The CME Project: Some Distinguishing Features

The *CME Project*, developed by EDC's Center for Mathematics Education, is a coherent, four-year, NSF-funded high school program designed around how knowledge is organized and generated within mathematics: the themes of algebra, geometry, and analysis. Many standard curricula look at each of these areas as sets of results and techniques. Many integrated programs look at them as threads that run through varying contexts. The *CME Project* sees these branches of mathematics not only as compartments for certain kinds of results, but also as descriptors for *methods* and *approaches*—the habits of mind that determine how knowledge is organized and generated within mathematics itself. As such, they deserve to be centerpieces of a curriculum, not its byproducts.

Other important parts of the discipline—probability, statistics, combinatorics, number theory, measurement—are integrated into these themes.

The primary goal of the *CME Project* is to develop in students robust mathematical proficiency. To achieve this, the *CME Project* strikes a balance between the common wisdom and tradition in this country—that students need to focus on one piece of mathematics at a time—and what has been learned about the added value of seeing connections among mathematical topics and to fields outside mathematics. The program builds on lessons learned from high-performing countries: develop an idea thoroughly and then revisit it only to deepen it; organize ideas in a way that is faithful to how they are organized in mathematics; and reduce clutter and extraneous topics. It also employs the best American models that call for struggling with ideas and problems as preparation for instruction, moving from concrete problems to abstractions and general theories, and situating mathematics in engaging contexts (including mathematics itself). The *CME Project* is a comprehensive curriculum that meets the dual goals of mathematical rigor and accessibility for a broad range of students.

The CME Project provides teachers and schools with a third alternative to the choice between traditional texts driven by low-level skill development and more progressive texts that have unfamiliar organizations. The *CME Project* gives teachers the option of a problem-based, student-centered program, organized around the mathematical themes with which teachers and parents are familiar.

The program also employs some unusual and effective approaches to mathematical topics—approaches that have been tested and refined, in some cases for several decades, by teachers and others affiliated with the program. The purpose of this note is to describe some of these approaches.

Solving Simple Equations

In the Algebra 1 course, students begin to make the connection between finding solutions to equations and finding inverses for functions. Before any of the formalisms about solving linear equations, they use a method we call *backtracking* to solve equations like this

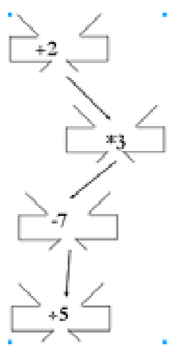
$$\frac{3(x+2)-7}{5} = 4$$

The course presents such equations with descriptions like

> "When I took a number, added 2, tripled the result, subtracted 7 from the answer, and divided the result by 5, I got 4. What number did I start with?"

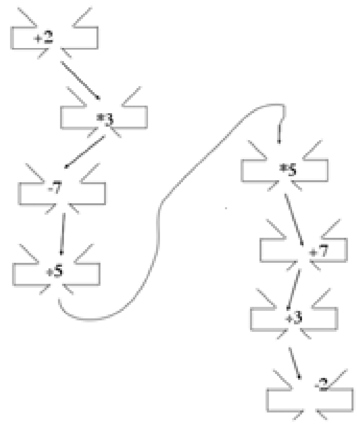
So, the left-hand side becomes a description of an *algorithm*, a function defined by a sequence of arithmetic calculations. Students model this algorithm in many ways; one useful representation is as a *machine network*:

This machine image is useful as a starting point, especially if students build computational models of functions on their calculators. Later, in Algebra 2, we move from the machine metaphor to the more robust notion of function as pairing, so that students begin to see that a function is defined by its behavior.



A machine model for $\frac{3(x+2)-7}{5}$

Students practice running several inputs through the network, and then we ask them to "pull back" an output to get the corresponding input. To do this, they do the "inverse steps in reverse order," finding a solution of the equation. In fact, as an extension, we ask them to build a network that solves the equation $\frac{3(x+2)-7}{5} = k$ for any value k of the right-hand side:



“Undoing” the algorithm

In another direction, one that connects to expression simplification and equality of functions, we ask students to find a simpler network that produces the same input-output pairs as the original network.

Algebra Word Problems

The difficulties that high school students have with algebra word problems are legendary. The quintessential word problem (“Mary is 10 years older than her brother was 5 years ago . . . ”) is the topic of cartoons and jokes. Teachers have devoted a great deal of effort to exposing the roots of the difficulties people have with word problems. Two very common perceptions are that students have difficulty with word problems because

- they have a general difficulty reading
- they are often not familiar with the contexts described in the problems.

But an analysis by some middle and high school teachers in Woburn MA showed that there’s got to be more to it. They observed that the following problem

> Mary drives from Boston to Washington, a trip of 500 miles. If she travels at an average rate of 60 MPH on the way down and 50 MPH on the way back, how many hours does her trip take?

causes no difficulty with prealgebra students who understand the connection between rate, time, and distance. But this problem

> Mary drives from Boston to Washington, and she travels at an average rate of 60 MPH on the way down and 50 MPH on the way back. If the total trip takes $18\frac{1}{3}$ hours, how far is Boston from Washington?

is baffling to many of the same students a year later in algebra class. This analysis led to an effective method—that we call *Guess-Check-Refine*—for solving these kinds of problems. Here's how it works for the second problem above:

Guess-Check-Refine is different from the well-known Guess and Check strategy for finding solutions or approximate solutions to numerical problems.

