments, working on projects, helping others, volunteering to research questions, and practicing during class. Assessed informally.	25 points per week
Commitment to learning: Taking on what is challenging to you, getting help, communicating with the teacher, engaging with the material, and documenting work in an online notebook and in the computer code itself. Assessed informally.	100 points per semester
Projects: quality, design, complexity, difficulty, originality, correctness, executability, code comments, and documentation.	100 points per project, depending on size
Resourcefulness: developing self-reliance and independence by getting help in multiple ways including using textbooks, finding Internet help, looking at old code (from yourself or others), getting help from different students, forming a group of students, and asking the teacher for help.	200 points per semester

Class Policies

Class computer use policy

Personal and school computers shall be used for Computer Science only. Programming is tiring. It is tempting to engage in off-topic activities. Thus, from time to time, students may take breaks during which they shall not use any technology. The Marin School supports the responsible use of technology on our campus. Students who violate the school Technology Policy, as written in the Parent and Student Handbook, will be expelled from the classroom. This will lead to disciplinary consequences.

Late work policy and tardy policy

The teacher enters grades once per week. Students get credit for late or partial work up to that weekly deadline. Students who are late to class or leave the classroom for an extended time during class receive a maximum of 60% of the day's in-class work credit.

Collaboration policy

I encourage study groups. You may work with others (not just students) unless instructed otherwise as long as all of you contribute. Put the name of each contributor on an assignment to avoid issues with plagiarism. **If you take a major piece of code from the Internet or a book, give it credit in a way that the teacher can easily see it.** Do not make significant pieces of code appear as your own work because that is plagiarism. Plagiarism is described in the Parent-Student Handbook.

Coding Projects (Term 3 and/or Term 4)

During the course of Computer Programming, it will normally be clear if a student should switch to the Coding Projects course that is not UC A-G approved.

Using programming skills learned as needed to complete their projects, students use their knowledge for applications such as games, simulations, web development, graphics, and data processing. Students select projects and present the results to their peers. Both text-based and drag-and-drop languages have been used in the past.

Let's have a great course! Please connect and email! – Fred