Arithmetic with H = -1: subtraction, negative numbers, division, rationals and mixed numbers

Thomas Colignatus
http://thomascool.eu
April 2 and May 23 2018

Abstract

$H = -1$ is an universal constant. H represents a half turn along a circle, like i represents a quarter turn. Kids know what it is to turn around and walk back along the same path. H creates the additive inverse with $x + H\,x = 0$ and the multiplicative inverse with $x\,x^H = 1$ for $x \neq 0$. Pronounce H as "ehta" or "symbolic negative one". The choice of H is well-considered: its shape reminds of -1 and even more (-1). Pierre van Hiele (1909-2010) already proposed to use $y\,x^{-1}$ and drop the fraction bar y / x with its needless complexity. Students must learn exponents anyway. The negative exponent might confuse pupils to think that they must subtract something, but the use of an algebraic symbol clinches the proposal. Also 5/2 can be written as $2 + 2^H$, so that it is clearer where it is on the number line. This approach also causes a re-evaluation of the didactics of the negative numbers. The US Common Core has them only in Grade 6 which is remarkably late. The negative numbers arise from the positive axis x by rotating or alternatively mirroring into $H\,x$. Algebraic thinking starts with the rules that $a + H\,a$ can be replaced by 0 and that $H\,H$ can be replaced by 1. Subtraction $a - b \geq 0$ may be extended into $a - b < 0$ with its present didactics, e.g. 2 - 5 = 2 - (2 + 3) = 2 - 2 - 3 = 0 - 3 = -3, but there is an intermediate stage with familiar addition $2 + 5\,H = 2 + (2 + 3)\,H = 2 + 2\,H + 3\,H = 0 + 3\,H = 3\,H$, that does not require (i) the switch at the brackets from plus to minus and (ii) the transformation of binary 0 - 3 to number -3. The expression $a - (-b)$ involves (scalar) multiplication which indicates why pupils find this hard, and $a + H\,H\,b$ is clearer. The use of H would affect the whole curriculum. There appears to be a remarkable incoherence in mathematics education and its research w.r.t. the negative numbers, which reminds of the problems that the world itself had since the discovery of direction by Albert Girard in 1629 and the introduction of the number line by John Wallis in 1673. This notebook provides a package to support the use of H in *Mathematica.* The notebook and package are intended for researchers, teachers and (Common Core) educators in mathematics education. Pupils in elementary school would work with pencil and paper of course.

Keywords

mathematics education, kindergarten, elementary school, highschool, Common Core, H = -1, negative number, division, rational number, fraction, mixed number, power, exponent, Mathematica, Wolfram language, programming, package

MSC2010

97M70 Mathematics education. Behavioral and social sciences

Cloud

A version of this notebook with package is also available at (same link, major update of notebook and packages):

https://www.wolframcloud.com/objects/thomas-cool/MathEd/2018-04-02-Arithmetic-with-H.nb

http://community.wolfram.com/groups/-/m/t/1313302

https://zenodo.org/record/1241383 or DOI 10.5281/zenodo.1241383

Contents

Start (subsection for the initialisation cells with the packages)

1. Introduction

1.1. *H* = -1

H = -1 is an universal constant. *H* represents a half turn along a circle, like *i* represents a quarter turn.

Kids know what it is to turn around and walk back along the same path.

Pronounce *H* as “ê:ta:” (ehta) or “symbolic negative one” or “symbolic negative unit”.

The choice of *H* as a symbol is well-considered: its shape reminds of -1 and even more (-1).

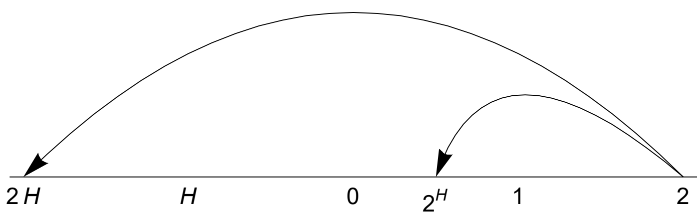
This notebook provides a package to support the use of *H* in *Mathematica.*

The literature section provides texts that discuss the use of *H.*

1.2. Key application of *H*

The key application of *H* is:

- For number x there is an additive inverse $H\, x$, with the rule $x + H\, x = 0$.
- For nonzero number x there is a multiplicative inverse x^H (“per x”), with the rule $x\, x^H = 1$.



1.3. The curriculum

I have considered writing two separate articles, one on the additive inverse (negative numbers) and one on the multiplicative inverse (exponent *H,* division, quotient), but it is better to present the integral analysis. Separate are already, and their arguments support the present analysis and the reader should not overlook those arguments: (2018a) on pronunciation of integers using the full place value system, and (2018b) with the tables of addition and subtraction with better use of place value.

The US Common Core State Standards (CCSS) (2018) have the negative numbers only in Grade 6. Holland and the UK have them in Grade 7, i.e. junior highschool. Elementary school has little experience with the negative numbers and this community will have little interest in such discussion. They have a vested interest in fractions for Grade 3-5 (i.e. in best definition: expressions with a fraction bar). Thus it is fortunate that the present discussion on the multiplicative inverse can be put into an integral analysis that highlights that the lower grades of elementary school better should not neglect the additive inverse (negative numbers). Increasing the understanding by teachers on these issues is an important step for increasing the likelihood of improvement.

Researchers, teachers and (Common Core) educators have a tradition to say that “there are no negative apples” and e.g. “there are no -3 apples”. Apparently they are not aware that eating an apple is the very cancellation represented by $x + H\, x = 0$. The equation:

3 apples + eating 3 apples = 0 apples

indicates that eating provides the cancellation, using H = *eating*, so that we really have the equivalent of -3 apples. Observe that "-3" is only a way of denoting a negative integer, and that $H\, 3$ is this very same integer. The very format of numbers like -3 might have blocked understanding of what negative numbers are, since the format suggests an object and not a process. We should however not look at objects only but also at activities, like counting and addition are activities as well. It is only accounting with credit and debet. For 2 apples + eating 5 apples = eating 3 apples, or $2 + 5\, H = 2 + 2\, H + 3\, H = 0 + 3\, H = 3\, H$, the interpretation is that you go hungry for want of eating three apples. The term eating × eating means cancelling the cancellation, or not do the eating. Having 3 apples minus eating 2 apples means that you have 5 apples, with the rule that $H\, H = 1$, like making two half turns gives a whole turn. If properly explained like this, elementary school teachers in Grade 2 should be able to master this idiom and then explain this to their students too.

The concern for the well-being of the kids is shared by all of course. The steps forward come from questioning dogma and from growing awareness that mathematics education (ME) and its research (MER) are a big mess and not a perfect and shining peak of civilisation. Remember though that a dogma prevents creative thinking that would overcome the dogma.

1.4. Redesign of mathematics education

For Mathematics Education (ME) and ME research (MER) we have (and observe the use of quotation marks):

- Conventional MER adopts (traditional) "mathematics" as given, and the research objective is to find better ways to teach this to students.
- The re-engineering approach has the diagnosis that "problems in didactics" are caused by so-called "mathematics" itself. Mathematics should be clear and convincing by itself. When students experience problems, then it isn't their fault but the fault of so-called "mathematics".

See https://zenodo.org/communities/re-engineering-math-ed/about.

Didactics of mathematics has an overall unity of (i) words (symbols), (ii) (symbolic) expressions, (iii) graphs, (iv) tables. Teaching one aspect without mentioning the other aspects is *not done*.

1.5. Readership

We use *Mathematica* to mimic what kids might do with pencil and paper. We will not assume that kids would learn about *Mathematica* before they learn arithmetic.

Our readership are researchers, teachers and (Common Core) educators, not only w.r.t. mathematics education but also from fields in education that use mathematics.

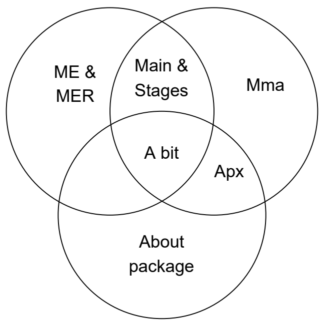
It is important to realise that mathematics education (ME) and its research (MER) have been dominated by (i) pedagogues with little insight in mathematics and (ii) mathematicians trained on abstract thinking instead of empirics. Remember the *math wars* - and much hasn't been resolved but only has gone underground. Advancement can by made by the proper combination of the clear light of day, scientific integrity, pedagogy, math, and empirics, see https://zenodo.org/communities/re-engineering-math-ed/about. An important impetus might come from educational fields that *use* mathematics, like physics, biology, psychology or economics, who are competent in math and who are aware of the importance of *empirics,* and who thus can ask critical questions and who have every reason to do so. Not only the kids but also the education in their fields suffer from the current mishap in mathematics education (ME) and its research (MER).

I also hope that the makers of *Mathematica* read this, and agree to include *H* as a universal constant indeed.

This notebook is mainly a report on content. At the same time it is a technical report on how to employ *Mathematica* and the new package for our topic. Some people in the readership will benefit much from the **Appendices** for this technical aspect.

1.6. Didactic core and technical report on programming

This topic of this notebook is mathematics education (ME) and its research (MER). This is often seen as “text *without* programming”. Our topic discusses mathematics while mathematics deals with patterns. The discussion on our topic can improve quite a lot when we use *Mathematica,* a system for doing mathematics by computer, see Wolfram (1991, 1998). This gives “text *with* programming”, both in the Main body of the text but also in the Stages that we discuss separately. This notebook also provides a report on the package, which is “text *about* programming”. The technical details are in the **Appendices**. The reader is invited to keep these three aspects on content and programming in mind, and not confuse the one with the other.



1.7. Structure of the paper

The didactic core is how pupils would work, mimicked by routines here. For teachers and researchers we must look at the properties of the routines. The main body of the text cannot avoid some issues of programming. The routines can be categorised by the 7 stages in the curriculum. The 7 logical stages create some richness in routines, and therefor we also have some repetition in the discussion, as “the same issue” might need attention within a new context again.

Section 2 gives an overview of our topic. **Section 3** looks at the negative numbers. **Section 4** gives an overview of the 7 stages of arithmetic for our topic and what *Mathematica* already provides for. **Section 5** gives conclusions. Our conclusions are rather meagre and *pro forma.* The main conclusion is provided by this notebook with its package itself: that this is a relevant topic to look into.

The main body of the text gives main points. There are two kinds of details with separate discussion.

- The separate discussions on the **stages** have the label "Stage" and not "Appendix" . The stages are linked to the US Common Core. The *classes* in the Common Core differ from the *stages,* and all differ from the Van Hiele *levels of insight*.
- The discussions on the **routines** have the label "Appendix" in their name. There is no master routine yet since we do not have a sufficient categorisation yet by stage and step within the stage.

Appendix A identifies properties of *Mathematica* that relate to the 7 stages (support or conflict).

Appendix B provides for routines that fit the intended use, using Appendix A.

Appendix C looks at the properties of the routines, and relates them to the stages in didactics.

Appendix D serves to highlight SimplifyH as a short but relevant routine.

Appendix E might have been part of Appendix A, but highlights the different treatment of rationals.

2. Overview on the inverses with *H*

2.1. Empirical theory but without testing yet

This paper distinguishes 7 stages in mathematical development of arithmetic on our topic. The *stages* differ from the *grades* and both differ from the Van Hiele *levels of insight*. The stages are linked up though with the US Common Core (2018) for kindergarten and K1-6.

It are the kids themselves who tell us what works for them. The following only develops theory. It is no use to test kids when we don't have a sound theory. Future research would involve experiments. This discussion within *Mathematica* has been quite helpful in identifying aspects that require attention for such experimenting.

The Dutch Ministry of Education, Culture and Science (2016) decided that highschool teachers are also qualified for kindergarten and elementary school. I have my hesitations on this, and restrict my *teaching* to highschool & downstream and not upstream, see also Colignatus (2015b) with my disclaimer on competence. The following only gives my *thoughts* on the subject matter for (my upstream) kindergarten and elementary school. For example, we distinguish between scalar multiplication (2 km) and grouping (5 baskets of 4 apples), and I really would not know whether this distinction is made in elementary school with proper accuracy.

There has been some testing on software though: employing the routines in different contexts helped me to improve their design and robustness. For example, the *Table of Basic Addition* now is oriented like the system of co-ordinates, and thus not upside down as in convention. Saying that 10 is higher than 1 now is also visually supported and not visually contradicted.