The first step is to guess at an answer; suppose Boston is 500 miles from Washington. The purpose of the guess is *not* to stumble on a right answer; rather, it's to focus students on the steps they take to check the guess. So, if the guess is 500 miles, then Mary takes $\frac{500}{60} = 8\frac{2}{3}$ hours to drive down and $\frac{500}{50} = 10$ hours to get home. The total trip is $18\frac{2}{3}$ hours, so 500 is not the right answer, but that's OK. We ask students to be explicit about what they did to check the guess. If they are not sure, they take another guess, and another, and another, until they are able to articulate something like

In spite of our proclamations that the point is not to get the right answer by guessing, many students are at first reluctant to take a guess, fearing they'll be incorrect.

> "You take the guess, divide it by 60, then divide it by 50, add you answers and see if you get $18\frac{1}{3}$."

The generic "guess checker" is then

$$\frac{\text{guess}}{60} + \frac{\text{guess}}{50} \stackrel{?}{=} 18\frac{1}{3}$$

This method was inspired in part by some educational theories about how people "encapsulate" isolated actions into coherent processes.

This gives them the equation that models the problem:

$$\frac{x}{60} + \frac{x}{50} = 18\frac{1}{3}$$

and from here, it's "pure" algebra.

Extension

One of the main themes in the program is *extension*. Extension in the CME Project takes two forms: *algebraic* (extending operations via their defining properties) and *analytic* (extension by continuity).

Example 1: Arithmetic with signed numbers.

Students have practiced arithmetic with non-negative integers since first grade. Our approach is to extend the "number facts" that many have memorized by extending patterns in the "tables" in ways that preserve the properties of the operations. Here's a piece of the multiplication table:

	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12
12					0	12	24	36	48	60	72	84	96	108	120	132	144
11					0	11	22	33	44	55	66	77	88	99	110	121	132
10					0	10	20	30	40	50	60	70	80	90	100	110	120
9					0	9	18	27	36	45	54	63	72	81	90	99	108
8					0	8	16	24	32	40	48	56	64	72	80	88	96
7					0	7	14	21	28	35	42	49	56	63	70	77	84
6					0	6	12	18	24	30	36	42	48	54	60	66	72
5					0	5	10	15	20	25	30	35	40	45	50	55	60
4					0	4	8	12	16	20	24	28	32	36	40	44	48
3					0	3	6	9	12	15	18	21	24	27	30	33	36
2					0	2	4	6	8	10	12	14	16	18	20	22	24
1					0	1	2	3	4	5	6	7	8	9	10	11	12
0					0	0	0	0	0	0	0	0	0	0	0	0	0
-1																	
-2																	
-3																	
-4																	

The multiplication table, reoriented

Notice the reorientation to make the table look more like a coordinate system. This is on purpose. For example, by graphing the line with equation $x + y = 12$ on this table, you get a picture of all the products of integers that sum to 12. Which product is largest?

So, how could one extend the table in ways that make the patterns in the rows and columns continue? There are several ways, but not surprisingly, the one that is most natural for many students is exactly the one that ensures that the extended arithmetic works the way it's supposed to: use the rows to continue the linear patterns to the left, and use the columns to extend down. We present students with some other ideas, as well (the rows increase again to the left of 0, for example) and we investigate why such choices "break" the rules of arithmetic. In other words, this is not an exercise in "extending the pattern;" rather, it is a search for an extension that *preserves rules for calculating*.

A detailed verification that the usual extension of the multiplications and addition tables preserves arithmetic properties like associativity and commutativity is too technical for most students at this stage. The development in the program is more informal, but it is faithful to the principles of this verification.

Example 2: Integer, rational, and real exponents

Most students come from middle school with some experience with positive integer exponents; they know that 2^3 means $2 \times 2 \times 2$ and, more generally, if n is a positive integer,

$$2^n \text{ means } \underbrace{2 \times 2 \times \cdots \times 2}_{n \text{ times}}$$

In Algebra 1, we use this definition to develop rules for arithmetic with positive integer exponents:

1. $a^n \cdot a^m = a^{n+m}$
2. $(a^n)^m = a^{nm}$

If we insist that these definitions extend to all integers, we are forced to make some definitions:

This kind of extension is connected to what we call the "duck principle." To show that some expression is equal to $\sqrt{10}$, show that it is positive and that its square is 10. If it walks like a duck . . . .

- 3^0 would have to be 1 if we want rule 1 to extend:

  $$3^5 \cdot 3^0 = 3^{5+0} = 3^5$$

  but the only number that can be multiplied by 3^5 to get 3^5 is 1, so 3^0 would have to be 1.
- Similarly, 3^{-1} would have to be $\frac{1}{3}$ if we want rule 1 to extend to negative integers:

  $$3^{-1} \cdot 3^1 = 3^{-1+1} = 3^0 = 1$$

  but the only number that can be multiplied by 3^1 to get 1 is $\frac{1}{3}$, so 3^{-1} would have to be $\frac{1}{3}$.

