C&! Course Descriptions
2025

Welcome to *C&!*!

We want you to have a *great* experience, learn good stuff, and make great friends for life. For that, it is important that you choose the right classes, ones which interest you and are at the right level. To accommodate the big age range at C&! we offer a wide range of classes. Read through the descriptions carefully and find classes that are right for you! We will make a great effort to get every student classes that are right for their level and interests. Class selection will start a few days after class description is made available.

Courses range from *Accessible*, that anyone at camp can enjoy, through *Intermediate* courses where some mathematical maturity is assumed (and may require some prerequisites), and all the way to *Prerequisites* courses that require more advanced prerequisites. The specific prerequisites depend on the course, and the details can be found in the description of the individual courses. Some courses fall under more than one category.

These divisions are meant to help you know what to expect, but remember they are imprecise! Also remember, just because a course is labeled *accessible* doesn't mean it is easy, and just because it is labeled *intermediate* doesn't mean it has no prerequisites! In every course we have the ability to take students as deeply into advanced mathematics as they want to go. Read the description carefully to find courses that are interesting for you and of the right level.

You'll find tags next to each course name. Here's what they mean.

H	Hands-on course where you'll be making things and get out of your seat as part of learning
P	Pencil and paper work will be an important part of learning in this course
L	Course with significant lecture component
C	Computer-based course, laptop needed
E	Exploratory course where students are offered problems and staff support, but are expected to self-direct their work

Once students start registering, we will update some courses based on demand and send an updated list.

And now, to work! Click on the course name to see the full description:

Accessible	Intermediate	Prerequisites
How Languages Change: Indo-European Historical Linguistics (LP)		
How Languages Work: Sound & Syntax (LP)		
Flavors of Infinity (HL)		
Math of Structure for junior (HP) mathematicians and makers		
Competition Geometry (HL)		
High School Geometry (HL)		
Numbers and Games (P)		
Number Sense (HL)		
The Euclidead algorithm (PL)		
The Shape of Thought: (H) Bits, Bytes, and Information		
Probability Puzzles (LP)		
Graph Theory (LP)		
Induction (PL)		
Mathematics of Fairness (HL)		
Recurrence (P)		
How Languages Mean: Language & Logic (LP)		
Chromatic Graph Theory (LP)		
	Continued Fractions (LP)	
	Quantum Mechanics (LP)	
	Geometry and Relativity (HPC)	
	Push, Pop, Parse (LH)	
	Generating Functions (E)	
	Quaternions (LP)	
	The Signal Protocol (L)	
		Math of Structure for older (HPL) mathematicians and makers
		Statistical Physics (LP)
		Error-Correcting Codes (PL)
		λ Calculus (PL)
		Quantum Computing 1 (PLC)
		Incompleteness (LP)
		How Computers Talk: LLM, Math, & Mind (LCP)

How Languages Change: Indo-European Historical Linguistics (Todd Krause, all levels)

Do you speak English like Shakespeare? Do you even speak it like your parents or your teachers? Take some time to look at the differences and think about how those changes come about.

And did you know English is related to German? Look at some common words to check: Vater 'father', Mutter 'mother', zwei 'two', Hand 'hand'. But it's also related to Italian, Greek, Russian, and even Hindi. How does that work? What does it mean for languages to be "related"?

This class will look at how Historical Linguistics describes relationships among languages. We will mine English for clues as to how languages evolve over time. We will compare Old English to other ancient languages looking for similar patterns in both grammar and vocabulary: maybe Latin? Ancient Greek? Gothic? (Not just a way of dressing...) Sanskrit? Old Church Slavonic? Tocharian? (Is that actually a thing?...) Get ready to climb the family tree and learn some new – old – languages!

Open to students of all backgrounds and ages.

Style: Lecture (L), Pencil & Paper (P) (Back to top)

How Languages Work: Sound & Syntax (Todd Krause, all levels)

Consider: you have some crazy thought. By some process you can't explain in detail, you pass this thought into a jumble of sounds. Those sounds smash into somebody's ear and – poof! – suddenly that thought's in their head too! How in the world does that work?!? The trick is, the sounds aren't such a jumble after all. There's method to the madness: there's a system, and somehow we all carry around a copy of this system in our heads. Are all those copies the same? How much can they differ and still work? What kinds of systems are there in the world? What do they have in common?