2.2. Didactics on division

Pierre van Hiele (1909-2010) already proposed to get rid of the fraction bar y / x in division, that causes needless complexity, and instead use $y\, x^{-1}$. The exponent -1 might confuse pupils to think that they must subtract something. However, the use of an algebraic symbol *H* clinches the proposal. Thus we get $y\, x^H$. The rule here is $x\, x^H = 1$. A direct consequence: In division of longer expressions using the fraction bar students might forget the brackets but the use of *H* would tend to generate more discipline, compare $1 / 2\, x$, $1 / 2x$, $1 / (2\, x)$ and $(2\, x)^H$.

The decimal position of 2^H on the number line can be found as:

```
2^H == 2^H × 10 × 10^H == 2^H × 2 × 5 × 10^H == 5 × 10^H ;
```

Another proposal: for "two *and* a half" we use $2 + 2^H$, which avoids the $2\frac{1}{2}$ that reads as "two *times* a half". The latter traditional notation of the “mixed number” is an exception upon the general rule that terms alongside each other (like 2 km) represents (scalar) multiplication. It tends not to be explained that the traditional mixed number format is such exception. Problematic is also that kids first learn 2 × 3 and $2\frac{1}{2}$ as general rules, and only later learn that 2 *a* or even 2*a* is possible too, which means that they first learn the exception and only later the general rule (and it is not explained which is what). This practice involves a lot of *unlearning*, which is more time-consuming than *learning* (from a blank slate).

The following is an example addition of mixed numbers. Since we are not used to these formats, the steps are mentioned. For simplicity we do 3 × 5 = 15 but we might also leave this unevaluated within brackets till we have the final expression.

```
MixedNumberH["Example"]
```

	Using H	Fraction bar
1	$(2 + 2 \times 3^H) + (3 + 4 \times 5^H)$	$\left(2 + \frac{2}{3}\right) + \left(3 + \frac{4}{5}\right)$
2	$5 + 2 \times 3^H + 4 \times 5^H$	$5 + \frac{2}{3} + \frac{4}{5}$
3	$5 + 2 \times 5 \times 5^H \times 3^H + 4 \times 3 \times 3^H \times 5^H$	$5 + \frac{5 \times 2}{5 \times 3} + \frac{3 \times 4}{3 \times 5}$
4	$5 + 10 \times 15^H + 12 \times 15^H$	$5 + \frac{10}{15} + \frac{12}{15}$
5	$5 + 22 \times 15^H$	$5 + \frac{22}{15}$
6	$5 + (15 + 7)\ 15^H$	$5 + \frac{15}{15} + \frac{7}{15}$
7	$6 + 7 \times 15^H$	$6 + \frac{7}{15}$

PM. The example uses *H* in italics also in StandardForm. This notebook uses *H* in output in TraditionalForm that generates those italics automatically. For input regular H is okay. Use input 3H because H3 would be a variable.

2.3. The number line

The use of the minus sign for both binary subtraction and the unitary negative numbers is a fluke of history (Vredenduin (1991)) that might actually be efficient in the sciences, but it is awkward for kids learning arithmetic. The minus sign is used from first grade as a binary operator $a - b \geq 0$. The extra use of the minus sign to indicate the negative numbers may require too many explanatory steps at the same time: via 0 - 3 = -3 or the scalar multiplication (-1) × 3 = -3. Who only considers to use another sign, might still lose out on the algebra of *H.*

This notebook with package investigates whether symbolic *H* might make a difference by providing for an intermediate step, and not just a step but one with major didactic relevance.

Kids would know what it is: to walk in some direction, ***turn around,*** and walk back on the same path. The following display of the number line is more than a gimmick, though we still have to wait for experiments on what would really work. Kids would understand how the *integers* 1, .., 10 look when rotated a half turn or in a mirror. Similarly for a rotation or mirroring of the *number line* of the nonnegative integers.

TextRotate[][10]

{01 ˙6 ˙8 ˙L ˙9 ˙5 ˙ㄣ ˙Ɛ ˙ᄅ ˙1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

TextReflect[10]

{01, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

The next step would be to explain to pupils that instead of such rotated or mirrored numbers we write *H x*. There would be no mystery what these numbers would be, because they would indicate the positions with respect to zero (or steps forward or backward from a relative zero). We use *H* because we don't want a conflict with their current understanding of $a - b \geq 0$.

IntegerLineH[10]

$\{10\,H, 9\,H, 8\,H, 7\,H, 6\,H, 5\,H, 4\,H, 3\,H, 2\,H, H, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

Let us identify a cookie by 1*,* with 1 written in chocolate on it. Let us identify a hungry cookie-monster by *H,* wearing a T-shirt with *H.* Let a cookie-monster get a full stomach from eating precisely one cookie, and then change to a T-shirt that says 0. The rule of the game is $x + x\,H = 0$, with x the number of cookies and $y\,H$ the number of cookie-monsters. Relevant questions are $x + y\,H = \ldots$? How many cookies are left or how many cookie-monsters go hungry ?

For example: $7 + 3\,H = (4 + 3) + 3\,H = 4 + 3 + 3\,H = 4 + (3 + 3\,H) = 4 + 0 = 4$.

The latter shows the many steps for the first time. The following gives the core for a bit more experience.

HToZero[7 + 3 H]

$$\begin{pmatrix} 7 + 3\,H \\ 3 + 4 + 3\,H \\ 4 \end{pmatrix}$$

In $7 + 3\,H = 4$, the 7 is an arrow of size 7 to the right on the number line and $3\,H$ is an arrow of size 3 to the left, and jointly they land in position 4. This is vector addition along the line.

Once we have the line we can explain to kids, and they could understand, that $a + H\,b = a - b$*,* first with only nonnegative outcomes but later also outcomes in the negative section of the number line.

$2 + 5\,H = 2 + (2 + 3)\,H = 2 + 2\,H + 3\,H = 0 + 3\,H = 3\,H$

HToZero[2 + 5 H]

$$\begin{pmatrix} 2 + 5\,H \\ 2 + 2\,H + 3\,H \\ 3\,H \end{pmatrix}$$

When a finds an inverse or cancellation in $H\,a$*,* so that $a + H\,a = 0$, then an obvious question is what cancels $H\,a$ itself. We get $H\,a + H\,H\,a = 0$. What do you see ?

$$\left\{\begin{pmatrix} 2\,H + 2 = 0 \\ 2\,H + H\,H\,2 = 0 \end{pmatrix}, \begin{pmatrix} 5\,H + 5 = 0 \\ 5\,H + H\,H\,5 = 0 \end{pmatrix}, \begin{pmatrix} 9\,H + 9 = 0 \\ 9\,H + H\,H\,9 = 0 \end{pmatrix}\right\}$$

It remains to be seen what kids might see indeed, and otherwise they might be told that $H\,H = 1$, and be shown by turning a full circle. For the game: a hungry cookie-monster bumping into a hungry cookie-monster turns all numbers into cookies. Do not forget to explain about cancelling cancellation. The basic model for reversibility is turning around in space but such games can be made reversible too.

Subtraction $a - b \geq 0$ may be extended into $a - b < 0$ with its present didactics, with e.g. 2 - 5 = 2 - (2 + 3) = 2 - 2 - 3 = 0 - 3 = -3, but the method with H (see above) provides an intermediate stage with familiar addition, that does not require (i) the switch at the brackets from plus to minus and (ii) the transformation of binary 0 - 3 to number -3.

When kids arrive at the issue of $a - H\, b$ and find $a + H\, H\, b$, they already know $H\, H = 1$.

Once the concepts and the associated algebra are mastered, the notation of the *Integers with the Minus Sign* (IwMS) or the *Negative Integers denoted by the Minus Sign* (NIdMS) (3 H is a negative integer too) (I will not use these abbreviations) would be less of a problem and less urgent. Writing 3 H for -3 is perfectly correct, also in university (when you once state $H = -1$).

The integers with the minus sign thus would only be introduced once the concept of the negative numbers has already been mastered (via H) and once the issue is seen as merely a more convenient way of writing the same.

```
Range[-10, 10]
```

{−10, −9, −8, −7, −6, −5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

It is not clear to me whether H would be a useful intermediate step indeed. Perhaps it is possible to first say that $a - b$ may also be written as $a + (-1) \times b$ and also as $a + (-b)$, and only then construct the number line, directly using the minus sign. Perhaps it suffices to train kids better on the use of brackets. Empirical research however hasn't had this focus since the negative numbers only appear so late in education.

Given all this uncertainty I do not want to take much risk and prefer the use of H at least as an analytical device to indicate the stages in education on our topic.

2.4. The additive inverse

We better reconsider the introduction of negative numbers. There is the expression $a - (-b)$ that pupils (now in junior highschool) find so difficult. This becomes $a + H\, H\, b$. Subtraction involves an element of multiplication, and is not merely "the opposite of addition" (without dwelling on what "opposite" would mean). **Section 3** will consider the issue. The following is usefully said here.

- With the unitary operation we can define the binary one, and with the binary operation we can define the unitary one: $H\, x = 0 - x$. H is superfluous and only a way of writing $0 - x$. At advanced understanding it would be a matter of taste what we prefer as the basic notion. One variable is simpler than two, but is it really ?
- For kids it should relate to what their first intuitions are. They count up and count down. But they also turn this way and back again along the circle. They know what a mirror is. This causes a stalemate in what would be "intuitive". Experiments should give the answer what works *best*, and this would depend upon time and place and culture, and such circumstantials should not interfere with the present steps in the argument.

- There is an argument of simplicity, see above for $a - b < 0$ and $a - (-b)$. However, empirical testing might show that having H as an intermediate stage only delays the command of subtraction. Kids might also first work with 0 - 3 in symbolic form, gain command, and only then switch to -3 as a notation issue only.
- The unitary operations generate the *inverses,* and this emphasizes the cancellation of terms. To me this is a strong argument, but one might hold that teachers might focus on cancellation too.

Let us look into this a bit deeper. The inverses help, via their very definitions, to focus on *how to cancel values.* "Simplify 2 + 5 H" invokes cancellation, helps to focus on *what is the largest number that can be cancelled* (in this case: which of 2 and 5 is the smallest), and gives (see above):

```
HToZero[2 + 5 H]
```

$$\begin{pmatrix} 2 + 5H \\ 2 + 2H + 3H \\ 3H \end{pmatrix}$$

When we teach kids subtraction $a - b$ and division a / b as *operations in general* then we may lock them in a way of thinking that is less effective than recognising the inverses. Subtraction is "the opposite of addition" and is indirect, and not direct, about cancellation, since e.g. also 5 - 3 = 4 - 2. The rules of subtraction include the additive inverse but do not emphasize it like H with its very definition. Just to be sure: using H helps to *focus* on cancellation, but we should avoid that kids think that it *forces* to cancellation. The teaching goal would be algebraic competence, so that kids know that 5 - 3 = 4 - 2 and 5 + 3 H = 4 + 2 H and other varieties. It depends upon circumstance and what the teacher instructs what the purpose of an exercise would be. "Simplify by means of the inverse" might differ from "Simpify by subtraction".

The didactic problem might be a different one. Apparently subtraction is taught in elementary school as the *absolute difference*, which indeed *forces* to cancellation. Subtraction in Grades 1-5 has $a - b \geq 0$ so that the pupil starts to expect that the first number will always be the highest. This approach in teaching locks into absolute difference, while subtraction is wider and allows 3 - 5 = -2. This requires later *unlearning*: what elementary school gladly hands over to secondary education. Thus, (i) at elementary school teaching actually is on absolute values, and it writes $a - b$ while it should write $|a - b|$, then (ii) junior highschool kids must unlearn this abuse of $a - b$ and learn the proper use of $a - b$, and then (iii) senior highschool must again master the concept of absolute value, with $|a - b|$ finally in proper attire. It is quite a round-about, but one can understand the *current* situation at elementary school that alway writing $|a - b|$ is superfluous when in their sums $a \geq b$ anyway. If this indeed is the diagnosis of the real problem, then its origin may still lie in the dual use of the minus sign for both the unitary and binary operations. In any case H might be an intermediate stepping stone to cure this teaching up the wrong tree.

- Thus, teaching in elementary school already focuses on cancellation by treating subtraction as *absolute difference,* and the latter would be counterproductive but only for later education and not in elementary school itself (except for a bit in Grade 6).
- The decisive criterion appears to be: what is best w.r.t. the whole curriculum. Minimise *unlearning.* The inverse element x^H with the crucial definition $x\, x^H = 1$, for values other than 0, indicates that the crux lies with the unitary operation. Others would feel that the binary $1 / x$ would be more intuitive, and that the presence of 1 would be essential. It appears however that the fraction bar has needless complexity, while kids must learn about exponents anyway. **The**

argument on the best theory should be clear.

