Programming course descriptions

Intro to Programming: graphics & mini-games

GRADES 8 AND UP — ONE YEAR — MATH/TECH DEPT.

This course teaches how to build expressions and functions that solve problems. Projects involve writing code to produce graphics and games, and the class is structured to maximize the time students spend writing programs (sometimes independently, and often with a partner) while learning how to go strategically from a problem statement to tested, reliable code. **Prerequisite:** Algebra I.

Programming for the Internet: Content, presentation, and protocols

GRADES 9 AND UP — ONE SEMESTER — MATH/TECH DEPT.

This course teaches how the Web, domain names, and email work, and how to write HTML, CSS, and programs that create and manipulate Web pages. We will also study layout and typography on the Web. Projects include a networked game and the development of a Web site. **Prerequisite:** Intro to Programming.

Intro to Computer Science: Abstraction, algorithms, and data structures

GRADES 10 AND UP — ONE SEMESTER — MATH/TECH DEPT.

This course investigates two important fundamental questions in computer science: "How can we avoid writing the same code multiple times?" and "How can we compare one method of solving a problem to another?" We will look at alternative ways of organizing programs and data to solve problems such as searching, sorting, and finding paths. We will implement common data structures such as binary trees, BSTs, graphs, priority queues, and suffix trees. The course also introduces the concepts of (1) space/time analysis of programs, (2) higher-order functions (functions that consume or produce functions), and (3) generative recursion and several generative-recursion algorithms (such as quicksort, merge sort, fractal drawing, or Bézier curve drawing). **Prerequisites:** Algebra II and Intro to Programming.

Language building blocks: Models of computation

GRADES 10 AND UP — ONE SEMESTER — MATH/TECH DEPT.

This course investigates the question: "What is computation?" We will study two models of computation: imperative (as in assembly language, C, Basic, or Forth) and functional (as in BSL, Haskell, or the λ-calculus). This includes studying program data (including the concepts of a language's call stack and data heap), environments, function application, variable lookup tables, and low-level representation of numbers, strings, and other forms of data we use, as well as object-oriented programming. Projects include writing an interpreter and a compiler. **Prerequisites:** Algebra II, Intro to Programming, and Intro to Computer Science.

Independent Study (semester) in programming

Independent study is available for students who have completed at least two semesters of programming classes. Past students have studied topics such as Java programming for AP Computer Science; web programming with Python; and virtual machines and interpreters.