This class will provide an introduction to the basics of modern linguistic analysis. What are nouns? You've seen them in English, but do all languages have nouns? Do verbs work the same way in all languages? In this class we will learn to analyze the fundamental components of a language: the different types of words that a language has. We will also explore how we identify the rules by which the elements fit together. What is the system we all carry around in our heads? The real challenge facing you as a budding linguist: how can you figure all this out for a language you don't even know yet?!? You might be surprised how some of the thinking you use in mathematics will keep you in good stead when analyzing languages. So come explore one of the great mysteries of the human mind: how it creates language!

No particular linguistic knowledge assumed. But you'll want to be comfortable with logical thinking and banging your head against concepts and data that might not immediately seem to make sense. Some familiarity with the basic elements of English grammar – nouns, verbs, conjunctions... participles?... clauses? – will be helpful, though not required.

Style: Lecture (L), Pencil & Paper (P) (Back to top)

Flavors of Infinity (Tanay Mehta, accessible)

To infinity and beyond–but what actually lies beyond? In this course, we'll consider what it means to be infinite in various situations. Are some infinities bigger than others, and how would you even tell? When numbers run forever, does "how many" mean the same thing as "who comes next," or do counting and ordering start to disagree? Can we use the idea of infinity to solve problems in geometry? Spend the week exploring these questions and learn about the consequences of the flavors of infinity. (Back to top)

Mathematics of Structure for junior mathematicians and makers. (Jason Cantrella, accessible)

In this course, we will explore the geometry and math of building rigid symmetric "tensegrity" structures from a hands-on point of view. Along the way, we'll learn about orientability, symmetry groups, and make some counting arguments in rigidity theory. Very few formal prerequisites are required, though some comfort with building and crafting is desirable. (This is a not a good choice for students with physical disabilities which prevent them from safely using small hand tools.)

Both courses (and hopefully) everyone/anyone else who wants to play (parents especially welcome) are invited to join in building a large tensegrity during camp.

Laptop welcome but not required. (Back to top)

Geometry for Math Contests (Vinay Kalyankar, accessible)

We begin with intuitive understanding of parallel lines, angles, and circles and build the formulas of areas of different shapes. We will delve into circle theorems and look at the different centers of a triangle. Utilizing the Pythagorean theorem and trigonometric definitions we will learn a few identities generally useful for math competitions. We will specifically employ Pythagorean triples to simplify competition problems.

Students will need a pencil, a triangle, a ruler, a protractor, and a compass. (Back to top)

Prep for High School Geometry (Vinay Kalyankar, accessible)

Want to get ahead in the game? This class will begin with fundamental constructions and measurements and lead to the results encountered in high school geometry. You will have the opportunity to discover plane geometry theorems from the intuitive knowledge of parallel lines, angles, and circles. By utilizing the connections between proportions and similar triangles, we define trigonometric functions. We will use Pythagorean theorem and discuss a few identities resulting from the definitions of trig functions. If time permits we will use the unit circle in the coordinate plane and see trigonometric functions in a new light.

Students will need a pencil, a triangle, a ruler, a protractor, and a compass. (Back to top)

Numbers and Games (David Metzler, accessible)

How is a game like a number? How do you "add" two games to make a new one? How could you say that one game is "larger" or "smaller" than another? In this course we will explore some aspects of John Conway's theory of combinatorial games, and how they relate to his theory of surreal numbers. We will spend most of our time playing certain games that are designed to bring out the basic ideas of both theories, while always looking for patterns. We will see how numbers tell us about when games are won or lost, and how games tell us about the existence of new kinds of numbers, such as infinity minus one or one over infinity. (Back to top)

Number Sense (Wendy K. Tam, accessible)

Fractals, Pascal's Triangle, figurate numbers... What seems like simple arithmetic can suddenly give rise to all sorts of fun. In this class we will put pencil (and pen and marker and scissors and tape) to paper and see first hand what structures arithmetic can produce and how it happens. Once we see patterns and structure, we will start learning how to formalize our observations and move from conjecture to proof. (Back to top)