- Thus the approach below is to introduce the negative numbers by means of *H,* and then define the double use of -*b* and *a* - *b* in terms of *H,* so that *a* - (-*b*) can no longer be an issue. The same can be achieved by using -1 instead of *H,* but we are considering didactics now, also involving some algebra, and what would work best (i) in the minds of the kids, and (ii) in their handwriting. Only empirics allows kids to tell us what works for them.
- PM. This approach might be self-defeating. We suggested that x^H is better than x^{-1} since the latter might suggest to kids that they must subtract something. By developing didactics for the negative numbers such that *H* plays a key role there too, we might actually cause that kids will have the same idea for x^H. However, the idea is that *H* will be an essentially algebraic notion throughout the curriculum, basically in multiplicative form, so that the substitution with -1 only applies on occasion. The other idea is that kids would master -*b* and *a* - *b* and *a* - (-*b*), so that eventually *H* would remain relevant only for x^H.

Again: Given all this uncertainty I do not want to take much risk and prefer the use of *H* at least as an analytical device to indicate the stages in education on our topic.

2.5. *A child wants nice and no mean numbers* (CWNN*)

This paper started with the idea to write some routines to support the didactics of x^H. While doing so, I got amazed by the treatment of the negative numbers in general. Then I intended to only briefly mention Colignatus (2015b), *A child wants nice and no mean numbers* (CWNN), since this extends again with kindergarten and Grade 1, but I now must refer to CWNN with emphasis, including the amendment in the paper (2018a). Both are joined in the label CWNN* (with the need for a new edition).

- CWNN* namely develops the place value system with proper pronunciation of the integers, also clarifying the role of grouping (multiplication). When the integers are pronounced correctly, say 21 = two·ten & one, then it is also clear that 2 × 10 + 1 = 21. It helps understanding 2 × 21 = 42 too. Colignatus (2018a) provides a *Mathematica* notebook with package that demonstrates this with sounds, for English, German, French, Dutch and Danish.

```
PlaceValue[21, Speak → False]
```

two·ten & ·one

- The US Common Core State Standards (CCSS) (2018) build up the place value system in kindergarten and Grade 1 and 2, while grouping (multiplication) is only introduced in Grade 3. This is inconsistent and didactically inverted. CCSS neglects that the very pronunciation of numbers in the full place value manner (CWNN*) already employs grouping. It is better to build up the place value system together with the notion of grouping and the table of ten and the powers of ten, i.e. to the degree that is required for kids at that moment. For example, after 9, the digits are exhaused, you introduce the new place value *ten*, also *explain that this is your new place value*, and proceed with *ten & one* etcetera. Building up the place value system must be done in a *measured* manner of course. We cannot assume that kids at that stage understand about *formal* multiplication. It makes much sense to first learn to *do* something before the full explanation is given, because you may only understand the explanation when you grasp what it is about. See the reference in (2018a) to Mannoury and “The meaning of a word is its use”. The point is: much more can be done than CCSS allows for.

- Colignatus (2018b) presents routines for the tables of addition and subtraction that make better use of the place value system, namely by using negative numbers. These tables now have gotten versions (included here) that use *H.*
- The negative numbers are introduced in the US Common Core only in Grade 6 and in Holland and the UK in Grade 7 (junior highschool). Currently fractions (expressions with a fraction bar) are discussed before the negative numbers. It would make more sense to first discuss the integers (whole numbers) before their inverses (rationals, quotients). This present notebook with its package will be better understood when the reader first checks what CWNN* does.
- Let us maintain consistency. As H represents a half turn, the half or 2^H would be a quotient, and if a discussion of those would be delayed, kids would not know that 2 halves = $2\ 2^H = 1$. My best suggestion at the moment is that elementary quotients like 2^H and 3^H (ancient Egypt knew only some like those) could be discussed in *words* (“per 2” and “per 3”), and indeed discussed as inverses, before being treated in formulas.

2.6. Vocabulary

2.6.1. Negative numbers

Minus is for the binary operation. We have -7 = *negative* 7 = H 7 = *ehta* 7, and not *minus* 7, though *minus sign seven* is correct as a literal statement to describe what has been written. (In Dutch we can say 7 *minus* 10 = *min* 3.)

```
PlaceValue[-123, Speak → False]
```

negative ·hundred & two·ten & three·

There is a distinction between (i) the single operation $-x$ that means $H\,x = (-1) \times x = 0 - x$ and (ii) the the use of the *digits* and thus the notation with concatenation of the minus sign with some digits (other than merely 0s). We can write "$-x$ (digits)" for the latter. While 7 H is literally "7 ehta" we can also say "negative 7" for its material meaning, whether "-7" has been defined or not. When we transform 123 into place value weights {1, 2, 3} then we also transform -123 into {-1, -2, -3}, see Colignatus (2018b).

Some handheld calculators distinguish between the long — *minus* and the short - *negative,* so that we have *a* — (-*b*). It doesn't survive in the handwriting of students, and those calculators are actually annoying (except perhaps for kids who need to learn about the distinction).

Thus, I will *employ* the same *minus sign* both for *minus* and *negative,* and will *pronounce* (*write* about) *negative* 7. The negative numbers however are both $H\,x$ ($x > 0$) and the numbers with the minus sign. "Negative numbers with the minus sign" is a correct subcategory of the negative numbers but "numbers with the minus sign" suffices.

Mathematica has the minus sign hard-coded in the integers (taking 2 *em* rather than 1 *em*). The following tests (routines with Q) employ that property. The tests are useful for the routines but also for testing answers provided by students.

```
? MinusSignIntegerQ
```

MinusSignIntegerQ[x] is IntegerQ[x] && Negative[x]

```
? IntegerLineHQ
```

IntegerLineHQ[expr] tests whether expr strictly
would be on the IntegerLineH, thus with expr one of ..., 2 H, H, 0, 1, 2, ...

```
? IntegerHQ
```

IntegerHQ[expr] is IntegerQ[expr] || IntegerLineHQ[expr], thus allowing integers with the minus sign and H, with the leniency of allowing also for –x with symbolic x (assumed to be an integer)

2.6.2. Rational numbers

For education *expressions* are key. If the answer is 5 then the teacher doesn't want to see $\sqrt{25}$ even though they are the same number. Education thus has *standard forms*. See Colignatus (2017a). There is a tendency amongst mathematicians (without a background in programming) to forget about *expressions* and to define numbers as *equivalence classes*, but explain to them that education has another objective.

The CCSS defines "fraction" by using variables, which they thus cannot show to kids:

- The Common Core definition: "Fraction. A number expressible in the form a/b where a is a whole number and b is a positive whole number. (The word fraction in these standards always refers to a non-negative number.) See also: rational number." http://www.corestandards.org/Math/Content/mathematics-glossary/glossary/
- The Common Core: "Rational number. A number expressible in the form a/b or – a/b for some fraction a/b. The rational numbers include the integers."

It is not clear to me why CCSS includes the potentiality of "expressible", i.e. that a number might be expressed with a fraction bar but need not be so (yet). In a / b we might take $b = 1$, and thus for the Common Core every nonnegative integer is a fraction, even when there is no fraction bar. I doubt that this is so useful. It is tempting to take the CCSS at face value, that what they define as "fraction" really is a fraction to them indeed, but it is also tempting not to believe them, and to think that they actually have another definition of rational number in mind and that they regard as fraction only nonnegative *expressions with a fraction bar*. The choice of $1/b$ in 3.NF.A.1 indicates this too:

```
CCSSMathContent["Grade 3", "NF", "A.1"]
```

Develop understanding of fractions as numbers.
CCSS.Math.Content.3.NF.A.1
Understand a fraction 1/b as the quantity formed by 1 part when a whole is partitioned into b equal parts; understand a fraction a/b as the quantity formed by a parts of size 1/b.

```
CCSSMathContent["Grade 3", "NF", "A.2"]
```

Develop understanding of fractions as numbers.
CCSS.Math.Content.3.NF.A.2
Understand a fraction as a number on the number line; represent fractions on a number line diagram.

We cannot know what CCSS "really" thinks, and it does not really matter much, since CCSS only gives an outline and not the details for the devil to hide in. NB. This discussion is about the didactics of the fraction bar, not about what division entails in mathematics.

My definitions:

- We distinguish expressions from their values. While $4\ 2^H = 2$ and thus the values are the same, the expressions "$4\ 2^H$" and "2" however differ. Education is much about starting with some expression and rewriting it into a *standard form*.
- Our present discussion is about the didactics on expressions. There are expressions with a fraction bar and expressions with an exponent *H.*
- As a general rule: avoid the term "fraction" since it is littered with confusions. The term "fraction bar" however would be unambiguous and refer to an expression rather than a numerical value.
- A **quotient** is an expression "$n\ d^H$" in which $d \neq 0$. For this quotient, n is called the numerator and d the denominator. The expression with a fraction bar is "n / d", which we can mention but will not study deeply because its further use causes more awkward terms.
- **Rational number**: A number expressible in the form $c + n\ d^H$ for integers or rational numbers n, d and c, where $d \neq 0$. This form makes a number a rational number, but it is not yet the standard form for a rational number. For example, $2 - 2^H$ would not be in standard form but we can already discern that it is a rational number. See below.
- The rules for the standard form might be seen as being quite involved. A quick method is to ask *Mathematica* to determine the integer and fractional parts of a rational number, and then compose the whole number from adding these. Let us use a negative number. If the fractional part is nonzero then traditionally we get a fraction bar but the proposal is to use exponent *H.* This use of *H* only involves the exponent while we can keep the minus sign for subtraction and negative integers.

The quotient part and *Mathematica's* FractionalPart are both unambiguous, and unambiguously the same, but I advise the term quotient to avoid contamination by confused notions about what a "fraction" would be. (E.g. *Mathematic's* FractionalPart differs from the CCSS fraction since the latter only has nonnegative numbers. Perhaps FractionalPart is what some might call a pure fraction, to distinguish this from the mixed number, but the mixed number would still be a "fraction" according to CCSS.)

```
-80 / 9  // ToH  // Evaluate // SortH (*standard form*)
```

$-8 - 8 \times 9^H$

```
num = -80 / 9; ip = IntegerPart[num]; fp = FractionalPart[num];
ip + fp // Inactivate (* Inactivate does not resolve the + - *)
```

$-8 + -\frac{8}{9}$

The *standard form for the rationals* using the minus sign is as follows.

1. If a rational number is equal to an integer, it is written as this integer, and otherwise:
2. The rational number is written as an integer *plus* or *minus* a quotient of natural numbers.
3. The integer part is not written when it is 0, unless the quotient part is 0 too (and then the whole is the integer 0).
4. The quotient part has a denominator that isn't 0 or 1.
5. The quotient part is not written when the numerator is 0 (and then the whole is an integer).

6. The quotient part consists of a quotient (form) with an absolute value smaller than 1.
7. The quotient part is simplified by elimination of common primes.
8. When the integer part is 0 then *plus* is not written and *minus* is transformed into the negative sign written before the quotient part.
9. When the integer part is nonzero then there is *plus* or *minus* for the quotient part in the same direction as the sign of the integer part (reasoning in the same direction).

```
? IntegerPart
```

IntegerPart[x] gives the integer part of x. >>

```
? FractionalPart
```

FractionalPart[x] gives the fractional part of x. >>

```
? MixedNumberH
```

MixedNumberH[expr] puts all Rational[n, m] or n
Power[m, −1] in expr into IntegerPart[expr] + ToH[FractionalPart[expr]]
MixedNumberH[x_Rational] is a helper routine
MixedNumberH["Example"] gives an example comparison of using H and the fraction bar

2.7. Support by *Mathematica* and the new package

2.7.1. What is required

H = -1 is simple, but introducing it *when it isn't used yet* comes with some complexity.

The body of the text deals with the issue and didactics on content, but there is also a technical aspect:

- Routines that mimic what kids would do with pencil and paper.
- Routines for teachers and research on mathematics education, both for clarification and for everything to work.

2.7.2. What is provided by *Mathematica*

Mathematica already provides most of the didactics using *H*. However, some elements appear to be missing. The new routines provide much of what is lacking. *Mathematica* has the following built-in:

```
-2 // FullForm
```

-2

```
-b // FullForm
```

Times[-1, b]

```
1 / b // FullForm
```

Power[b, -1]

We have subtraction *a* - *b* as the *binary operation*. This can also be seen as *a* + -*b,* so that we might argue that there is no need for that binary operation called “subtraction”. *Mathematica* already provides for this approach. However, it still allows us to call it subtraction.

```
? Subtract
```

$x - y$ is equivalent to $x + (-1 * y)$. >>

To summarise, we have these four built-in properties.

```
{-2, -b, a - b, 1 / b} /. -1 → H (*RuleAnyToH*)
```

$\{-2, b\,H, a + b\,H, b^H\}$

The integers with a minus sign -1, -2, -3, ... are hard-coded within *Mathematica*. Obviously this would remain so because of its application in the world of science. The inclusing within *Mathematica* of *H* as a universal constant alongside *i*, Pi and E does not require also such adaption of the integers with the minus sign. Potentially, though, for education, there might be created a setting of the Preferences, that employs *H* in above use of Times and Power.