In the same way, $3^{-2} \cdot 3^2 = 1$, so 3^{-2} would have to be $\frac{1}{3^2}$.

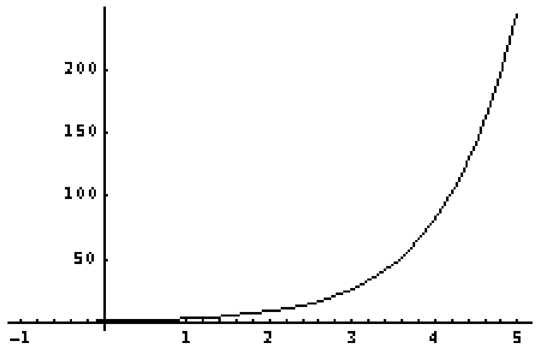
- If we want to extend the rules to fractional exponents, $3^{\frac{1}{2}}$ would have to be (by rule 2) a number whose square is 3:

  $$\left(3^{\frac{1}{2}}\right)^2 = 3^{\frac{1}{2}\cdot 2} = 3^1 = 3$$

  There are two choices here, $\sqrt{3}$ and $-\sqrt{3}$, and we make the choice that $3^{\frac{1}{2}} = \sqrt{3}$.

Similarly, $3^{\frac{5}{8}}$ is a number whose 8th power is 3^5, so we define it to be $\sqrt[8]{3^5}$. Later in the program, we show that this is the same as $\left(\sqrt[8]{3}\right)^5$

- We can extend the meaning of exponents to all integers (in Algebra 1) and all rational numbers (in Algebra 2) in this way—by forcing rules 1 and 2 to extend. But what about irrational exponents? What "must" $3^{\sqrt{2}}$ mean? For this, we use extension by *continuity*. The graph of $y = 3^x$ for all rational x looks like this:



$y = 3^x$, full of holes

Even though it looks smooth, this graph is full of holes, one over each irrational number. If we fill in the holes we get the definition of 3^x for irrational x. Put another way, $3^{\sqrt{2}}$ is the number that is approached by the sequence

$$3^{1.4}, 3^{1.41}, 3^{1.414}, 3^{1.4142}, 3^{1.41421}, \ldots$$

Where the sequence of rational exponents has $\sqrt{2}$ as a limit.

Again, this is all informal and intuitive, but the ideas can be made precise and *are* made precise in courses in post-calculus analysis.

Equations and Graphs

Many high school students do not understand the fact that one can test a point to see if it is on the graph of an equation by seeing if its coordinates satisfy the equation; equations are, for these students, a kind of code from which one can read off information that allows one to produce a graph.

Our approach to this phenomenon is to provide students with opportunities to connect equations and graphs without elaborate formalisms, using the idea that the equation of a graph is the *point tester* for the graph: it tells you whether or not a point is on the graph by checking some numerical fact about its coordinates.

For example, we ask students to find the equation of the horizontal line that passes through $(5, 1)$. They typically have no trouble drawing the line, and, when we give them several points, they usually have no trouble explaining why each is on or off the line: to see if a point is on the line, check to see if its y-coordinate is 1. So, the point-tester is $y \stackrel{?}{=} 1$, and the equation is $y = 1$.

Another example: What's the equation of the line whose graph bisects quadrants 1 and 3? The check to see if a point is on the line is that its coordinates are the same. The equation is thus $y = x$.

These are simple examples, but they reinforce the meaning of the correspondence between equations and their graphs. And the point-tester idea works well for more complex equations and their graphs—it's an idea that runs throughout the program.

Example: Equations of Lines and Slope

In Algebra 1, the point-tester idea helps students find equations for lines. The method involves a somewhat unorthodox approach to slope.

Our teaching experience tells us that "the slope of a line" approach places some undue cognitive demands on students—students are asked to think about a number (slope) that is an invariant of an infinite geometric object (the line). This is difficult for a couple reasons:

1. The invariant is not part of the geometric object itself—it is a numerical quantity derived from the geometry of the line.
2. And it is derived via a calculation that seems at first glance to depend on a *choice* of two points on the line.

Indeed, the slope of a line is an example of the *derivative* that students will study in calculus.

Our development starts with the more concrete idea of "slope between 2 points," a number that can be calculated directly from coordinates.

We use the notation $m(A, B)$ for the slope between A and B.

Our approach to equations for lines synthesizes this perspective on slope with the point-tester idea. We make an explicit assumption (that will be proved in the Geometry course):

Assumption

Three points A, B, and C lie on the same line if and only if

$$m(A, B) = m(B, C)$$

The proof requires results about similar triangles.

Suppose you are given two points, say $A = (3, -1)$ and $B = (5, 3)$. What is the equation of the line that contains A and B?. Students develop the habit of checking several points to see if they are on the line, *keeping track of their steps*. At first, we give them some points to check—say $X = (7, 6)$, $P = (1, -5)$, and $Q = (9.5, 10.5)$. In each case, they check the slope between the point to be tested and, say, B, and the see if it is equal to $m(A, B)$ (that is, 2). The generic check is that the slope from (x, y) to $(5, 3)$ should equal 2, so the point-tester is

$$\frac{y-3}{x-5} = 2$$

Some care has to be taken with the fact that x can't be 5 on the left-hand side of this equation.