The Euclidean algorithm (Rolfe Schmidt, accessible)

In this class we will look at one of the most fundamental algorithms in Mathematics: Euclid's algorithm for computing the Greatest Common Denominator (GCD) of two natural numbers. If you have seen GCDs before, you may be wondering why computing them is so important. If you have those doubts, then this class will change your mind! We will start by using it to solve Diophantine equations (algebraic equations over the integers) and then see how these solutions, seen through a modern lens, let us perform the basic operations that underlie much of modern cryptography. (Back to top)

The Shape of Thought: Bits, Bytes, and Information (Hadas Zeilberger, accessible)

What do the letter 'A', the number 9, and the color red have in common? You can write them, draw them, speak them, and each one carries meaning and contains information. In this class, we'll explore how information is represented, transmitted, and transformed. We'll start with the bit, the tiniest unit of digital thought, and see how bits team up to make letters (like in ASCII), colors (as in RGB), and more. You'll learn how compression lets us shrink files without losing the story, and how encryption scrambles secrets so only the right eyes can read them. Come discover how everything from memes to music, from messages to movies, can all be represented by a clever dance of 0s and 1s. (Back to top)

Probability Puzzles (James Degnan, accessible) (Back to top)

We'll review basics of probability such as conditional probability, Bayes Theorem, and random variables, to look at some classic probability problems such as The Monty Hall problem, the St. Petersberg Paradox, Gambler's Ruin, and the probability that a stick broken in two places can have the three pieces form a triangle.

Graph Theory (James Degnan, accessible) (Back to top)

This is an introduction to graph theory, where graphs are vertices with edges connecting them. Such graphs can be used to represent social networks, evolutionary trees, and security cameras in a building. We will cover topics such as Eulerian cycles (with an application to the Bridges of Königsberg), Hamiltonian cycles, and the traveling salesperson problem.

Induction (Vivian Fang, accessible) (Back to top)

Let's learn how to prove statements for (countable) infinities, without doing infinite work! Concepts: proofs by induction, strong induction. We will then go through some of neat proofs in computer science that use induction: stable marriage, graph algorithms, and if we have time, balls-into-bins.

Recurrence (Matthew Cho, accessible)

How many moves does it take to solve the Towers of Hanoi puzzle with 64 disks? How many regions can you get when you cut a circle with 100 straight lines? How many 30-bit bit strings are there with no neighboring "1"s? How do you pick a seat for the Josephus Problem? All of these problems have similar solutions: we can solve the problem by brute force for small numbers, then we can find a rule that lets us write a solution for a bigger number in terms of smaller ones. These rules are called recurrences and they are a powerful, general, and fundamental part of discrete mathematics. In this class we will combine individual and collaborative problem solving to uncover general principles in the mathematics of recurrences. (Back to top)

Mathematics of Fairness (Wendy K. Tam, accessible)

Voting involves the "adding up" of votes. Representation is about "dividing" a small set of representatives among a larger group of constituents. Both of these "mathematical exercises" attempt to achieve fairness in some sense. In this course, we will explore the mathematics behind notions of fairness and discover that the mathematics of fairness is simultaneously simple as well as as complex as any mathematics you will encounter. (Back to top)

How Languages Mean: Language & Logic (Todd Krause, accessible/intermediate)

I'm writing words. You're reading the words. You're saying them in your head. And they mean something to you. How in tarnation does that happen?!? What does it mean for words to have meaning? It turns out that, if you try to answer that question carefully, you very quickly end up discussing logic. Mathematical logic. Yes, I mean you start doing math. Huh? That's pretty neat – you start out studying how people do language, and you end up saying just as much about how they do math!

Have you ever thought about it? If math is founded on anything concrete, it's certain rules of logic. But we int erpret logic through the medium of language. The way we construct meaning in language lies at the very crux of how we construct logical systems like math. Come explore the basic concepts of what linguists call semantics, the study of meaning. Together we'll explore some of the basic... and maybe advanced... logical concepts used in its study.