Unfortunately, *Mathematica* uses Minus[*x*] for -*x,* while we use *minus* for the binary operation Subtract. Minus[2] is immediately hard-coded into -2. It would have been bettter to use Negative[*x*] for -*x,* and NegativeQ for what now is Negative.

```
Minus[x]
```

$-x$

```
? Negative
```

Negative[x] gives True if x is a negative number. >>

Remarkably 1 / *b* is coded as b^{-1} but displayed with a fraction bar. Even more remarkable, this also happens for input b^{-1} itself, that has the proper form in both input and FullForm but not in display.

```
{1 / b , b^-1}
```

$\left\{\frac{1}{b}, \frac{1}{b}\right\}$

In the early years, *Mathematica* printed numerator and denominator on separate lines, separated by another line with dashes. There still is a remnant in ToString, with the newline character “\n”.

```
b^-1 // ToString // FullForm
```

"1\n–\nb"

The puzzle is only complete when all pieces are at their proper places, and we must do some programming to get them there.

2.8. A quote by Vredenduin 1991

The use of *H* introduces an aspect of *abstract thinking* alongside *mere calculation,* though calculation can also be regarded as the routine application of some abstract rules. This abstraction is not necessarily at the highest *Van Hiele level* though. Thus, we distinguish between (i) (natural) basic abstraction and (ii) the more involved abstract modeling at the highest Van Hiele level.

Piet Vredenduin (1909-1996), a Dutch teacher of mathematics and author in mathematics education, helps us to see the distinction. In his booklet on positive and negative numbers, that also relates how confused our ancestors have been on the negative numbers, he closes with this statement (1991, p119, my translation):

"from reality, a particular structure is distilled by abstraction. This structure is investigated apart from reality by means of deduction. Because of its origin the derived properties of this structure can be applied to reality again."

This is the obvious answer to Eugene Wigner's suggestion of *magic*, in his statement on the "*unreasonable effectiveness of mathematics in the natural sciences*". I actually gave this answer myself before I read Vredenduin - see also Colignatus (2015e) - but it is quite nice to quote Vredenduin precisely from this booklet on positive and negative numbers.

The quote clarifies the distinction between *basic* and *developed* abstraction. My suggestion is that abstraction forms part of the notion of *thinking* itself. Since pupils are human, their minds are not strangers to thought and abstraction. As teachers we can encourage students to work with abstraction in a more controlled manner. Abstraction is also a very simple notion, for it means: leaving out things. Abstraction however appears hard to do, when one depends upon context (e.g. for lack of space in working memory) and when it is obviously possible to leave out what you should actually focus on.

We can deal with expressions with *H* via *algebra* or *tables of addition and subtraction*. Before we look at such tables, we better first master some algebra (for the tables rely upon this too, actually).

There is the approach that kids better learn solid methods first (by drilling) before they would be capable of understanding what they are doing. I already indicated that this has a core of truth. However, my approach is that mathematics should be clear by itself. Mathematics by itself will support various teaching and learning strategies. Tradition appears *not* to be clear *at all*. The traditional approach to negative numbers and rational numbers is cumbersome (which invites drilling) and thus is better be replaced by a clear approach, whatever the teaching and learning method. (If you must drill: do it on something that is clear.)

2.9. Considerations on the multiplicative inverse

The use of exponent *H* causes a shift of attention from the *rules on the fraction bar* to the *rules on exponents.* Kids might not be ready for exponents, but this must be established by empirical research. We should not reason from the current situation with the fraction bar for its sake only.

Kids must learn the rules for exponents at some time anyway, the sooner the better, and there is really little use of learning about "division" in two separate ways - and spending most time on the clumsy fraction bar merely because it had more momentum in history.

Many, if not most, people who learn that x^{-1} is the same as $1 / x$, wonder why there are two notations for the same thing. Van Hiele set the step to wonder what would be didactically best. The proposal to use the exponent for didactics thus is not new. The proposal to use the universal symbol *H* that can be interchanged with the value -1 is not so new either: Peter Harremoës (2000) discussed a much more involved idea on the negative numbers though with different notation. My innovation is to see the relevance for didactics: to combine these notions, choose the single element -1, select H as the symbol that may remain unevaluated, and clarify the meaning for didactics in texts like this notebook.

There may be people who regard $x^{-1} = 1 / x$ as a great discovery, i.e. the wonder that division is related to exponent -1. I am sorry to say that these people then lack in acumen, and this might even hold for mathematicians. The meaning of the identity sign is that $1 / x$ is superfluous, just like $H x = 0 - x$ means that one of these representations is superfluous. The meaning of our present discussion is that $1 / x$ is counterproductive in didactics. See below for a comment on Group Theory.

The following gives an overview of the relevant algebraic rules for *H,* and they are simpler than the rules for the fraction bar. (If the idea is that the fraction bar employs more space of writing, and the exponent doesn't, by current conventions, and if this space of writing is important, then employ more space of writing for the exponent too.)

```
?? RuleAlgebraH
```

RuleAlgebraH gives algebraic rules that tend to keep H. Used in AlgebraH[expr]

```
Attributes[RuleAlgebraH] = {Protected}
```

$$\text{RuleAlgebraH} = \{1 + H \to 0,\ \text{HoldPattern}[H\,H] \to 1,\ H^2 \to 1,\ H^H \to H,\ 0^H \to \text{Indeterminate},\ \text{HoldPattern}[x\_\ x\_^H] \rightarrow 1,\ x\_^{1+H} \rightarrow 1,\ (x\_^H)^H \rightarrow x,\ (x\_\ y\_\_)^H \rightarrow y^H x^H\}$$

2.10. Two possible strategies for R&D and implementation in education

Writing this notebook and programming its package was quite a delicate issue. The next step of suggesting an application of H in mathematics education and its research is so too. We no longer have the small classroom and full freedom for the teacher, but we have systems with Common Core State Standards, universities and colleges packed with researchers and trainers of teachers, and, not to forget, textbook publishers.

My research and this paper started out with x^H. This could be used in Grade 3-6 without telling kids that $H = -1$. It suffices to use algebra with rules like $x\, x^H = 1$ for $x \neq 0$.

The next obvious idea was that H might also be used for the expression a - (-b) that kids find hard. It is no longer difficult with $a + H\, H\, b$.

This caused me to do a more fundamental evaluation of the curriculum on negative numbers (i.e. with its consequences for division, rationals, fractions (i.e. expressions with the fraction bar) and mixed numbers).

This discussion no longer belongs here and starts in **Section 3**.

The paper has become quite involved, and a solution was to put details per stage and the routines into **Appendices**. The reader may still recognise the structure of (i) a core argument on x^H (stage 7) and (ii) a supplementary - though more fundamental and apparently for school more dramatic - argument on the negative numbers.

Thus, this paper distinguishes two strategies for R&D and implementation:

1. Keep H unevaluated and use it only as a symbol for the multiplicative (scalar) inverse x^H with rules for algebraic manipulation, like the fraction bar $1 / x$ also has rules for manipulation. This would replace the treatment of “fractions” in Grade 3-5 by a more efficient curriculum, since kids have to learn about exponentiation anyway. This option would leave the current treatment of the negative integers unaffected.
2. *First discuss the integers (whole numbers) before doing fractions.* In the US Common Core, negative numbers are only introduced in Grade 6. Thus, interchange the treatment of negative numbers and rationals. This helps kids to get used to the notion of an inverse as well. Start with the additive inverse and the negative integers in Grade 2 and complete them in Grade 3, so that Grade 4-6 can look at the multiplicative inverse x^H and benefit from better command of algebra.

3. Didactics on the negative numbers

3.1. The importance of empirical research

It may well be that the use of H for addition and subtraction can remain restricted to being a small tool for the local illumination of $a - (-b)$ as $a + H\,H\,b$ with the rule that $H\,H \to 1$. This can only be shown by field experiments. It are the kids who tell us what works for them.

Alternatively, the use of H might also appear to be a useful tool to bridge gaps of understanding by introducing $H\,x$ before introducing $-x$ (digits). Yet, H might also delay such understanding. My idea is to eliminate *unlearning* but perhaps the introduction of H would require such *unlearning* when kids arrive at the stage to learn about the integers with the minus sign.

A serious question is whether negative numbers using H could be introduced in Grade 1 already: and I also pose this question to indicate my own lack of knowledge for this area. Currently I propose to start with H in Grade 2 because of my preference to assume that kids can read and write and have some command of the place value system. Colignatus (2018b) presents tables of addition and subtraction that have better use of the place value system: and those also use negative numbers (though we may tell kids that they use subtraction). It would make sense to be able to use tables with H in Grade 1 too.

Indeed, the introduction of the negative numbers is best evaluated while also other confusions in mathematics education are resolved. It is not urgent to know what component would bring the greatest improvement as long as the whole package appears to be better.

Empirical testing might generate results that are only locally relevant, in some time and place and culture. Perhaps empirical testing only generates “general results” that only confirm the internal logic. Or, perhaps this application of H only remains a theoretical exercise. For now, it helps to identify the 7 stages below, and the relevant questions for field testing.

3.2. Transition from counting to addition

We looked at contexts in real life but the true problem might be quite different.

The following table will be used by kindergarten teachers and not the pupils, but the pupils would see it at least in Grade 1. The table highlights the issues - I thank pedagogue and teacher drs. A.I. Roessingh for this insight. Key didactics is to properly *internalise* the *Table of Basic Addition*, since this is the transition from counting to addition. Pupils who get stuck at counting - still using their fingers - will not have properly *internalised* this table. (Its “version in sounds” is a key part of memorisation.) NB 1. The table is oriented like the system of co-ordinates. NB 2. Taking an element within the table also shows subtractions, e.g. 7 - 5 = 2 and 7 - 2 = 5.

```
TableOfBasicAddition[]
```

10	10										
9	9	10									
8	8	9	10								
7	7	8	9	10							
6	6	7	8	9	10						
5	5	6	7	8	9	10					
4	4	5	6	7	8	9	10				
3	3	4	5	6	7	8	9	10			
2	2	3	4	5	6	7	8	9	10		
1	1	2	3	4	5	6	7	8	9	10	
0	0	1	2	3	4	5	6	7	8	9	10
+	0	1	2	3	4	5	6	7	8	9	10

Key insight: *This same requirement on internalisation likely holds w.r.t. the basic negative numbers.*

Since I want to avoid the confusion between operation with *minus* and the operation that creates the negative numbers, I now assume that the number line has been extended for kids with *H*.

```
IntegerLineH[5]
```

$\{5H, 4H, 3H, 2H, H, 0, 1, 2, 3, 4, 5\}$

We might use the right hand fingers for the positive numbers 1-5, the left hand fingers for the negative numbers (using coloured bands to indicate the special use, or paint finger nails), and a double fist for zero. We now have addition *a* + *H b,* which is still addition. Say “2 plus negative 2 gives 0” (and use *plus*). I would be in favour of short “neg” and “pos” but this is not at issue here. Kids would have to memorise such a Table Of Basic Addition to work with negative numbers, in the same way as they have to memorise the table for 0-10 in order to make the transition from *counting* to *addition.*

```
TableOfBasicAdditionH[-5, 5]
```

5	0	1	2	3	4	5					
4	H	0	1	2	3	4	5				
3	$2H$	H	0	1	2	3	4	5			
2	$3H$	$2H$	H	0	1	2	3	4	5		
1	$4H$	$3H$	$2H$	H	0	1	2	3	4	5	
0	$5H$	$4H$	$3H$	$2H$	H	0	1	2	3	4	5
H		$5H$	$4H$	$3H$	$2H$	H	0	1	2	3	4
$2H$			$5H$	$4H$	$3H$	$2H$	H	0	1	2	3
$3H$				$5H$	$4H$	$3H$	$2H$	H	0	1	2
$4H$					$5H$	$4H$	$3H$	$2H$	H	0	1
$5H$						$5H$	$4H$	$3H$	$2H$	H	0
+	$5H$	$4H$	$3H$	$2H$	H	0	1	2	3	4	5

3.3. Place value system: implicit vectors and negative numbers

Kids who haven't learned arithmetic cannot later learn algebra (well). In another perspective, arithmetic is a form of algebra. The rule $x + H\,x \to 0$ is algebraic but might also be seen as calculation when applied to numbers. In the end it might be merely an issue of perspective. The tables of addition and subtraction that use the place value system use both algebra and calculation.

