Computation in Everyday Mathematics

The treatment of computation in *Everyday Mathematics* involves three stages:

- In the early phases of learning an operation, children are encouraged to invent their own procedures. They are asked to solve non-routine problems involving the operations before they have developed or learned systematic procedures for solving such problems. This approach requires students to focus on the meaning of the operation. It also provides a meaningful context for developing accurate and efficient procedures.
- Later, when students thoroughly understand the concept of the operation, they examine several alternative algorithms. In this stage, students are encouraged to experiment with various algorithms and to become proficient with at least one.
- Finally, students are asked to demonstrate proficiency in at least one method. The program offers a focus algorithm for each operation to facilitate and support this phase of instruction. All students are expected to demonstrate proficiency with the focus algorithms, though they are not required to use them if they have alternatives they prefer. Focus algorithms provide a common language for further work, especially across grade levels and classrooms, and offer reliable alternatives for children who have not developed effective procedures on their own.

$$\begin{array}{r|c|c|c|c} & 3 & 17 & 12 & \\ & \cancel{4} & \cancel{8} & \cancel{2} & 6 \\ - & 3 & 9 & 3 & 4 \\ \hline & & 8 & 9 & 2 \end{array}$$

Algorithm Invention

Everyday Mathematics believes that children should be encouraged to invent and share their own procedures. As children devise their own methods, they use prior mathematical knowledge and common sense, along with new skills and knowledge. They also learn that their intuitive methods are valid and that mathematics makes sense. Ideally, children should develop a variety of computational methods and the flexibility to choose the procedure that is most appropriate in a given situation.

It is important that children have a chance to develop their own computation methods *before* they receive formal instruction in algorithms, especially standard algorithms. Learning standard algorithms too early may inhibit the development of their mathematical understanding and cause them to miss the rich experiences that come from developing their own methods. *Extensive research shows the main problem with teaching standard algorithms too early is that children then use the algorithms as* *substitutes for thinking and common sense.* For example, the authors of *Everyday Mathematics* presented the problem shown on the far right to a large number of children.

Most second and third graders immediately resorted to the standard algorithm, often failing to get the correct answer. Only a handful of children interpreted the problem as asking, "What number plus 1 gives 300?" or "What number is 1 less than 300?" or "What is the number just before 300?" and answered 299 without performing any computation.

In the modern world, most adults reach for calculators when faced with any moderately complex arithmetic computation. This behavior is sensible and should be an option for children too. Nevertheless, children do benefit in the following ways from developing their own noncalculator procedures:

- Children are more motivated when they don't have to learn standard paper-and-pencil algorithms by rote. People are more interested in what they can understand, and children generally understand their own methods.
- Children become adept at changing the representation of ideas and problems, translating readily among manipulatives, oral and written words, pictures, and symbols. The ability to represent a problem in more than one way is important in problem solving.
- Children develop the ability to transform a given problem into an equivalent, easier problem. For example, they recognize that 32 – 17 can be transformed into 35 – 20.
- In trying out creative problem-solving strategies, and in refining those strategies for use on a more permanent basis, children gain experience in decision making.

Alternative Algorithms

After children have had plenty of opportunities to experiment with their own computational strategies, they are introduced to several algorithms for each operation. Some of these algorithms may be identical to or closely resemble methods children devised on their own. Others are simplifications or modifications of traditional algorithms or wholly new algorithms that may have significant advantages to children and in today's technological world. Still others are traditional algorithms, including the standard algorithms customarily taught in U.S. classrooms.

Demonstrating Proficiency

Everyday Mathematics believes that children should be proficient with *at least* one algorithm for each operation. The program offers one *focus algorithm* for each operation to support this phase of childrens' mathematical development.

Students learn one focus algorithm for each operation. Focus algorithms are powerful, efficient, and easy to understand and learn. They also provide common and consistent language, terminology and support across grade levels of the curriculum. All children are expected to demonstrate proficiency with algorithms. Once they can reliably use the focus algorithm, children are encouraged to use it or any other method to solve problems. The aim of this approach is to promote flexibility and use of alternative methods while ensuring that all children know at least one reliable method for each operation.

$$\begin{array}{r}300\\ -\ \ 1\\ \hline\end{array}$$

Addition Algorithms

This section presents just a few of the dozens of possible algorithms for adding whole numbers.

Focus Algorithm: Partial-Sums

You can add two numbers by calculating partial-sums, working one place-value column at a time, and then adding all the sums to find the total.