No particular prior knowledge of linguistics is assumed. But the class How Languages Work provides a good foundation for the type of thinking we'll apply here. We'll also build up, or refresh our memories of, logical concepts as we go. If you have exposure to implications (if-then statements), sets, variables, and functions, then you'll be off to a good start.

Style: Lecture (L), Pencil & Paper (P) (Back to top)

Chromatic Graph Theory (Jeremy Van Horn Morris, accessible/intermediate)

A mathematical graph is a combinatorial object described by a collection of points (called vertices) and lines between them (called edges). Mathematical graphs are of huge importance to both math and computer science but their study often leads to strange and difficult combinatorial questions. One branch of Graph Theory asks about how to "color" a graph subject to some constraints. The famous Four Color Theorem is really a problem in chromatic graph theory. We'll take a look at this connection and other strange problems of the field. (Back to top)

Continued Fraction Expansions (Yo'av Rieck, Intermediate)

Continued fraction expansions are a mean of representing numbers; it is essential for the representation of irrational numbers (and in fact can be used to *prove* that certain numbers are irrational!). But it is also useful when we wish to approximate *rational* numbers. For example, in 1680 Christiaan Huygens designed a mechanical planetarium. In order to make the task feasible, he had to find simple rational numbers that approximate the motions of the planets and their moons, which he was able to do using — drumroll, please! — that's right, continued fraction expansions! He completed the construction successfully with the help of Johannes van Ceulen, a clockmaker.

In this course we will explore this algorithm and learn how to run it by hand using pictures (namely, using the *Farey graph*). That's right: you will find the approximations by doodling with crayons! We may also get to some more advanced topics, showing that continued fraction expansions not only give good approximations, but in fact they give the *best possible* approximations.

Surprisingly, this algorithm ties back to the Euclidean Algorithm! The two courses complement each other nicely and students are encouraged to take them at the same time. (Back to top)

Quantum Mechanics (Julien Piet, Intermediate) (Back to top)

This course provides an introduction to the fundamental principles of quantum mechanics and their applications. Students will learn how matter and energy behave at the atomic and subatomic scale, where classical physics no longer applies. We'll explore concepts such as wave-particle duality, quantum superposition, the uncertainty principle, and quantum entanglement. Notions of algebra are helpful. No background in physics is necessary.

Geometry and Relativity David Metzler, Intermediate)

You may have heard of "spacetime", but what is it? Well, it's where we all live! How should one think about it? In this course we will discover that it is best understood as an unusual kind of geometry, where certain rules are the reverse of what we expect. We will begin by making a careful analogy between ordinary Euclidean geometry and the geometry of spacetime in two dimensions, as described by Einstein's special relativity (interpreted mathematically by Hermann Minkowski). We will focus particularly on some of the supposedly "paradoxical" results of special relativity and see how they are not paradoxes at all, just natural features of Minkowski geometry.

Students should know basic Euclidean and analytic geometry (up through confidence with the distance formula), how to graph an equation in the plane, and how to do basic vector operations.

(Back to top)

Push, Pop, Parse (Daniel Hader, Intermediate)

How do programming languages efficiently evaluate expressions like "$1+(3\times(5-2)+7)\div4$"? How do they implement recursive function calls? What do these questions even have in common anyway? The answer to all of these questions is a delightful data structure called a "stack". In this course we'll learn about stacks and their role in computing with a particular focus on the problem of parsing expressions. We'll also learn two different ways of writing arithmetic expressions without the need for parentheses or PEMDAS (Łukasiewicz notation and abstract syntax trees), and we'll see how stacks enable functions in code to call themselves. Finally, by the end, we'll have built our own arithmetic expression parsers in Python.

This course will require students to bring a computer installed with Python version 3.10 or greater.

Tags: Lecture, Hands On

(Back to top)

Generating Functions (Matthew Cho, Intermediate/Prerequisites)

This course introduces generating functions, a powerful tool in counting and probability. We'll start with the definition of generating functions, and apply them to problems that are difficult to approach directly. We will apply generating functions to solve recurrence relations, including the explicit formula for the Fibonacci numbers, and end by using generating functions to prove facts about partitions of integers.