This equation is then simplified and transformed into a linear equation in x and y. The course proceeds to develop fluency in sketching lines from their equations and finding equations for given lines, but only after this foundation is solid.

Functions

Students in the *CME Project* Algebra 2 course use technology in many of the same ways that technology is used in other programs: to test out conjectures, to reduce computational drudgery, to graph equations and functions, to perform statistical analyses on data, and to provide examples of theorems and results. And we also use the calculator as a context—figuring out the what's "behind" the built-in functions. For example, one lesson helps students understand the mathematics that underlies the functions on a calculator that compute standard deviation, variance, mean, and best fit lines.

We make another use of technology that is less standard: *students use technology to build computational models of mathematical objects.* One of the most important examples of this in the last two courses is that students build computational models of mathematical *functions*.

Current high-end mathematical calculators (the TI-89 family, for example) and most computer mathematics systems contain a capability that will eventually be available on most machines—something we call a *functional language*. What this means is that one can create user-defined functions—we call them *models*—in a language that is quite close to ordinary mathematical notation, and then they can use the functions as if they were built-ins. For example, to build a model of the function f defined by $f(x) = 3x+7$, you might type, in an appropriate editor, something like this:

```
f(x)
 func
   return 3x+7
 endfunc
```

The `func endfunc` syntax is particular to the TI system. Don't think of this as "programming"—think of it as "building a model of f."

Once this model has been defined, you can use it as if it were a primitive:

- you can evaluate it at inputs:

  ```
  f(5)
  >22
  ```

- You can tabulate it in the data-matrix editor or in the home environment:

  ```
  seq(f(k), k, 0, 5)
  > {7, 10, 13, 16, 19, 22}
  ```

- You can graph it. On a graphing calculator like the TI, you can assign it to one of the "$y =$" variables:

  y_1 `= f(x)`

  and then enter the graphing screen.

Notice how this reinforces what it means to graph a function: "the graph of a function f" is the graph of the *equation* $y = f(x)$.

- You can compose the model with other functions, built-in or user-defined.

  ```
  f(5^2 +3)
  > 91

  (f(1)+2)^2
  >144
  ```

So, you can calculate $f(\sqrt{2})$ or $f(\sin(45))$. If your system supports computer algebra, you can even ask for `f(2a+1)` and the system will produce some version (simplified or not) of $3(2a + 1) + 7$.

For example, in Algebra 2, students find a rule that agrees with this table:

Input: x	Output: $f(x)$	Δ	Δ^2
0	-2	2	14
1	0	16	14
2	16	30	14
3	46	44	
4	90		

The development in the text shows that outputs can be matched with a quadratic function of the form $f(x) = 7x^2 +$ something. At this point, students can build a model for $g(x) = 7x^2$, tabulate it next to the outputs, subtract the outputs of g from the outputs in the table, and then use the methods of the text to figure out the rest of the formula for $f(x)$.

But this can easily be done by hand, so what value is added?

By building the model in a language that is faithful to mathematical notation, students build an image in their minds of the function g. When students build computational models of mathematical functions, they are reviewing, expressing, and getting a chance to examine the own ideas about these functions. At one level, they are getting the benefit that generally comes from writing out one's ideas carefully and in detail: that process, by itself, helps one organize one's thinking, and externalize it enough to review and examine it. Without computational technology, students had to be satisfied with their written notes. The students who could bring these notes to life entirely in their heads would have more success than those for whom the notes just sat motionless on the paper. But when the "notes" are executable on a calculator, students can run the models they've made, verify their correctness or debug them, and even use them as parts of more complex models. Students who are not yet skilled enough to hold many parts of a model in their heads can build the parts one by one, show how they go together and, for the present, leave the orchestration to the calculator or computer. In short, computers can help students tinker with the physics of mathematics.

Most functional languages (including the current-generation TI languages) are rich enough to support piecewise and recursively defined functions. The recursively defined models are ideal for some of what we do in the program. For example, the function $f(x) = 3x + 7$ can be defined recursively (on non-negative integers) by

$$f(x) = \begin{cases} 7 & \text{if } x = 0 \\ f(x-1) + 3 & \text{if } x > 0 \end{cases}$$

This Δ notation is developed in the text. Each Δ column is the column of successive differences of the column to its left.

We make a point of the fact that students are to find *a* rule, not *the* rule. There are many rules that agree with this table.

In many parts of algebra, precalculus and calculus, students need to think of functions as *objects* so that they can transform them and calculate with them. This is notoriously difficult for many students—at least as hard as the struggle younger students have when they need to think of fractions as numbers. Out experience is that this model-building is extremely effective as a device that helps students think of functions as things in their own right.

In the physical sciences, the value of model-building and labs is well-established. Mathematical objects are objects of the imagination, so they don't often have physical models. But they *do* have computational ones, and the value of building them is similar to the value of lab activities in science.