The place value system has these features:

- When kids can count then they also know what they are counting: fingers, cars, apples. Thus we also can discuss with them what they use as the unit of account. For the place value system, the trick is to use multiple units of account. We have blue baskets that contain only 1 apple and red baskets that contain 10 apples. When we have 10 blue baskets with 10 apples, then we replace this with one red basket of 10 apples, and count 1 of 10.
- In currect education vectors do not seem to exist, but they are used. For us, there is a difference between 10 and {1, 0}, but teachers employ the number 10 in place value manner with ample space for the positions, which we can best model as {1, 0}. Thus, below we will use the notation {*x, y, z*} for such use, though TableForm will print without brackets and commas again. The values *x, y, z* are the *weights* of the place values, and can also be called the "count for 100, count for 10, count for 1".

```
ToSingleDigits[countfor10, countfor1] (*highest place value first*)
```

countfor1 + 10 countfor10

In terms of vectors we have {1, 1} + {0, 9} and then the rule for overflow.

```
{1, 1} + {0, 9} → ToSingleDigits[1, 1] + ToSingleDigits[0, 9]
```

$\{1, 10\} \to 20$

- When we do not want to change a sum then we add 0, but we also can use "1 of 10 - 10 of 1 = 0" on the place values. The negative numbers have a natural use in the tables of addition and subtraction that use the place value system in a better manner. We now use a minus sign that might be explained as subtraction, but below we will use *H.*

```
{1, -10} → ToSingleDigits[1, -10]
```

$\{1, -10\} \to 0$

Level transition from 10 counts for 1 into 1 count for 10.

```
ToSingleDigitsTable[10]
```

	10^1	10^0
1	0	10
2	1	−10
3	+	+
4	1	0

The tables of addition and subtraction consist of two parts.

1. The first part (first line of + + + or - - -) is the addition or subtraction per position.
2. The second part (second line of + + +) calls ToSingleDigitsTable that reallocates these results according to the rules of the place value system.

This addition table first creates line 4 as output, and the second step takes line 4 as input.

```
AdditionTable[11, 9]
```

	10^1	10^0
1	1	1
2	0	9
3	+	+
4	1	10
5	1	−10
6	+	+
7	2	0

Good didactics is to start with the (2), so that the tables of addition and subtraction that start with (1) can use what has already been trained on.

```
? ToSingleDigits
```

ToSingleDigits[vec] multiplies the elements with 10^pos, counting
pos from 0 from right to left. Vec can be a list or a sequence of elements

```
? ToSingleDigitsTable
```

ToSingleDigitsTable[vec] decomposes the place
value vector. Vec must have integers, and may be a list or a sequence.
ToSingleDigitsTable["Tested", vec, table] complements the table, and then vec and table must
be lists. The test is on the proper length of vec for the table (overflow digits). The routine is
called by AdditionTable, SubtractionTable and MixedAdditionTable for completion, and its
option setting controls their output. Default option is Differences –> True; otherwise levels

3.4. Considering Grade 2

I look at Grade 2 because I want to assume that kids have learned basic reading and writing, and have some grasp of the place value system, see Colignatus (2018ab).

3.4.1. Van Hiele levels. Materials → basic abstraction → developed abstraction

I agree with the notion in didactics that kids would work with materials first and models later. This relates to the *Van Hiele levels of insight*. Taking 3 apples from a basket of 5, so that 2 remain, is a commonplace material method in Grade 1 to teach $a - b \geq 0$. The reference to the use of materials should not obscure that the kids actually perform basic abstraction, for this is what *thinking* entails. It are not robots who take 3 apples from the basket. Subsequently I would protest when a negative number as in 5 + (-3) = 2 might be seen as a “model” at the highest Van Hiele level. Traditional arguments would be, roughly put: “negative numbers do not exist”, and “you cannot have -3 apples”. The latter puts the negative numbers at the highest Van Hiele level of abstraction, so that they would be inaccessible to kids who are supposed to be working at lower levels. This however would be an invalid application of Van Hiele’s theory. There is also abstraction at a lower level than such involved modeling (check Vredenduin). Counting and addition are activities rather than objects, and we still teach them. Negative numbers are a modification of such activity, namely reversing and cancellation, and would be at the same level of comprehension.

There are also contexts from real life: (i) kids can walk in a direction, turn around and walk back along the earlier path, which turn is given by $H = -1$, (ii) there is the system of 2D co-ordinates, and kids might be able to read a small map, (iii) kids know about mirrors, (iv) see the argument on -3 apples and the hungry cookie-monster above.

Kids in Holland must understand what -1 degree Celsius is. It is not clear to me how teachers in Holland deal with this, as the negative numbers appear only in junior highschool (Grade 7). The US Common Core is quite prim with the introduction of the negative numbers, namely after the fraction bar and only in Grade 6. The US might avoid the temperature problem by using Fahrenheit. Bellamy (2015) reports on the UK and properly warns that these temperature measures have an interval scale and that only Kelvin has a ratio scale.

Perhaps as a last resort, parents could open bank accounts for their kids's pocket money, and allow figures in the red. Perhaps the world financial system would support the introduction of negative numbers at an early stage in education.

3.4.2. Negative numbers exist in tables of addition and subtraction

We actually have this situation:

- The current curriculum has few examples of the use of negative numbers, and thus it might appear that they are not required, but actually there has been ***deliberate effort*** to exclude them, glossing them over where they better be mentioned. Teachers within the system might not be aware of such deliberation as this occurred somewhere in past tradition.
- The negative numbers clearly "exist", namely in addition in the place value system. Conventionally these negative numbers are not shown, but the suggestion is to *show them.* See Colignatus (2018b). For example:
  $15 + 8 \to \{1, 5\} + \{0, 8\} = \{1+0, 5+8\} = \{1, 13\} = \{1, 13\} + \{1, -10\} = \{1+1, 13-10\} = \{2, 3\} \to 23$,
  using $\{1, -10\} \to 0$.
  The RHS table gives the levels only, the LHS table gives the differences too .

```
BothMixedAdditionTablesH[15, 8]
```

	10^1	10^0
1	1	5
2	0	8
3	+	+
4	1	13
5	1	$10\,H$
6	+	+
7	2	3

	10^1	10^0
1	1	5
2	0	8
3	+	+
4	1	13
5	2	3

In Holland, the RHS table is commonplace. This means that the calculation on 13 + (-10) is glossed over. It is only said that “the overflow is passed on to the next place”. To a large extent it is fair that kids can learn to ride a bike without fully understanding Newton’s laws, yet a good case can be made that an early introduction of the negative integers would both be understood and work wonders (such as understanding of the place value system). The LHS table uses the algebraic rule that $x + x\,H = 0$ (so that we do not change the sum) and assumes that kids can can play the cookie & hungry cookie-monster game.

Importantly, we now have a unified method that also allows for subtraction, seen as addition with *H*. The following would be a table of addition using *H,* see Colignatus (2018b) for the form without *H.*

```
BothMixedAdditionTablesH[14 , 6 H]
```

	10^1	10^0
1	1	4
2	0	$6H$
3	+	+
4	1	$2H$
5	H	10
6	+	+
7	0	8

	10^1	10^0
1	1	4
2	0	$6H$
3	+	+
4	1	$2H$
5	0	8

This example crosses the zero boundary.

```
BothMixedAdditionTablesH[17 , 25 H]
```

	10^1	10^0
1	1	7
2	$2H$	$5H$
3	+	+
4	H	2
5	1	$10H$
6	+	+
7	0	$8H$

	10^1	10^0
1	1	7
2	$2H$	$5H$
3	+	+
4	H	2
5	0	$8H$

3.4.3. Types of multiplication

It all depends how difficult kids find the notation and relation $H\,7 = (-1) \times 7 = -7$, or the relation of a number to a *scalar multiplication* of itself. When elementary school has explained multiplication as *grouping* only, then students may get stuck. This doesn't mean that the topic by itself would be complicated, only that elementary school has turned a blind eye to this distinction in types of multiplication, and to the negative numbers in general. Above notation of course assumes (-1) as a basic concept without yet an explanation of $(-1) \times 1 = -1$.

3.4.4. Not all abstraction is at the highest Van Hiele level

Thus a key point is: there is quite a difference between abstraction as in Euclidean axiomatics (earlier taught in junior highschool, which inspired Van Hiele) and the abstraction that we are discussing here.

There is also this joke: A biologist, an economist and a mathematician watch an empty house. A man and a woman go into the house, and some time later they leave the house again, taking along a third person. The biologist clarifies: “They have procreated.” The economist clarifies: “The assumption that the house was empty was false. There was at least one person already there.” The mathematician says: “Well, the assumption that the house was empty has been made, and there is no need for new assumptions on biology, for there is now -1 person in the house.”

In other words, it is a matter of abstraction indeed, but not immediately at the highest level, and the issue is whether kids can be guided from basic abstraction to a proper handling of the negative numbers, by means of proper didactics and examples from daily life and history.

3.5. Incoherence in Math Ed research

There appears to be a remarkable incoherence in mathematics education and its research w.r.t. the negative numbers (too), which reminds of the problems that the world itself had since the discovery of *direction* by Albert Girard in 1629 and the introduction of the *number line* by John Wallis in 1673 - see the discussion by Vredenduin (1991).

The critical reader will look at my list of references (though see also CWNN*), to check whether I have read sufficiently wide to warrant this diagnosis of "remarkable incoherence". There are no such wide references, and the reason is that I looked at some and was quite disappointed - and there is no need to list those and explain my reaction. Let me give three examples though.

Skemp (1973) is recommendable but his treatment of the negative numbers (p201) is cursory and off-topic. He gives examples of operations that cancel each other, and then states: "In the mathematical realm, adding two and subtracting two are opposite operations which cancel. So let us represent these operations by (+2) and (-2), using parentheses here to show that the + or - sign is fused with the 2 to represent a new kind of number." Skemp's "So" is overworked. Apparently (-2) now is an operation itself, and (-2)[x] = x - 2. Admittedly it is feasible and we can enhance this by equivalent classes so that (-1-1) is equivalent to (-2), but I am afraid that we are far adrift from "number sense" now, and this doesn't help education of the negative numbers.

David Tall is famous - together with Eddie Gray - for the notion of a "procept". The various uses of the minus sign make it a procept indeed. While Tall seems to support the deliberate use of procepts, I would rather advise to deconstruct them into components, to support the better understanding of the separate uses. David Tall (2013:101) mentions under "squaring a negative number" that some students cannot distinguish -3^2 and $(-3)^2$, but this may well be a problem on using brackets and not be a problem of understanding negative numbers by itself. Overall, Tall in this book is doesn't help on our topic- but his book has another intention. One should know though that his book misrepresents the contribution by Pierre van Hiele, see elsewhere.

Anna Bellamy (2015:4) states: "I could not find a standard textbook that would present a convincing argument to Year 7 students why it is that "two minuses give a plus"." This should be "why two negatives give a positive", but she herself makes this distinction and is only quoting the confused student here. Her finding is: "the main challenge in introducing negative numbers lies not in broadening the number system, but in (re)defining the four basic operations: addition, subtraction, multiplication and division." (p1). Perhaps, but when most kids have a confused grasp of such operations anyway, and kids are confused because their teachers are confused, then you cannot really draw that conclusion, and then you are not merely dealing with the introduction of negative numbers but dealing with the redesign of mathematics education itself, see https://zenodo.org/communities/re-engineering-math-ed/about/.

I stopped reading these texts. My strategy now is to first write up what my diagnosis is, and proceed from there.

3.6. Comment on variables

Our discussion uses variables. This essentially serves teachers. Pupils will work with numbers, and our variables are their schemes (learning from examples). However, it is very likely that pupils already can work with variables. They already know that "John" is a name, that can be used to identify a particular person, but there are more people of that name. Or they know that "a number" stands for a number without specifying which. Thus x is a name that can be used to identify a particular number (and not indicated which: actually *any* number). The use of variables in elementary education however is another topic, and not material to this present notebook.

Identify where the situations are incomparable so that kids would be lost:

- Harry is a cop running after Rick who has stolen an apple. George runs along and tries to help catch Rick. George is a world champion in running, for both short and long distances. Who runs fastest is called the winner. Who is the likely winner: George, Harry or Rick ?
- There are three numbers: 4, 6 and 7, and x is the highest even number. What is x: 4, 6 or 7 ?

We might not use x but a more fancy expression, and allow kids to use their own variable names, with x as one of the options.