Example: Partial Sums

```
                                              268
                                            + 483
Add the hundreds (200 + 400).                 600
Add the tens (60 + 80).                       140
Add the ones (8 + 3).                       +  11
Add the partial sums (600 + 140 + 11).        751
```

Column-Addition

To add using the column-addition algorithm, draw vertical lines to separate the ones, tens, hundreds, and so on. Add the digits in each column, and then adjust the results.

Example: Column Addition

```
                                          hundreds  tens  ones
Add the digits in each column.                   2     6     8
                                               + 4     8     3
                                                 6    14    11

                                          hundreds  tens  ones
Since 14 tens is 1 hundred plus 4 tens,          2     6     8
add 1 to the hundreds column, and              + 4     8     3
change the number in the tens                    7     4    11
column to 4.

                                          hundreds  tens  ones
Since 11 ones is 1 ten plus 1 one, add           2     6     8
1 to the tens column, and change the           + 4     8     3
number in the ones column to 1.                  7     5     1
```

For some students, the above process becomes so automatic that they start at the left and write the answer column by column, adjusting as they go without writing any of the intermediate steps. If asked to explain, they might say something like this:

"200 plus 400 is 600. But (looking at the next column) I need to adjust that, so I write 7. 60 and 80 is 140. But that needs adjusting, so I write 5. 8 and 3 is 11. With no more to do, I can just write 1."

Opposite-Change Rule

If you add a number to one part of a sum and subtract the same number from the other part, the result remains the same. For example, consider:

$$8 + 7 = 15$$

Now add 2 to the 8, and subtract 2 from the 7:

$$(8 + 2) + (7 - 2) = 10 + 5 = 15$$

This idea can be used to rename the numbers being added so that one of them ends in zeros.

Example: Opposite Change

- Rename the first number and then the second.

```
     Add 2.         Add 30.
   268          270          300
 + 483        + 481        + 451
                             751
   Subtract 2.    Subtract 30.
```

- Rename the second number and then the first.

```
    Subtract 7.     Subtract 10.
   268          261          251
 + 483        + 490        + 500
                             751
     Add 7.         Add 10.
```

Subtraction Algorithms

There are even more algorithms for subtraction than for addition, probably because subtraction is more difficult. This section presents several subtraction algorithms.

Focus Algorithm: Trade-First Subtraction

This algorithm is similar to the traditional U.S. algorithm except that all the trading is done before the subtraction, allowing students to concentrate on one thing at a time.

Counting-Up

To subtract using the counting-up algorithm, start with the number you are subtracting (the subtrahend), and "count up" to the number you are subtracting from (the minuend) in stages. Keep track of the amounts you count up at each stage. When you are finished, find the sum of the amounts.

Example: Trade-First Subtraction

Examine the columns. You want to make trades so that the top number in each column is as large as or larger than the bottom number.

hundreds	tens	ones
9	3	2
− 3	5	6

To make the top number in the ones column larger than the bottom number, borrow 1 ten. The top number in the ones column becomes 12, and the top number in the tens column becomes 2.

hundreds	tens	ones
	2	12
9	~~3~~	~~2~~
− 3	5	6

To make the top number in the tens column larger than the bottom number, borrow 1 hundred. The top number in the tens column becomes 12, and the top number in the hundreds column becomes 8.

hundreds	tens	ones
	12	
8	~~2~~	12
~~9~~	~~3~~	~~2~~
− 3	5	6

Now subtract column by column in any order.

hundreds	tens	ones
	12	
8	~~2~~	12
~~9~~	~~3~~	~~2~~
− 3	5	6
5	7	6

Example: Counting Up

To find 932 − 356, start with 356 and count up to 932.

356
Add 4 to count up to the nearest ten.
360
Add 40 to count up to the nearest hundred.
400
Add 500 to count up to the largest possible 100.
900
Add 32 to count up to 932.
932

Now find the sum of the numbers you added.

$$\begin{array}{r} 4 \\ 40 \\ 500 \\ +\ 32 \\ \hline 576 \end{array}$$

So, 932 − 356 = 576.

Example: Left-to-Right Subtraction
To find 932 − 356, think of 356 as the sum 300 + 50 + 6. Then subtract the parts of the sum one at a time, starting from the hundreds.

	932
Subtract the hundreds.	− 300
	632
Subtract the tens.	− 50
	582
Subtract the ones.	− 6
	576

Left-to-Right Subtraction
To use this algorithm, think of the number you are subtracting as a sum of ones, tens, hundreds, and so on. Then subtract one part of the sum at a time.

Example: Same-Change Rule
Add the same number.

Add 4.	Add 40.	
932	936	976
− 356	− 360	− 400
	Subtract.	576

Example
Subtract the same number.