This notation is introduced briefly in Algebra 1, and it is used extensively in Algebra 2. Look how close the computational model is:

```
f(x)
 func
   if x = 0 then
      return 7
   else
      return f(x-1)+3
    endif
  endfunc
```

Building this model, analyzing it, and experimenting with it really helps students with several key ideas:

- It helps them get quite facile with the mathematical "case" notation.
- They can compare the closed form to the recursive form, and this leads to questions about domain.
- It's often easier to see a recursive pattern in a situation (monthly payments on a loan, for example) than a closed-form one. A recursively defined model can then be tabulated and students can look for closed-form models.

Computer Algebra Systems can be used in a similar way, allowing students to build computational models of mathematical objects connected to polynomial algebra. And in a later section of this note, we look at applications of this model-building to the teaching of mathematical induction.

Proof and Justification

Mathematical proof is an essential ingredient in every *CME Project* course. The program helps students see the various roles of proof in mathematics: as a method for establishing logical connections (and hence certainty), as a means for obtaining "hidden" insights, and as a research technique. Proofs in algebra are usually *generic calculations*; for example students establish the identity

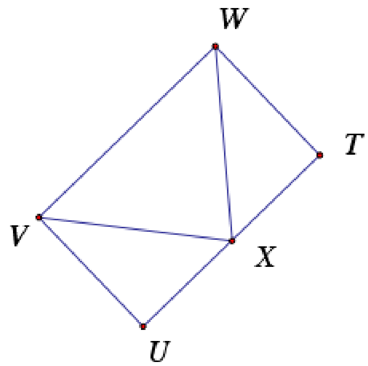
$$\left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2 = ab$$

and they use this to show that a square maximizes the area for rectangles of fixed perimeter.

Most of the lessons involve a mix of deduction and experiment.

We also develop *combinatorial* proofs of some algebraic identities, including the binomial theorem.

The program makes a distinction between how proofs are conceived and how they are presented, and the emphasis is on the former. In Geometry, for example, students study some concrete methods for finding proofs of geometric facts. One of these, the *Reverse List*, asks students to start with what they are trying to prove and work backwards by asking the questions, "What do I need?" and "What can I use?" Below is an example of a reverse list that might be created when analyzing the proof of a geometric fact:



This is a somewhat contrived problem, but problems like this are useful when students are first learning to create proofs in geometry. Later in the course, students are asked to prove geometric results of their own that are phrased as statements rather than as "given-prove" problems.

$TUVW$ is a rectangle, X is a midpoint. Prove that $\triangle VXW$ is isosceles.

The reverse list analysis is a stylized version of the process many mathematicians use to construct much more complex arguments. For the problem above, it looks like this:

The goal is for students to come up with the intuition themselves and *then* to help them formalize their ideas as a rigorous proof. As one writer puts it, "This aids both the students and the teacher, providing explicit ways of thinking that help one understand intrinsically and therefore treasure the theorems."

Given: $TUVW$ is a rectangle. X is the midpoint of TU.

Prove: Triangle XWV is isosceles.

To prove that $\triangle XWV$ is isosceles:

- NEED: $\triangle XWV$ is isosceles.
- USE: a triangle is isosceles if two sides are congruent.
- NEED: $\overline{XW} \cong \overline{XV}$.
- USE: CPCTC
  - NEED: Congruent triangles; choose $\triangle WXT$ and $\triangle VXU$.
  - USE: SAS
    * NEED first side: $\overline{TW} \cong \overline{UV}$
    * USE: opposite sides of a rectangle are congruent.
      · NEED: TUVW is a rectangle.
      · USE: Given.
    * NEED angle: $\angle T \cong \angle U$.
    * USE: all angles of a rectangle are congruent.
      · NEED: TUVW is a rectangle.
      · USE: Given.
    * NEED second side: $\overline{TX} \cong \overline{UX}$
    * USE: The midpoint of a segment divides it into two congruent segments.
      · NEED: X is the midpoint of $\overline{TU}$.
      · USE: Given.

The trick, of course, is to find the a "Use" statement that has the right conclusion and whose hypothesis can be satisfied. Doing this is often a mix of experiment and intuition. Strategies for finding the right thing to use are also developed in the course.

Once students have analyzed what they are trying to prove with this or any of the other techniques we develop, writing a proof tends to be much easier. Students can then choose to write a proof in a variety of formats: 2 column statement-reason, paragraph, outline, or other methods. This method of analyzing a proof tends to be the messiest and most time consuming, and, hence, the most tempting to skip. It is a centerpiece in our approach.

Example: Proof by Mathematical Induction.

Starting in Algebra 1 and continuing throughout the program, students looked at input/output tables like the ones below, and find both closed-form and recursive functions that fit the tables. Students became quite good at finding these functions, modeling them on their calculators, and answering questions about them.

Table A

Input	Output
0	2
1	4
2	6
3	8
4	10

Table B

Input	Output
0	0
1	2
2	6
3	12
4	20

Consider Table B as an example. A closed form function that agrees with the table is $f(n) = n(n+1)$. This is a common method: look for what you must add to the input or multiply the input by to get the output. (Students who looked for what to add to the input to get the output would notice that $f(1) = 1+1$, $f(2) = 2+4$, $f(3) = 3+9$, and decide that $f(n) = n + n^2$.)