This would seem to fit the US Common Core for Grade 2.

```
CCSSMathContent["Grade 2", "OA", "A.1"]
```

Represent and solve problems involving addition and subtraction. CCSS.Math.Content.2.OA.A.1
Use addition and subtraction within 100 to solve one– and two–step word problems involving situations of adding to, taking from, putting together, taking apart, and comparing, with unknowns in all positions, e.g., by using drawings and equations with a symbol for the unknown number to represent the problem

However, looking at the CCSS "Glossary Table 1": they use the question mark (2 + 3 = ?) instead of "x = 2 + 3, what is x ?". I would prefer that the question mark is not abused as a variable. (CCSS is somewhat crooked: using a symbol but it cannot be a proper variable, because then some theorists of mathematics education might hold that kids cannot use variables yet.)

By another fluke of history, the multiplication sign × looks too much like x, especially when discipline to use italics for x is lacking. We better keep the international convention on the x-axis and y-axis. Later multiplication either disappears as in $2\,x$ or is replaced by $2 * x$. My suggestion is that elementary school follows the latter convention. If you really must, start with 2 × 3, but once x appears on the scene then use 2 * 3. This is also supported in *Mathematica.*

```
? Times
```

$x*y*z$, $x\times y\times z$, or $x\,y\,z$ represents a product of terms. ≫

3.7. Comment on group theory

Van Hiele continued his proposal on the inverse element x^{-1} by proposing *group theory for school*, arguing that pupils can do more in abstract thinking than often presumed. The use of H might be an amendment to group theory itself (which amendment of H is not discussed by Van Hiele). The Abelian group requires inverse *elements,* and is tricky about the *operation* that creates the elements. Mathematicians are lawyers of space and number, and they can create mental constructions that seem right but that are not necessarily the most relevant ones. For the operation of addition, the inverse element of x currently is $-x$ but would become $H\,x$. For the operation of multiplication, the inverse element of x currently is $1 / x$ or x^{-1} but would become x^H. Strikingly, the inverse element basically uses an operation, and engages what we currently perceive as *properties of a higher* *level*: for addition we use (scalar) multiplication in $H\,x$, and for multiplication we use exponentiation. Current group theory covers this up by looking at the *elements* only, and by suggesting that $-x$ would be at the same level as addition, and that $1 / x$ is at the same level as multiplication. This focus on *elements* and not *operation* serves only to have the Abelian group look nice and neat, but without attention for what is really happening. Thus: (a) It would be clearer to presume the existence, not of “inverse elements”, but the existence of an “operator to generate inverse elements w.r.t. another operator”, which operator already has features of the next level of modeling. (b) An argument that current education is structured along the Abelian groups because addition and subtraction would be “at the same level”, and multiplication and division would be “at the same level”, abuses the term “same level” (defining what needs to be proven) and thus would be rather blind to what is really happening. (The Abelian group is a theory and not a discovery for ideal didactics.)

(A portal and no source: https://en.wikipedia.org/wiki/Abelian_group)

PM. For multiplication, we distinguish between scalar multiples and grouping. We do not write 2 km as 2 × km. While H is a scalar, we still tend to call $H\,x$ the additive inverse, with $x + H\,x = 0$. Thus we would call x^H the scalar inverse instead of the grouping inverse. However, the term “scalar inverse” tends to confuse with $H\,x$. Thus I settled for “multiplicative inverse” though it is less precise.

4. Stages in didactics and properties of *Mathematica*

4.1. Structure

We start with the stages in didactics on our topic. We identify properties of *Mathematica* that relate to those stages, in support or in conflict with the intended use (**Appendix A**). We develop routines that use and respect those properties (**Appendix B**). Subsequently, we look at the properties of the routines, and relate them to the stages in didactics (**Appendix C**). Finally, we discuss the stages, in the main body of the text again, using the routines and what *Mathematica* already has provided. We manage these different aspects by (i) discussing the stages in the main body of the text, (ii) moving the issues on programming to Appendices, and (iii) putting details on the stages in separate discussions too.

4.2. Stages

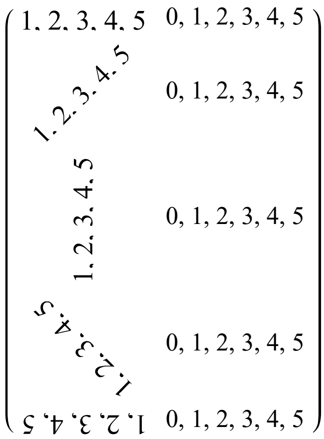
Working with the minus sign has four aspects: (i) multiplication with a scalar, rotation in the plane, or mirroring along the number line, (ii) the operation on two elements (“extending” plus), (iii) the exponent, (iv) notation: concatenating the minus sign with digits (theory of expressions). Thus there is a mixture of content and notation. The use of H helps to better distinguish these aspects.

We can recognise the following 7 stages in the curriculum, if we were to introduce the negative numbers. These stages are introduced here, and details are in the separate discussions. There, we relate the Stages to the Grades relevant for the US Common Core State Standards (CCSS) (2018).

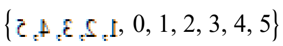
1. Kindergarten. Speaking and listening, using practical examples. Though there is no reading and writing, the *vocabulary* should be unambiguous (at university level of precision). For example, *large* and *small* concern absolute sizes, while *more* and *less,* and *high* and *low,* concern relatives and thus may imply negative numbers (currently glossed over with deliberate effort). Temperatures are “below zero” and not “smaller than zero”.
2. Reading and writing. Addition and subtraction with nonnegative integers and differences, i.e. with $a - b \geq 0$. Perhaps we should use $a + H\, b$ from the start, and only later introduce $a - b$, but this better be tested and it seems wiser to currently start from the usual base.
3. Introduction of the negative integers (i.e. H) and the number line with H by rotating or mirroring the positive numbers. Thus not the numbers with the minus sign yet. This is only a definition, and can be given in 15 minutes, but it requires attention and precision.

The current routine rotates at the center of the string on the left, for simplicity of programming, but it would only be correct to rotate at zero. The rotation routine input doesn’t use *angles* but *turns* in [0, 1].

```
(TextRotate[#][5] & /@ ({0, 1, 2, 3, 4} / 8 (*turns*)) ) // MatrixForm
```



```
TextReflect[5]
```



```
IntegerLineH[5]
```

{5 H, 4 H, 3 H, 2 H, H, 0, 1, 2, 3, 4, 5}

- $H\, x$ is clearly both a scalar multiplication of x and a position resulting from this.
- There is a clear relation between this number line and the rule $x + H\, x \to 0$.
- There is no conflict with the already existing use of $a - b \geq 0$.

4. Addition with H. This is vector addition along the number line. With 5 an arrow to the right and 3 H an arrow to the left, the result is an arrow 2 to the right, which is $5 - 3 = 2$. Application to $a\, H$ itself gives $a\, H + a\, H\, H = 0$ and $H\, H = 1$. Pupils already know $a - b$ for nonnegative outcomes, and we can identify $a - b = a + H\, b$, so that addition with the new numbers means a subtraction in the familiar numbers.
5. Extension with $a + H\, b = H\, c < 0$. This arises naturally, e.g. $5 + 7\, H = 5 + 5\, H + 2\, H = 2\, H$. Subtraction $a - H\, x$ gives $a + H\, H\, x$. A key rule is $H\, H = 1$.

6. Writing -*c* as alternative to *H c*. Once kids are familiar with working with the extended number line then it is only a question when it would be the right moment to switch to the integers with the minus sign (IwMS). This is an international standard that they must master too. Kids translate expressions with *H* into expressions with the minus sign. Working with the minus sign, they might want to transform back to *H* to make sure that they understand it properly.

```
IntegerLineH[5] // FromH
```

{−5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5}

- This number line gives only positions and there is no clear scalar multiplication in -*a*. Only by deliberate discussion it is seen that (-1) × *a* = -*a* and also 0 - *a*. The interpretation by rotation or mirroring is not clear either (while *H* has been introduced as such).
- There is an implied relation between this number line and the rule *a* + (-*a*) → 0, but it can be doubted whether the line is conventionally introduced this very manner. (“Here on the left, we want a number *x* so that 5 + *x* = 0. Let us write this *x* as -5 so that we have 5 + -5 = 0.”) Brackets would tend to be needed here, since *a* + -*a* might read as *a* +- *a*, but perhaps the latter is intended.
- If introduced “just like that”, there is a direct conflict with the use of the binary *minus*. There is no time to build this up, and it requires the immediate definition *a* + (-*a*) = *a* - *a* = 0.

It is not clear to me whether *H* would be a useful intermediate step indeed. Perhaps it is possible to first say that *a* - *b* may also be written as *a* + (-1) × *b* and also as *a* + (-*b*), and only then construct the number line, directly using the minus sign. Perhaps it suffices to train kids better on the use of brackets. Empirical research however hasn’t had this focus since the negative numbers only appear so late in education. Given all this uncertainty I do not want to take much risk and prefer the use of *H* at least as an analytical device to indicate the stages in education. (Lines copied to above.)

7. Exponent *H*. Once kids have mastered subtraction, *H* remains useful forever for the exponent. We get: multiplicative inverse element, rational number, mixed number.

```
? IntegerLineH
```

IntegerLineH[n, m] gives a list of integers, from n to the left of
0, to m to right of 0. Input of negative numbers like −n and −m is allowed
IntegerLineH[n] is IntegerLineH[n, n]

```
? FromH
```

FromH[expr] replaces H by −1

4.3. Common Core on fraction bar and negative numbers

Grade 2 has pupils working with counts of hundred, but it would be better to have -10 first, since the objective is to get to understand the place value system and the properties of addition and subtraction, with -10 appearing in the tables of addition and subtraction.

Common Core is early with the fraction bar in Grade 3, 4 and 5, and late with (some) negative numbers only in Grade 6, see http://www.corestandards.org/Math/Content/NF/. In Holland kids have to wait till Grade 7 (junior high) before they see -1.

My suggestion is to turn this around. The negative integers can be handled in Grade 2 & 3, and the numbers with exponent *H* (no fraction bar) can be dealt with in Grade 4, 5, and 6.

- Stage 3 would apply to Grade 2.
- Stage 4 would apply to Grade 2 & 3.
- Stages 4, 5 and 6 would apply to Grade 3.
- Stage 7 would apply to Grades 4, 5, 6.

See also the *Vocabulary* in the main body of this paper.

PM. The Common Core states on the current order (my emphasis):

Common Core Grade 2: "Students extend their understanding of the base-ten system. This includes ideas of counting in fives, tens, and multiples of **hundreds,** tens, and ones, as well as number relationships involving these units, including comparing. Students understand multi-digit numbers (**up to 1000**) written in base-ten notation, recognizing that the digits in each place represent **amounts of thousands, hundreds,** tens, or ones (e.g., 853 is 8 hundreds + 5 tens + 3 ones)."

```
CCSSMathContent["Grade 3", "NF", "A.1"]
```

Develop understanding of fractions as numbers.
CCSS.Math.Content.3.NF.A.1
Understand a fraction 1/b as the quantity formed by 1 part when a whole is partitioned
into b equal parts; understand a fraction a/b as the quantity formed by a parts of size 1/b.

```
CCSSMathContent["Grade 3", "NF", "A.2"]
```

Develop understanding of fractions as numbers.
CCSS.Math.Content.3.NF.A.2
Understand a fraction as a number on the number line; represent fractions on a number line diagram.

The Common Core introduction: "In Grade 6, instructional time should focus on four critical areas: (1) connecting ratio and rate to whole number multiplication and division and using concepts of ratio and rate to solve problems; (2) completing understanding of division of fractions and extending the notion of number to the system of rational numbers, which includes **negative numbers**; (3) writing, interpreting, and using expressions and equations; and (4) developing understanding of statistical thinking."

4.4. Routines and stages

Educational settings can contain various elements or combinations: positive numbers, zero, numbers with *H,* or a mixture including also the numbers with the minus sign. To provide for the stages and this richness in educational settings, it is useful to have a supporting richness in routines.

In below table, the first three routines or rules are discriminatory on what they exclude, and the last three routines or rules are discriminatory on what they include. While kindergarten would not read and write (at least not what we are currently discussing), kindergarten teachers and researchers however can use some routines to help design the curriculum and lesson plans.