Subtract 6.	Subtract 50.	
932	926	876
− 356	− 350	− 300
	Subtract.	576

The Same-Change Rule
If you add or subtract the same number from both parts of a subtraction problem, the results remain the same. Consider, for example:

$$15 - 8 = 7$$

Now add 4 to both the 15 and the 8:

$$(15 + 4) - (8 + 4) = 19 - 12 = 7$$

Or subtract 6 from both the 15 and the 8:

$$(15 - 6) - (8 - 6) = 9 - 2 = 7$$

The same-change algorithm uses this idea to rename both numbers so the number being subtracted ends in zeros.

Multiplication Algorithms

Children's experiences with addition and subtraction algorithms can help them invent multiplication algorithms. For example, when estimating a product mentally, many children begin to compute partial products: "Ten of these would be. . ., so 30 of them would be. . ., and we need 5 more, so. . ." Beginning in third-grade *Everyday Mathematics,* this approach is formalized as the partial-products multiplication algorithm. This algorithm and others are discussed in this section.

Focus Algorithm: Partial Products

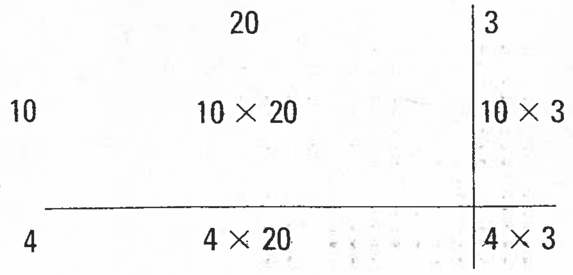
To use the partial-products algorithm, think of each factor as the sum of ones, tens, hundreds, and so on. Then multiply each part of one sum by each part of the other, and add the results.

Rectangular arrays can be used to demonstrate visually how the partial-products algorithm works. The product 14 × 23 is the number of dots in a 14-by-23 array. The diagram below shows how each of the partial products is represented in the array.

Example: Partial Products

To find 67 × 53, think of 67 as 60 + 7 and 53 as 50 + 3. Then multiply each part of one sum by each part of the other, and add the results.

	67
	× 53
Calculate 50 × 60.	3,000
Calculate 50 × 7.	350
Calculate 3 × 60.	180
Calculate 3 × 7.	+ 21
Add the results.	3,551



$$\begin{aligned} 14 \times 23 &= (10 + 4) \times (20 + 3) \\ &= (10 \times 20) + (10 \times 3) + (4 \times 20) + (4 \times 3) \\ &= 200 + 30 + 80 + 12 \\ &= 322 \end{aligned}$$

Modified Repeated Addition

Many children are taught to think of whole-number multiplication as repeated addition. However, using repeated addition as a computation method is inefficient for anything but small numbers. For example, it would be extremely tedious to add fifty-three 67s in order to compute 67 × 53. Using a modified repeated-addition algorithm, in which multiples of 10, 100, and so on, are grouped together, can simplify the process.

Example: Modified Repeated Addition

Think of 53 × 67 as fifty 67s plus three 67s. Since ten 67s is 670, fifty 67s is five 670s.

So, 53 × 67 is five 670s plus three 67s.

$$\begin{array}{r} 67 \\ \times\ 53 \\ \hline 670 \\ 670 \\ 670 \\ 670 \\ 670 \\ 67 \\ 67 \\ 67 \\ \hline 3{,}551 \end{array}$$

Division Algorithms

One type of division situation involves making as many equal-size groups as possible from a collection of objects: How many dozens can you make with 746 eggs? How many 5-passenger cars are needed for 37 people? Such problems ask, "How many of these are in that?" More generally, $a \div b$ can be interpreted as "How many bs are in a?" This idea forms the basis for the division algorithms presented in this section.

Partial-Quotients

The partial-quotients algorithm uses a series of "at least, but less than" estimates of how many bs are in a.

Example: Partial-Quotients

Estimate the number of 12s in 158.
You might begin with multiples of 10 because they are simple to work with. There are at least ten 12s in 158 ($10 \times 12 = 120$), but there are fewer than twenty ($20 \times 12 = 240$). Record 10 as a first guess, and subtract ten 12s from 158, leaving 38.

```
12)158   10  first guess
   120
    38    3  second guess
    36   ---
     2   13  sum of guesses
```

Now estimate the number of 12s in 38.
There are more than three ($3 \times 12 = 36$) but fewer than four ($4 \times 12 = 48$). Record 3 as the next guess, and subtract three 12s from 38, leaving 2.

158 ÷ 12 $\longrightarrow$ 13 R2

Since 2 is less than 12, you can stop estimating. The final result is the sum of the guesses ($10 + 3 = 13$) plus what is left over (the remainder of 2).