In finding a recursively defined function that matches the table, students look at differences (or, sometimes, ratios) of outputs. For example,

Input	Output	Differences
0	0	
	$\rangle \rightarrow$	$2 - 0 = 2$
1	2	
	$\rangle \rightarrow$	$6 - 2 = 4$
2	6	
	$\rangle \rightarrow$	$12 - 6 = 6$
3	12	
	$\rangle \rightarrow$	$20 - 12 = 8$
4	20	

Since these represent consecutive even numbers, the general difference will be $2n$, with some constant (in this case 0) added on to start in the correct place. So a recursively defined function that agrees with the table is

$$g(n) = \begin{cases} 0 & \text{if } n = 0 \\ g(n-1) + 2n & \text{if } n > 0 \end{cases}$$

Students use their calculators to model both functions, and they can tabulate both of them to find further outputs of the functions.

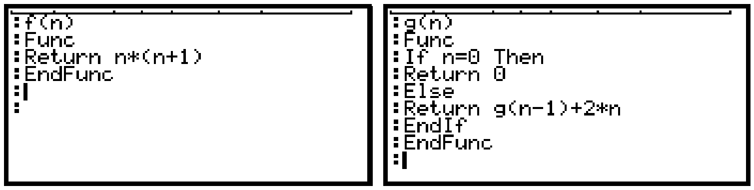


Figure 4: Two functions that agree with the table

One question that is not so easy to answer is, "Will these two functions be equal for every positive integer input?" For many inputs, students can use their calculator to check. However, the TI-89 calculator runs out of memory finding $g(44)$. So how can we be sure that $g(44) = f(44)$?

One way, of course, is to take the output the calculator gives you at $g(43)$, add $2 \cdot 44$, and compare it to the output the calculator gives you at $f(44)$. (That is, pretend to be the calculator, and do the last step yourself.) However, this becomes tedious quickly, and clearly you will never be able to check every number. Can we somehow construct an argument that the two functions must be equal for the input 44 without relying on any direct computations? Well, here is one such argument:

$$\begin{aligned} g(44) &= g(43) + 2 \cdot 44 \qquad \text{(This is just the recursive definition of } g.\text{)} \\ &= f(43) + 2 \cdot 44 \qquad \text{(We checked that } g(43) = f(43).\text{)} \\ &= 43 \cdot 44 + 2 \cdot 44 \qquad (f(43) = 43 \cdot 44.) \\ &= 44(43 + 2) \qquad \text{(We just factored 44 out of the sum.)} \\ &= 44 \cdot 45 \qquad \text{(Some arithmetic)} \\ &= f(44) \qquad (f(44) = 44 \cdot 45.) \end{aligned}$$

So we showed that $g(44) = f(44)$, though it's not clear that we did any less work than the direct computation route. Of course, now we can use the fact that $g(44) = f(44)$, and ask about what happens for the input 45. Well,

$$
\begin{aligned}
g(45) &= g(44) + 2 \cdot 45 \qquad \text{(This is just the recursive definition of } g.\text{)} \\
&= f(44) + 2 \cdot 45 \qquad \text{(We already know that } g(44) = f(44).\text{)} \\
&= 44 \cdot 45 + 2 \cdot 45 \qquad (f(44) = 44 \cdot 45.) \\
&= 45(44 + 2) \qquad \text{(We just factored 45 out of the sum.)} \\
&= 45 \cdot 46 \qquad \text{(Some arithmetic.)} \\
&= f(45) \qquad (f(45) = 45 \cdot 46.)
\end{aligned}
$$

Throughout these activities, students read the proofs, fill in steps and explanations, and discuss what they have proved. This leads to answering questions like: "Imagine that a more powerful calculator reported that $f(100) = g(100)$, but it ran out of memory computing $g(101)$. Are f and g equal at 101? How do you know?" And, eventually, "Imagine that a calculator reported that $f(n-1) = g(n-1)$, but it ran out of memory computing $g(n)$. Are f and g equal at n? How do you know?"

To answer this last question, students created proofs by mathematical induction, like the proof below for Table B:

$$
\begin{aligned}
g(n) &= g(n-1) + 2n \qquad \text{(This is just the recursive definition of } g.\text{)} \\
&= f(n-1) + 2n \qquad \text{(We imagine that } g(n-1) = f(n-1).\text{)} \\
&= (n-1)n + 2n \qquad (f(n-1) = (n-1)n.) \\
&= n(n-1+2) \qquad \text{(We just factored } n \text{ out of the sum.)} \\
&= n(n+1) \qquad \text{(Some arithmetic.)} \\
&= f(n) \qquad (f(n) = n(n+1).)
\end{aligned}
$$

This approach has many advantages, some of which we anticipated in designing the curriculum, and some which we did not.

- First, students are very clear about what they are proving. They have come up with the closed and recursive functions themselves; they know the functions came from a particular table. Moreover, they *believe* the functions to be equal. This is not true of most approaches to mathematical induction, where students are given an equality—with no rationale, background, or exploration—and asked to prove it is a true statement.