This course assumes familiarity with algebra and knowing what a binomial is, but there will be problems hard enough to challenge even the strongest students.

(Back to top)

Quaternions (Jeremy Van Horn Morris, Intermediate/Prerequisites)

The complex numbers were just starting to be developed in the 16th and 17th centuries but they initiated a radical shift in our understanding of the algebra of numbers and polynomial equations. This led to a broadened understanding of functions and analysis, adding elements of geometry and topology, which then paved the way for a mathematical unification of the electric and magnetic fields and later quantum mechanics. In order to allow the complex numbers to occur, we had to break some rules of algebra and arithmetic. A similar evolution began in the 19th century when Hamilton broke even more rules and landed upon the quaternions, a four-dimensional number system that we use today. At the cutting edge of research, the quaternions are connected to the quantum mechanical notion of spin and we use them in physics, engineering, computer gaming and graphics to encode the symmetries of space. If you know and enjoy the rules of algebra, you have the background to break them and see what else is possible! (Back to top)

The Signal Protocol (Rolfe Schmidt, Intermediate/Prerequisites)

The Signal Protocol How can two people communicate securely over an insecure network? What does "secure" even mean here? This class will show you how cryptographers think about secure messaging today. After a quick review of our cryptography toolkit, we will build the obvious secure messaging protocol. Then we will attack it and realize that it wasn't as secure as we'd like. So we'll change our definition of security and design a new protocol. But guess what? We'll attack that one too. Rinse and repeat. After a few iterations, though, we will end up with a protocol that has the same basic design as the Signal Protocol, which is used to protect the communications of billions of people around the world today. If there's time, we may even have a peek at Signal's post-quantum protocol updates.

Prerequisites: Elementary number theory including modular arithmetic. Basic knowledge of symmetric key cryptography including block ciphers, hash functions, and message authentication codes. Understanding of Diffie-Hellman key exchange.

Intermediate enrollees: Intermediate students who don't feel fully prepared for the class but are comfortable feeling a bit lost sometimes may also get a lot out of this class. There will be plenty of chances to get un-lost as we move from protocol to protocol.

Pre-camp preparation: If you don't have the cryptography prerequisites there is time to pick them up! I highly recommend Dan Boneh's Coursera class – it's more than you need for this class, but you will be prepared to see the subtleties. For Diffie-Hellman key exchange, there are many online presentations with different styles. Here is a video that could get you started.

Computers welcome! There will be an option to program, but it is not required. (Back to top)

Mathematics of Structure for older mathematicians and makers (Jason Cantrella)

Mathematics of Structure for older mathematicians and makers. In this course, we'll use some linear algebra and computation to deeply understand the design and construction of tensegrity structures. We'll cover the main theorems of rigidity theory, explore computer implementations with Sage and Mathematica, and spend a day exploring the Maxwell-Cremona correspondence and the proof of the Carpenter's Rule theorem. Students should understand vectors and matrices.

Everyone/anyone else who wants to play (parents especially welcome) are invited to join in building (together with the students from *Mathematics of Structure for junior mathematicians and makers*) a large tensegrity during camp.

Laptop welcome but not required.

Prerequisites: linear algebra (vectors, matrix multiplication and matrix-vector multiplication, solving linear equations). (Back to top)

Statistical Physics (Julien Piet, Prerequisites)

This course provides an introduction to the fundamental principles of statistical physics. Students will learn how to derive the macroscopic properties of systems from the microscopic behaviors of their individual components. Key topics include temperature and entropy. The course aims to provide a comprehensive understanding of how microscopic interactions give rise to observable phenomena. A basic understanding of calculus is helpful. No background in physics is necessary, but knowing some basics can be helpful. Familiarity with basic probability concepts is also beneficial. (Back to top)

Error-Correcting Codes (Hadas Zeilberger, Prerequisites)

Have you ever tried talking with your mouth full and still been understood? If so, your listener was performing error correction–reconstructing your message despite the noise. The same phenomenon is used to send pictures from space to earth, and to store music on your computer. In this course, we'll explore the underlying object, linear error-correcting codes. We'll begin by building up the mathematical tools we need: finite fields, vector spaces, basic linear algebra and a little bit of graph theory. Then we'll dive into what makes a code "good" and examine some examples of "good" codes. Finally, we will explore applications of error-correcting codes; including noisy transmission (e.g for sending an image from space to earth), and data storage.