For this table it is important to understand that -1, -2, -3, ... are hard-coded in *Mathematica.*

Adapts	*-expr*	-2, -3, ...	-1	Times[-1, b]	Power[b, -1]	*Stages*
RuleNegativeToH	yes	yes	yes	yes	yes	1 - 5
MinusSignToH	-	yes	yes	yes	yes	1 - 5
RuleAnyToH	depends	-	yes	yes	yes	6
TimesToH	-	-	-	yes	-	1 - 6
ToH	-	-	-	-	yes	7
BothToH	-	-	-	yes	yes	6

```
{(-1 /. RuleNegativeToH),   MinusSignToH[-1],   (-1 /. RuleAnyToH)}
```

$\{H, H, H\}$

```
{TimesToH[-1],  ToH[-1],   BothToH[-1]}
```

$\{-1, -1, -1\}$

? RuleNegativeToH

RuleNegativeToH is (x_?Negative :→ – H x).
Observe that it also affects Power[x, –1], which might not be intended

? MinusSignToH

MinusSignToH[expr] applies RuleMinusSignToH, has HoldFirst, and leaves the result into HoldForm. The occurence of H (H n) is turned into n H H instead of n H^2

? RuleAnyToH

RuleAnyToH replaces –1 by H, using only –1 –> H. This substitutes indiscriminately but leaves –2, –3, ... unaffected. See TimesToH and ToH for particular patterns (not using Inactive), and RuleMinusSignToH for the negative integers

? TimesToH

TimesToH[expr] replaces variables –x by H x. Has HoldFirst, applies RuleTimesToH, and then ReleaseHold. For the negative integers use RuleNegativeToH, though this might affect Power[x, –1] too

? ToH

ToH[expr] checks whether some elements can be turned into a Rational number with a MixedNumber format, then turns all Power[x, –1] expressions into x^H.
It has property HoldFirst, applies the replacement routines to the unevaluated expression, and then has ReleaseHold. ToH works like a "PowerToH" (deliberately not defined here), and internally handles RationalToH and PowerHeadToH

? BothToH

BothToH[expr] is TimesToH[ToH[expr]], with HoldFirst. The idea is that education proceeds in steps, such that TimesToH is important first, BothToH intermediately, and later only ToH

4.5. Routines for education

We mimic students and do not suppose that they will use these routines. The intention of these routines is not to find a final solution, which *Mathematica* already provides, but to mimic the steps that students would take. This appears to require attention, as *Mathematica* may have the tendency to simplify $4\ 2^H$ into 2^{2+H} while we have no need for that.

- Most routines *func*[*expr*] have supporting rules (Rule*func*}. Functions may use (various) rules once or repeatedly depending upon the implementation. *func*[*expr*] might be thought of as a button, while the rules might be applied for separate steps. A strategy is to provide functions that show the steps, like **HToZero**[*expr*] here. However, questions can be very diverse, HToZero is a lone example of such a routine, and the present focus is on the principles of design rather than providing such routines.
- The routines transforming from minus and negative to *H* have a HoldFirst property. Translating 1/2 + 1/ 4 into the format with *H* should not first simplify into 3/4. An alternative design would be to require HoldForm[1/2 + 1/4] as input, but this seems overdone for such frequent occurrence. One may always use ClearAttributes[*f*, HoldFirst], see the routine **HoldFirstH** that can do this clearing and setting for the four main routines.
- Pupils will first use $x + H\,y$ and exercises will be to simplify (find "the answer"). This involves use of **HToZero**[*expr*], **AlgebraH**[*expr*] and **SimplifyH**[*expr*]. When learning minus, they would use **MinusSignToH**[*expr*] for checking up. Simplification by means of AlgebraH removes the HoldForm. However, at other places *Mathematica* may change 4^H into 2^{2H}, and then we require some redress.
- Once the minus sign has been mastered, *H* is only relevant for the exponent. Pupils would learn rationals and mixed numbers by means of exponentiation. We would want them to know a bit about the fraction bar too, given its common use. They would learn that $y / x = y\,x^H$, but they would not have to learn the more complex rules for the fraction bar, such as $1 / (y / x) = x / y$. The latter rule in algebra is $(y\,x^H)^H = y^H\,(x^H)^H = y^H\,x$. For checking up, **ToH**[*expr*] remains the relevant routine for the remainder of school. PM. Deliberately, it isn't called PowerToH, since the latter might be confusing in *Mathematica*, because of the distinction witin *Mathematica* between Rational[*n, m*] and *n* Power[*m*, -1].
- There might be an intermediate stage. When division and its exponential form are introduced, then this might be in combination with subtraction. Then **BothToH**[*expr*] is the relevant routine. As said, these routines test on particular patterns. When this would be too subtle, then **RuleAnyToH** replaces -1 by *H* indiscriminately. If the numbers with the minus sign are to be included, e.g. $-3 \to 3\,H$, then use **RuleMinusSignToH**. This however also affects the representation of the rationals.
- PM. In all cases **FromH**[*expr*] substitutes $H \to -1$, and *Mathematica* will provide for an expression with a fraction bar if this happens conventionally.

4.6. A summary overview

We may summarise our discussion.

In elementary education, kids learn addition and subtraction first. When kids count from 6 to 7 then they also count back, and when they add 6 + 1 = 7 then they can subtract 7 - 1 = 6. For the lowest levels, with nonnegative numbers, $a - b \geq 0$ would be unproblematic, except when it causes a lock-in on absolute differences, or when prevention of the lock-in comes at a loss of the awareness of cancellation.

Currently grouping (multiplication) is only discussed informally in Grade 1-2 and formally in Grade 3. The development of the place value system will benefit when pupils will have seen grouping (multiplication) with 5 of 4 = 5 × 4 = 20.

Our topic arises when the negative numbers are introduced, with the stumbling block of $a - (-b)$. We would not have negative surfaces and volumes yet, but negative co-ordinates are quite practical.

At that didactic moment, the use of H is a foundational step. The use of H helps kids to start thinking algebraically, a bit more abstract than working with numbers only (though the numbers are quite abstract too).

- *The key notion:* The creation of the negative numbers better is not confused with the minus sign as the pupils have been using for $a - b \geq 0$.
- H creates the negative numbers by rotating or mirroring the positive numbers.
- H involves multiplication, namely scalar multiples as different from grouping.
- This allows vector addition along the number line. "Arrow 10 to the right plus arrow 3 H to the left results into arrow 7 to the right."
- Algebraically we have negative number $b\ H$ and addition $a + b\ H$, and the rule that $a + a\ H = 0$.
- Applying this rule on $a\ H$ itself gives $a\ H + a\ H\ H = 0$ and thus $H\ H = 1$.
- At this stage, pupils will have experience with $a - b \geq 0$. They discover that this is the same as $a + b\ H \geq 0$. They get used to the extended use of plus with $a + b\ H = c\ H < 0$. A simple example is: $2 + 3\ H = 2 + 2\ H + H = 0 + H = H$.
- The extended use of subtraction gives $a - H\ b = a + H\ H\ b$. The rule is that $H\ H = 1$.
- Pupils will first work with H as symbol only, but at some stage the value $H = -1$ can be identified.
- H fully covers subtraction on *content*. What remains is the *notation* that uses the minus sign. This is notational only.

The *use of the minus sign* involves: using a single sign for both a single operator for rotating or mirroring $H\ b$ and an operator for more numbers $a + H\ b$. Subtraction arose in history perhaps only with a confusion of these two properties, yet eventually there was deliberate design, and we can recognise the flexibility and economy of symbols, though no respect for the problems arising in education.

Using H we can define, as another way of saying the same, and identifying $H = -1$ (though still using H symbolically, i.e. unevaluated and algebraically):

- $-b = H\ b$. Pupils could agree that -3 writes and reads faster than H 3. Also short for $0 - b$.
- $a - b = a + (-b) = a + H\ b$. Pupils could agree that $a - b$ writes and reads faster than $a + H\ b$.
- The price is: $a - (-b)$ but we now know $a - H\ b = a + H\ H\ b = a + b$, with the algebraic rule $H\ H = 1$.

Thus subtraction involves multiplication. When $a - (-b)$ is presented as subtraction, and turns out to be addition then we may understand why kids find this hard, for this important aspect of multiplication is not mentioned or explained.

Alternatively, H is used only in remedial teaching. However, remedial teaching would only work if it uses better methods: and if the use of H would be considered a better method, then it should be used for all. Remedial teaching rather works by taking the same steps at a tailored pace.

Once subtraction has been mastered, H will no longer be used for subtraction, and then retains its value for division i.e. exponentiation. The development of the multiplicative inverse will benefit when kids in Grade 3 have seen squares 5 of 5 = 25. It likely is possible to introduce the expression x^2 for surface and x^3 for volume, but given the handwriting we would use x^2 and x^3 or colours for exponents first.

It remains a point that pupils might confuse x^H with $x\ H$, and it might be that they need something like x^H or perhaps even a fraction bar in y / x in order to observe the correct operation. However, pupils can observe minute differences, if only alerted on them, and it likely is only a matter of finding the right stepping stone. Perhaps a different colour might be used in the first steps in exponentiation. Only empirical research can show what would work.

5. Conclusions

For *education:*

- This notebook with packages is for teachers, researchers and educators on math education. It provides a working environment for research and experiments on the use of H = -1 in education. Kids would work with pencil and paper. (Though it is another issue: kids would better also be introduced into computer algebra - a bit more refined than mere “programming”.)
- Hopefully the mathematics education (ME) and its research (MER) community including the educators (e.g. at CCSS) agrees that mathematics education and its research need to return to the drawing board.
- Overall, ME & MER are in a mess, and we cannot expect that this proverbial Baron of Münchhausen will be able to lift himself from the morass he is in. I am writing as an economist now, who has done some study on institutional economics. My suggestion is that each nation sets up a National Center for Mathematics Education, with a governance open to the stakeholders: parents, teachers, researchers, government, publishers and employers, based upon the principles and ethics of science in open access fashion, namely that only empirical methods allow the kids to tell us what works for them. See Colignatus (2009, 2015a), *Elegance with Substance* and (2015b), *A child wants nice and no mean numbers*.

For *Mathematica:*

- It would be best that H is included in *Mathematica’s* list of mathematical constants: tutorial/MathematicalConstants. This actually also holds for *Đ* = 10 (deca, using capital Eth), so that $Đ^H$ = “per 10” = decim. The above indicates the properties that would be useful for such an implementation as well. Obviously the functionality changes when Simplify, N, and rules for pattern recognition are adapted. Wise choices must be made here.
- H is already supported to a large extent by Plus & Times[-1] and Times & Power[-1]. Thus the proposal to properly include H as a universal constant only makes the structure more coherent. The symbolic use of H in these functions only applies to a particular phase in education, but could be supported by an option in the Preferences. Tongue in cheek: If *Mathematica* supports the use of different natural languages, then hopefully also for the language of mathematics.

On the overlap, see **Appendix G** on terminology and copyrights.

6. Stage 1. Kindergarten. Place value

The following only indicates my own thinking without empirical testing, see above and Colignatus (2015b) for my disclaimer on competence and empirical testing.

There are steps for the integers 0-10 (kindergarten and Grade 1), 0-20 (kindergarten and Grade 1) and 0-100 (Grade 1). There is the difference between scalar multiplication (2 km) and grouping (5 baskets of 4 apples).

The US Common Core is ambitious for kindergarten and includes 20:

“Students use numbers, including written numerals, to represent quantities and to solve quantitative problems, such as counting objects in a set; counting out a given number of objects; comparing sets or numerals; and modeling simple joining and separating situations with sets of objects, or eventually with equations such as 5 + 2 = 7 and 7 – 2 = 5. (Kindergarten students should see addition and subtraction equations, and student writing of equations in kindergarten is encouraged, but it is not required.) Students choose, combine, and apply effective strategies for answering quantitative questions, including quickly recognizing the cardinalities of small sets of objects, counting and producing sets of given sizes, counting the number of objects in combined sets, or counting the number of objects that remain in a set after some are taken away.”

```
CCSSMathContent["Kindergarten", "CC", "A.3"]
```

Know number names and the count sequence. CCSS.Math.Content.K.CC.A.3
Write numbers from 0 to 20. Represent a number of
objects with a written numeral 0–20 (with 0 representing a count of no objects).

We better define “number” as “pronunciation, numeral and word”. Kids in kindergarten and Grade 1 live in a world of sounds. My suggestions:

- There seems to be a division of labour, and division of perks of competence, in which the language department tells to mathematics education: “You can have the numerals, and we keep the words and their pronunciation.” It doesn’t work like this. Kindergarten should focus on *proper definitions* and *vocabulary*. Kids at that age still live in a *world of sounds,* and those sounds better be helpful rather than counterproductive. It is okay to use common English at home, regarding English as a dialect of the language of mathematics, but in school the numbers 11-20 better be pronounced ten & one, ten & two, ..., ten & nine and two·ten. This pronunciation supports the decimal place value system, and the later tables of addition and subtraction. See Colignatus (2015f, 2018c) on *The need of a standard for the pronunciation of the integers,* and Colignatus (2018a) on *Pronunciation of the integers with full use of the place value system.*

PM. The trailing center dots are deliberate, see Colignatus (2018a).

```
PlaceValue[#, Speak → False] & /@ Range[11, 15]
```

{·ten & ·one, ·ten & two·, ·ten & three·, ·ten & four·, ·ten & five·}

It is also important to realise that when counting from 0-9, and when the (single) digits are exhausted, that the next *ten* introduces the *new base* in the place value system. A teacher should also *say* so. It is simple to explain and it might be quite confusing when it isn’t explained. Perhaps a car salesman doesn’t have to explain to who buys a car that they should also get a driver’s license but we cannot assume that kids understand the place value system without proper explanation.

