COMPUTER

We humans are surrounded by the stuff that used to be science fiction—portable computers, artificial intelligence, 3D printing, electronic games, online journals, instant reference books, genetic sequencing, video and still cameras everywhere, nanotechnology, increasingly massive datasets—with more innovations coming backed by data processing, design, number crunching, and computer science. Our courses show students how to be more than just consumers or users: they will be independent creators on computers, able to control and help shape the tools of today and tomorrow. Using software that runs similarly on Mac, Windows, Unix/Linux, and tablet computers, our courses teach a range of topics including programming, graphics, circuitry, web, spreadsheet analysis, logic, and other skills that are useful for doing everything from analysis to artwork. Classes are full year and meet twice a week unless otherwise noted. Visit tinyurl.com/sacc2015 for more information about any of these classes.

3D MODELING AND PRINTING

(Arum)

3D printers are personal fabrication tools that are a part of an evolving modern world of technology that allows students to become producers, inventors and artists. Students will create, design, invent and prototype while efficiently and inexpensively taking their digital designs into the real world. Students will be able to easily understand the strengths and limitations of their work and will be encouraged to modify their designs, thereby participating in an iterative engineering design process. Students will learn various 3D modeling techniques and explore several 3D modeling software tools and packages.

ALGORITHMS FOR GENETIC SEQUENCING

(Roam)

For experienced programmers, this class introduces programs that analyze genetic sequences. There are numerous exercises in pattern-matching and string comparisons, calculating family trees based on DNA sequences while taking into account the basic operations of mutation, insertion, deletion, and transposition. Though we mostly use simplified models of DNA (without worrying about protein folding), this topic gives us a chance to study "design patterns," datastructures and algorithms for large data sets, and basic molecular models. **Prerequisite(s)**: two programming courses.

ANIMATION ON COMPUTERS

(The Department)

In this class, learn about computer-aided methods of animation. Use Flash to complete frame by frame animations, including the traditional walk-cycle project. Use AfterEffects for more advanced techniques such as pinning digital puppets and working within a 3D space. Create

special effects such as lightning and explosions. Additional projects may include stop motion, green screen projects, 3D movie titles (like the iconic *Star Wars* titles), music videos, and a final animated movie using techniques of the student's choosing. No prior experience is required, though attention to detail and perseverance are a must!

GAME PROGRAMMING
(The Department)
Designing games presents unique challenges distinct from the design issues of other interactive media. In addition to the user interface, one must consider story, culture, modeling, and implementation. This course will explore developing usable and engaging games, human computer interaction, thematic structures, graphic design, sound effects, and game aesthetics. The course will take into account the history of non-digital and digital games, role-playing, puzzles, interactive fiction, and 3D modeling. Students will plan and create games both individually and collaboratively using a variety of languages and tools such as Unity3D, Livecode, Javascript, Java, and Python. We'll explore the creative possibilities of game design and experience the beauty and logic of programming. **Prerequisite(s)**: some programming experience or permission of the instructor.

GENERATIVE ART
(Arum)
Codes are instructions, information, rules, and routines that govern our lives and also our technologies. As long as there have been codes, there have been artists to break them, play with them, and reinvent them. This course will explore how to use codes creatively to generate artworks and how artists make codes, use codes and break codes in the process of creating art. In addition to learning how to program with processing, we will also look at instructional procedures, conditional design, self-imposed constraints, generative art, and parametric design. No prior programming experience is necessary but a willingness to work hard is required.

iPHONE PROGRAMMING
(The Department)
Learn how to program with Objective-C, Interface Builder, and XCode on the iPhone and iTouch's unix-based operating system. Understand the way the iPhone applications work and how to build them. Actively and creatively explore this new field of little computers using the iPhone as the main research platform. No iPhone required. **Prerequisite**: some programming experience.

PHYSICAL COMPUTING 1
(Arum)
Learn how to interact physically with a computer without using the mouse, keyboard or monitor. Move beyond the idea that a computer is a box or a system of information retrieval and processing. Using a microcontroller, a single-chip computer that can fit in your hand, write and execute interactive computer programs that convert movement into digital information. Work with components such as resistors, capacitors, diodes and transistors as well as integrated circuits. Through lab exercises and longer creative assignments learn how to program, prototype and use components effectively. Control motors and interpret sensor data, as well as explore

advanced concepts in interface, motion and display. **Prerequisite**: some programming or permission of the instructor.