Prerequisites: Linear Algebra (matrix multiplication and determinants) (Back to top)

λ Calculus (Vivian Fang, Prerequisites)

We'll show how any computation can be expressed by a function, and dig into what "undecidability" means. We'll cover concepts including the Church-Turing thesis, the halting problem, Gödel's incompleteness theorem, and Rice's theorem. (Back to top)

Quantum Computing (part 1) (Tanay Mehta, prerequisites)

Quantum computers exploit the peculiar nature of quantum mechanics to perform calculations in a novel way that is sometimes more powerful than classical computers. In this advanced course, we'll learn the mathematical basis for manipulating "qubits" and understand how they behave differently from traditional "bits". We'll also learn some simple algorithms, and program them ourselves on a simulator called qiskit. We will program in python 3 (no experience is needed). Students need to bring a laptop (any laptop will do).

Prerequisites: Probability theory, linear algebra (matrices, determinants, etc). (Back to top)

Incompleteness (Daniel Hader, Prerequisites)

The 1930s marked a critical point in the history of mathematics. The century prior saw the development of symbolic logic and set theory, tools providing a rigor unlike anything before; Whitehead and Russell had recently published "Principia Mathematica", building essentially the whole of mathematics from logic alone; and Gödel himself had just shown that the proofs of all valid sentences in first order logic could be effectively enumerated. Ostensibly, the study of mathematics was reaching it's culmination as something to be fully automated; indeed, many smart people genuinely believed humanity was on the brink of "solving" mathematics. This couldn't have been further from the truth however and, during this time, a series of results were published by the likes of Gödel, Tarski, Gentzen, and Church demonstrating beautiful and downright bizarre limitations on the parts of mathematics accessible to finite procedures.

In this course, we will explore two of these results: the celebrated incompleteness theorems of Kurt Gödel. We'll start with an introduction to first order logic and learn how to formalize theories like those of arithmetic and sets. Then we'll see how we can use these theories as a sort of mirror, allowing our system of mathematics to "look at itself" figuratively speaking. Next, we'll see how this "mirror" allows us to construct a strange mathematical sentence whose meaning forbids both its own proof and disproof. Finally, we'll explore some of the consequences of the existence of such sentences.

(Back to top)

How Computers Talk: Large Language Models, Math, & Mind (Todd Krause, prerequisites)

Welcome to the Age of AI! Now computers can think! Never has p_{Doom} seemed so high. . . have you watched your Battlestar Galactica lately? Artificial Intelligence is getting pretty scary.

Wait. . . can computers think? Well, they can talk! And their talking has become really good. Of course they can also say stuff that's just plain bonkers. But, be real: so do humans. So that's not exactly proof of non-sentience.

The plan for this course is to drag you kicking and screaming through the design of a Large Language Model (LLM), like ChatGPT, Gemini, or Claude. Of course, those models are. . . well, large, and our computers are small. So I can only walk you through the basic architecture behind the models and some toy implementations. Along the way, we'll look at the linguistic insights that underlie the components of LLMs, and we'll learn about the mathematics that turns those linguistic ideas into executable code.

The upshot is that you should both understand enough about LLMs to implement a more refined version on your own, and also have a solid enough grounding to navigate the debate about language and thinking that these models inspire.

We're going to look a lot at linguistics, mathematics, and programming. For the language aspects of our discussion, the class How Languages Work will be particularly helpful. For mathematics, we'll be working mostly in the fields of linear algebra (vectors, matrices, dot products) and some calculus (derivatives and partial derivatives). The coding will happen in Python, in Jupyter or Colab Notebooks. You're not expected to be a practiced hand at any of these topics: we'll discuss the linguistics, I'll show the calculations, and all the code will have line-by-line comments. But hopefully you'll have the ambition to wade through the material and contribute to the discussion. 'Cuz this is the world you're going to be living in!

(Back to top)