- Second, students never feel they are "assuming what they want to prove" in the induction step. Starting with the concrete problem of "extending a tabulation by one more input," they use the metaphor "I imagine they tabulated the same up to some number and then my calculator ran out of memory. Can I show that they are equal at the next number?"
- Finally, the shortcoming of the calculator shows students that they cannot, in fact, check that the functions are equal for any number they care about. If they really wanted to know if $f(1000000) = g(1000000)$, the only reasonable way to do it is by induction. (Of course, a computer would have better results, but it would *eventually* fail as well.)

Proof and deduction are core to the program, but the investigations don't usually start with ready-made polished theories. Instead, they first help students develop experience with ideas, internalizing them and building intuitions about them. *Then* students bring things together with precise definitions, axioms, theorems, and corollaries. This style is closer to how actual mathematical work is done, and it is consistent with our goal of developing *mathematical habits of mind*.

Part of what it means to work as a mathematician is to gain skill reading and understanding polished presentations. So, we sometimes break with the "experience before formality" design and ask students to read, say, a proof of a theorem and to restate it in their own words.

Statistics and Probability

Statistics and probability are integrated throughout the *CME Project* courses, and the nature of these developments is consistent with the themes of the courses. For example, in the algebra courses, students study the effect on the mean and median of a data set when simple transformations (translations or scalings, for example) are performed. They use algebra to compare several measures of deviation (mean absolute deviation, mean squared deviation, and standard deviation), and they establish the identity

$$(\overline{x})^2 - \overline{x^2} = \sigma^2$$

using algebraic properties of summations. They also study the algebraic properties of variance, especially its *linearity*. One of the main goals of the statistics strand is to help students develop a sense for when particular tools are appropriate. For example, when is the use of a regression line *not* a good idea? Another goal is for students to build an understanding of what the statistical

In this way, the study of statistics helps students in their study of algebra and Geometry (and *vice-versa*).

functions on a calculator do and, where feasible, to prove the important functional properties of these operations.

In Geometry, students study *geometric probability*. The emphasis here is on the structural similarities between area and probability—both are mathematical *measures*.

The basic premise of geometric probability is that the probability of picking a random point in a bounded region is proportional to that region's area.

Example: Regression Lines.

A good example of the *CME Project* approach to statistics is our development of lines of best fit. After some experiential work at estimating best fit lines, students begin to study some of the algebraic properties. In particular, they learn learn the definition of the best-fit line (it minimizes the sum of the squares of the deviations from the y-coordinates of the data), and they learn that the best-fit line always contains the *balance point* (centroid) of the data (the point whose coordinates are the averages of the x and y coordinates of all the data points). In Algebra 2, we state the formula for the slope of the regression line, and in Precalculus the formula is derived, making connections with properties of conic sections. This is typical of how the program proceeds from experience to precision and how it takes the mystery out of many functions built into mathematical software.

In upper grades, students can prove that the best fit line contains the centroid using properties they know about quadratic functions.

Fitting Polynomials to Data

Many middle and high school programs develop methods for fitting linear and quadratic functions to tables by using first and second differences. The CME Project pushes this theme much further.

Fitting functions to tables is a major theme in our program, one that starts in Algebra 1 and that continues through Precalculus.

In an input-output table like the one below, every output is the sum of the output for 0 and the elements in the Δ column up to the number above the output. We call this the *hockey stick* property of a difference table:

The Δ column is calculated via the successive differences of the output column. So, the outputs are the running totals of the differences.

Input	Output	Δ
0	1	−2
1	−1	12
2	11	38
3	49	76
4	125	126
5	251	188
6	439	262
7	701	

Input	Output	Δ
0	1	−2
1	−1	12
2	11	38
3	49	76
4	125	126
5	251	188
6	439	262
7	701	

Input	Output	Δ
0	1	−2
1	−1	12
2	11	38
3	49	76
4	125	126
5	251	188
6	439	262
7	701	

Hockey Sticks

In Algebra 1, students learn that constant first differences in a table imply that a linear function will fit the data, using this hockey stick idea. In this table:

Input	Output	Δ
0	3	5
1	8	5
2	13	5
3	18	5
4	23	5
5	28	5
6	33	

The hockey stick simply adds up 5s: the output for 4 is $3 + 4 \cdot 5$, the output for 5 is $3 + 5 \cdot 5$ and (if the constant differences continue) the output for n would be $3 + n \cdot 5$.

This idea is generalized in Algebra 2 and Precalculus, ending with a wonderful and classical result due to Newton that shows how to fit a polynomial of minimal degree to any data table whose inputs are in an arithmetic sequence.

This method has some surprising connections to Pascal's triangle.

What if the data inputs do not go up by the same amount? There's another classical method, called Lagrange interpolation, that lets you find the polynomial of minimal degree that agrees with any data set. We develop Lagrange interpolation in Algebra 2 and return to derive some of its properties in Precalculus.

Polynomial Algebra and Formal Calculations

One way to use polynomials as modeling tools is to look at polynomials as objects that define functions. In this way, gravity can be modeled with a quadratic polynomial, and volume can often be modeled with a cubic one. When polynomials are viewed as functions, the "x" is thought of as a *variable*, a generic element of some replacement set.