PHYSICAL COMPUTING 2

(Arum)

Students combine theory and practice to interface microcontrollers and transducers. We learn how to make devices respond to a wide range of human physical actions. Building on previous knowledge acquired in Physical Computing 1, we build projects from schematics, make programs based on class examples, and make interfaces talk to each other. Topics may include: networking protocols and network topologies; mobile objects and wireless networks of various sorts; digital logic building blocks and digital numbering systems. Students are involved in short production assignments and final projects, and create a digital portfolio to document their work and research. **Prerequisite(s)**: Physical Computing 1 or permission of the instructor.

PHYSICAL COMPUTING WORKSHOP

(Arum)

Creating interactive work relies on building a relationship between the object and the viewer. By gathering information in the form of input, processing that into meaningful data, and outputting that contextually, new forms of engagement and interaction with an audience can be established. This class is for students who have prior experience with Physical Computing and would like the opportunity to develop their own project and spend time researching, testing, prototyping and documenting it. **Prerequisite(s)**: Physical Computing 1 or permission of the instructor.

PROGRAMMING 1

(The Department)

Explore the science and art of computer programming. For students who want to create and modify their own computer software, this course uses the high-level programming languages Java (an internet-savvy version of C++) and Livecode (a multimedia descendent of Hypercard) to introduce the basics of computer control. We use loops, variables, procedures, input, output, and branching decisions (with Boolean logic) to control graphics, sounds, and information.

PROGRAMMING 1 (INTENSIVE)

(The Department) (4x per week)

Explore the science and art of computer programming. Learn important problem-solving and design strategies like modularization and iterative design which can apply to both programming and non-programming environments. This intensive, four periods per week class is for students who want to master fundamental programming concepts which include loops, variables, procedures, input, output, conditionals and data structures. Assignments will allow students to control graphics, sounds, and data while also encouraging them to think creatively, reason systematically, and work collaboratively.

PROGRAMMING 2

(The Department)

A continuation of Programming 1, for students who are becoming more confident in their ability to combine data types and complex computer routines. We use Java (an internet-savvy version of C++) to look more deeply at object-oriented programming: class definitions, inheritance, methods, fields, arrays, and collections. Large projects include writing an interactive, animated project with control windows and graphics. **Prerequisite(s)**: Programming 1 or permission of the department chair.

SOFT CIRCUITS: WEARABLE, SOFT AND EXPERIMENTAL CIRCUITS

(Arum)

Standard electronic components can be hard, brittle or difficult to work with. Embedding them in soft environments, like clothing or toys, can be awkward. Building circuits using paper, fabrics, thread and paint opens up new possibilities for soft, curly, organic, visible and attractive electronics. This class will explore materials, components and construction techniques for successfully integrating soft materials with standard electrical supplies. The results will be light, thin, flexible, durable, aesthetic, and even expressive circuits. We will also cover techniques for integrating the Arduino Lilypad microcontroller and XBee radio communication to create interactive and social objects. Students will develop wearable devices and accessories as a means of self-expression and communication. Explore the relationship between the body, fashion, technology and social interaction. Experiment with materials and objects, and develop concepts to refine, construct and test. Make t-shirts that interact with each other, objects that light up in the dark, toys that talk to each other, or artwork that responds to a user's movements. The possibilities are almost endless.

VOXEL ENGINES

(Poindexter)

Imagine an entire world small enough to carry in your pocket. Are you with me? If you can imagine it, you are almost there. "Voxels" make this possible; they are not magic; they are, in fact, totally logical. Imagine the world is made of little chunks. These chunks are volumetric pixels or voxels. Voxel engines are used to represent 3D data in scientific applications and to create procedurally generated terrain in the game Minecraft. In this class, we will have one project for the whole year: to create a voxel engine that procedurally generates terrain from the ground up in Java. We will start by learning to make a single block out of a set of points in 3D space, to package these points as a mesh and to get them to show up on screen. From there, we will add the ability to change the appearance of blocks, build randomly generated surfaces out of clumps of blocks and add caves and biomes. As part of our efforts, we will encounter concepts and strategies that are useful in programming in general. Having a basic understanding of object-oriented programming is a prerequisite for taking the class. **Prerequisite(s)**: Programming 1 or permission of the instructor.