There is another use of polynomials in which the "x" is an *indeterminate*. In this view of a polynomial, the letters are just placeholders (rather than "valueholders")—what's really important are the operations *between* the letters. The difference can be illustrated with two common activities in school algebra:

simplifying and graphing. When students simplify polynomials, they are thinking of them as formal objects; the fact that $x^2 - 1 = (x-1)(x+1)$ comes from the fact that, if the right side is expanded by "the rules of algebra" you end up with the left side. Graphing, on the other hand, requires that you think of substituting values for x, or that you imagine x sweeping across some domain, producing points on a graph.

If two polynomials are equivalent formally, they define the same function. In Algebra 2, we derive an important converse to this: if two polynomial functions agree at "enough" inputs, one can be obtained from the other by the "rules of algebra."

Of course, both of these points of view are important and not completely disjoint, and we often want students to be able to move between them, sometimes in the same problem. In Algebra 1, for example, it is often helpful to forget the fact that the letters stand for numbers, although we always know that they are, in fact, just placeholders for numbers.

There are times, however, when the *form* of a calculation is more important, and we never think of replacing the letters with values. For example, at several points in the program, we ask students to investigate the distribution of possible sums when 3 (or more) dice are thrown. We then ask them to find the coefficient of x^9 when

$$(x + x^2 + x^3 + x^4 + x^5 + x^6)^3$$

is expanded. The object here is *not* to perform the expansion by hand or machine, but to calculate without calculating, reasoning where an x^9 term can occur when one multiplies out the expression. The formal calculation here models the counting problem for three dice, providing another way algebra can be used as a modeling tool. Such reasoning requires a kind of "decontextualization"—a formal approach to polynomial calculations that is important enough to deserve increased attention in the later years of high school.

The program uses formal calculations with other kinds of mathematical objects. For example, the powers of $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$ yield Fibonacci numbers.

The lines between form and value are not clean. By replacing x by -1 in the unexpanded form, students can show that the number of outcomes with an odd sum is the same as the number of even outcomes when any number of dice are thrown.

Example: Factoring.

Our approach to factoring polynomials provides students with enough practice so that they can reason about the various factoring methods we develop. Two that are worth noting are:

1. In Algebra 1, students factor quadratic polynomials with leading coefficient 1 (so-called monic polynomials), using the usual search for sums and products. In Algebra 2, we use the correspondence between the roots of a polynomial equation and the linear factors of the polynomial to reduce

the case of non-monic factoring to the monic one by a method that essentially scales the roots of the polynomial.

2. This scaling method is a special case of substituting or *chunking* to reduce complex factoring problems to more simple ones. For example:

$$25x^2 + 15x - 4 = (5x)^2 + 3(5x) - 4$$

which has the same *form* as $z^2 + 3z - 4$, and

$$x^4 + 3x^2 - 4 = (x^2)^2 + 3(x^2) - 4$$

which again has the same form as $z^2 + 3z - 4$.

Complex Numbers

Complex numbers are numbers of the form $a + bi$ where a and b are real numbers and $i^2 = -1$. Most texts introduce the symbol i as a way to solve quadratic equations with no real roots. The first example is invariably

$$x^2 + 1 = 0.$$

This masks the history a bit. In fact, the "imaginary unit" i and complex numbers first came about in the solution to *cubic* equations. A formula similar to the quadratic formula was developed in the 16th century for cubic equations. When this formula is applied to cubic equations with *real* roots, square roots of negative numbers enter the calculations, only to drop out in the end. This historical message is an important one: new mathematical systems are often invented for the purpose of solving problems in an existing system, and this "move up and them back down" process shows up all over mathematics. The mathematics of the cubic formula is quite technical for Algebra 2, but we develop complex numbers in a way that conveys the spirit of the history without the technical details. A derivation of the cubic formula will show up later in the program as an optional project.

The connection between complex numbers and coordinate geometry via the correspondence $a + bi \leftrightarrow (a, b)$ is one of the richest in mathematics, and we develop many of its consequences in Algebra 2 and Precalculus.

When points on the plane are thought of as complex numbers, the plane is called the *complex plane*.

Example: The Geometry of Complex Number Multiplication.

One wonderful result of this geometric correspondence is that one can get geometric pictures of addition and multiplication of complex numbers. Addition is connected to vector addition, but the multiplication rule is more complicated. Experiments suggest that if z and w are complex numbers, the length of zw is the product of the lengths of z and w and the argument of zw is the *sum* of the arguments of z and w. And, in fact, it's true. But getting there usually involves developing and using trigonometric identities. In the summer of 2002, a group of teachers at the Park City Mathematics Institute in Utah discovered a very simple and elegant way to see what's going on, using nothing more than similar triangles. We'll develop that method in Algebra 2 and Precalculus, and them apply it to trigonometric identities in Precalculus.

The *length* of a complex number is the distance of the corresponding point from the origin on the complex plane. It's *argument* is the angle made by the vector from the origin to the complex number and the positive real axis.

Conclusion

This description of features is not inclusive, but it is already too long. We didn't describe our approach to combinatorics, sequences and series, area, multiple angle formulas for sine and cosine, applications, or geometric optimization. But we hope that what's provided here can give readers, reviewers, and teachers a sense for the kinds of mathematics we think is important and for the well-tested approaches we take to developing that mathematics.