Proper explanation of the current numbers would tell kids that “eleven” is “one left over” after subtracting 10, and that “twelve” is “two left over” after subtracting 10, but when teachers do not take the effort to explain this, then this only means that they have no interest in the pronunciation of the words, are blind to the importance of pronunciation, and see the words only as labels instead as functionally relevant for the place value system that they are supposed to teach.

- The following table will be used by kindergarten teachers and not the pupils when they cannot read and write yet. It has already been printed above so we now use small print.

```
TableOfBasicAddition[]
```

10	10										
9	9	10									
8	8	9	10								
7	7	8	9	10							
6	6	7	8	9	10						
5	5	6	7	8	9	10					
4	4	5	6	7	8	9	10				
3	3	4	5	6	7	8	9	10			
2	2	3	4	5	6	7	8	9	10		
1	1	2	3	4	5	6	7	8	9	10	
0	0	1	2	3	4	5	6	7	8	9	10
+	0	1	2	3	4	5	6	7	8	9	10

- Key didactics is to properly *internalise* the "*Table of Basic Addition*", since this is the step from counting to addition. Pupils who get stuck at counting - still using their fingers - will not have properly *internalised* this table. (Its "version in sounds" is part of memorisation.)
- The property of TOBA[10] is that kids only know the numbers up to some level, and they can only internalise what they already know, whence we get a triangle shape. Using a square table would generate 20, though see next Section. NB. The table is oriented like the system of co-ordinates, so that “1 is lower than 10” is visually supported and not contradicted.
- The *Table of Basic Addition* also shows both counting back and subtraction. Pick a number within the table, say 7, and see how it can be found, say 7 = 5 + 2 = 2 + 5. Thus it is also possible to say that 7 - 2 is 5. Again, it are kindergarten teachers who will use this table to have discussions with the kids. It is not clear to me whether this table could already be shown to the kids themselves.
- Use standard *plus* and *minus* and no other local inventions (e.g. Dutch so-called-easy “erbij” and “eraf”). One can also say that the difference between 7 and 5 is 2, so that the difference is given by the largest minus the smallest.

7. Stage 2. Grade 1 & 2. Addition and subtraction in the nonnegative realm

The key phenomenon is that we have learning to read and write now.

7.1. Mathematics is language too

The US Common Core for Grade 1 states, and note the word “grouping”:

“In Grade 1, instructional time should focus on four critical areas: (1) developing understanding of addition, subtraction, and strategies for addition and subtraction within 20; (2) developing understanding of whole number relationships and place value, including **grouping** in tens and ones; (3)

developing understanding of linear measurement and measuring lengths as iterating length units; and (4) reasoning about attributes of, and composing and decomposing geometric shapes."

See Colignatus (2018a) for the explanation that one should not use "ones", "tens", "hundreds".

The term "grouping" is not explained in the CCSS Glossary. It is not clear to me whether such grouping in Grade 1 may also be seen as a *de facto* competence on multiplication. Apparently grouping is only introduced "informally" and not "formally" with the multiplication sign. However, this creates an awkward distinction between "words" and "mathematical symbols" while such symbols are words *too*. It is proper to use words precisely and thus also the symbols. Currently there are "language-people" who take some distance from mathematics and who seem to decide what "language' would be, but they cannot have this prerogative when they clearly show bias by excluding mathematics from language. Thus, do not fall in the trap of distinguishing "informal" and "formal" and other terms that "language-people" use to express their confusions, Thus, keep emphazising clarity. Be kind to the "language-people" though, because they are victims of the mess that mathematics education is in (but do not accept their belief that mathematics education already is perfect).

When Grade 1 is introduced to grouping then this should be supported by words (symbols), (symbolic) expressions, graphs, tables. Check CWNN*. Rather than "3 times 2" we may better speak about "3 of 2", and also speak about "grouping 3 groups of 2".

7.2. Table Of Basic Addition

CCSSMathContent["Grade 2", "OA", "A.1"]

Represent and solve problems involving addition and subtraction. CCSS.Math.Content.2.OA.A.1

Use addition and subtraction within 100 to solve one– and two–step word problems involving situations of adding to, taking from, putting together, taking apart, and comparing, with unknowns in all positions, e.g., by using drawings and equations with a symbol for the unknown number to represent the problem

CCSSMathContent["Grade 2", "OA", "B.2"]

Add and subtract within 20. CCSS.Math.Content.2.OA.B.2

Fluently add and subtract within 20 using mental strategies.

[Footnote] By end of Grade 2, know from memory all sums of two one–digit numbers.

Would A.1 have priority over B.2 ? Why the last part of B.2 so late ? It seems to me that CCSS is *confused:* The TableOfBasicAddition is *basic* and must be mastered in Grade 1, to make the transition from counting to addition, and thus be mastered *before* working till 100. If it hasn't been mastered, do remedial teaching. (A remedial teacher might allow kids to solve 5 + 95 = 100 by counting from 5 to 100, merely to show them that 95 + 5 is faster, and so on. Use nine·ten & five, instead of dialect nine-tee-five.)

Thus, kids will now read and write too, to internalise the *Table of Basic Addition* for 0-10*,* see above.

A *Table Of Basic Addition* is triangular, except when negative integers are included. Thus the integers to 20 (two·ten) should rather not be introduced by a square table but by the following triangle (printed small).

```
TableOfBasicAddition[0, 20]
```

20	20																				
19	19	20																			
18	18	19	20																		
17	17	18	19	20																	
16	16	17	18	19	20																
15	15	16	17	18	19	20															
14	14	15	16	17	18	19	20														
13	13	14	15	16	17	18	19	20													
12	12	13	14	15	16	17	18	19	20												
11	11	12	13	14	15	16	17	18	19	20											
10	10	11	12	13	14	15	16	17	18	19	20										
9	9	10	11	12	13	14	15	16	17	18	19	20									
8	8	9	10	11	12	13	14	15	16	17	18	19	20								
7	7	8	9	10	11	12	13	14	15	16	17	18	19	20							
6	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20						
5	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20					
4	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20				
3	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20			
2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20		
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
+	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

It is true, though, that the (non-basic) *Tables of Addition* for the integers [0, 10] generate a square matrix. This is where the number 20 enters the scene. Tradition has focused on this table rather than the triangle, and perhaps it suffices indeed: with the other values recoverable by implication (1 + 19 = 1 + 9 + 10 etcetera). The table is again oriented on the system of co-ordinates (supporting the reading of maps in kindergarten and Grade 1). PM. The parallel for the TOBA[10] would be to use the square matrix for 5, with at most 5 + 5 = 10. It is better to use the TOBA[10] itself however, when kids still have to learn the basics.

```
SquareTableOfAddition[]
```

10	10	11	12	13	14	15	16	17	18	19	20
9	9	10	11	12	13	14	15	16	17	18	19
8	8	9	10	11	12	13	14	15	16	17	18
7	7	8	9	10	11	12	13	14	15	16	17
6	6	7	8	9	10	11	12	13	14	15	16
5	5	6	7	8	9	10	11	12	13	14	15
4	4	5	6	7	8	9	10	11	12	13	14
3	3	4	5	6	7	8	9	10	11	12	13
2	2	3	4	5	6	7	8	9	10	11	12
1	1	2	3	4	5	6	7	8	9	10	11
0	0	1	2	3	4	5	6	7	8	9	10
+	0	1	2	3	4	5	6	7	8	9	10

7.3. Abacus

For Grade 1, the abacus would be important for addition and subtraction in the realm of [0, 20]. Having no physical abacus, one might use pictures (graphs) but at a loss of physical interaction. The abacus works like vector addition and subtraction (a string of beads is like an arrow). PM. I haven't looked for uses of the abacus for negative numbers.

7.4. Place value and CWNN*

Kids would use the place value system, pronounce the numbers in full place value manner, and do addition and subtraction in place value manner too. There are numerous expositions about introducing kids to the place value system. What I have seen hide the negative numbers. The following uses Colignatus (2018ab), and sets the stage for using *H*.

I intended to write some routines to support x^H, got amazed by the treatment of the negative numbers, and intended to only briefly mention Colignatus (2015b), *A child wants nice and no mean numbers* (CWNN) (pdf online), now amended by (2018a), so together CWNN*, but I now must refer to it with emphasis. This includes kindergarten and Grade 1 into this paper. CWNN* namely develops the place value system with proper pronunciation of the numbers, also clarifying the role of grouping. (Copied to above.) To emphasize the point, and to lower the barrier for others to check the argument, I wrote (2018a) and include its package also in this notebook.

When the integers are pronounced correctly, say 24 = two·ten & four, then it is also clear that 2 × 10 + 4 = 24, so that the pronunciation generates the answer to the sum.

CCSS is seriously confused. Multiplication is only introduced in Grade 3, while the place value system uses multiplication.

```
CCSSMathContent["Grade 2", "NBT", "A.1"]
```

Understand place value. CCSS.Math.Content.2.NBT.A.1
Understand that the three digits of a three−digit number represent amounts of hundreds, tens, and ones; e.g., 706 equals 7 hundreds, 0 tens, and 6 ones. Understand the following as special cases: A.1.A, A.1.B

Apparently, for CCSS "7 hundreds, 0 tens, and 6 ones" does not representmultiplication ? CCSS has grouping as something informal:

```
CCSSMathContent["Grade 2", "OA", "C.3"]
```

Work with equal groups of objects to gain foundations for multiplication. CCSS.Math.Content.2.OA.C.3
Determine whether a group of objects (up to 20) has an odd or even number of members,e.g.,by pairing objects or counting them by 2s;write an equation to express an even number as a sum of two equal addends.

It is a rather tricky: to speak about odd and even, when *even* is defined as having factor 2, but you will not mention its definition because you won't refer to grouping in proper ("formal") manner. CCSS should be able to see that this is incoherent. See also the reference to grouping above.

Thus I would suggest that Grade 1 already has grouping, in decent mathematical manner, with unity of words, symbolic expression, graphs and table. Some people might call this "formal" but it is only the use of words (symbols) with proper *precision*. The following are the groupings for outcomes of maximal 10. With this table you can have a discussion about odd and even with proper definition. It may be convenient to already speak about the "table of 2, max 10" since we haven't 20 yet. The term "times" does not express what it does, and it is better to use terms with "group": Grouping three groups of two = 3 × 2 gives a group of 6. For convenience I drop the zero's.

```
TableOfBasicGrouping[] /. 0 → ""
```

10	10									
9	9									
8	8									
7	7									
6	6									
5	5	10								
4	4	8								
3	3	6	9							
2	2	4	6	8	10					
1	1	2	3	4	5	6	7	8	9	10
×	1	2	3	4	5	6	7	8	9	10

Let me also show the TOBG for [-10, 10], printed smaller for page width. It would belong to a later Stage, but it adds perspective to the current argument on the relevance of grouping from the beginning. The table helps see that two negatives generate a positive.

```
TableOfBasicGrouping[-10, 10] /. RuleMinusSignToH
```

10										10 H	0	10									
9										9 H	0	9									
8										8 H	0	8									
7										7 H	0	7									
6										6 H	0	6									
5									10 H	5 H	0	5	10								
4									8 H	4 H	0	4	8								
3								9 H	6 H	3 H	0	3	6	9							
2						10 H	8 H	6 H	4 H	2 H	0	2	4	6	8	10					
1	10 H	9 H	8 H	7 H	6 H	5 H	4 H	3 H	2 H	H	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
H	10	9	8	7	6	5	4	3	2	1	0	H	2 H	3 H	4 H	5 H	6 H	7 H	8 H	9 H	10 H
2 H						10	8	6	4	2	0	2 H	4 H	6 H	8 H	10 H					
3 H								9	6	3	0	3 H	6 H	9 H							
4 H									8	4	0	4 H	8 H								
5 H									10	5	0	5 H	10 H								
6 H										6	0	6 H									
7 H										7	0	7 H									
8 H										8	0	8 H									
9 H										9	0	9 H									
10 H										10	0	10 H									
×	10 H	9 H	8 H	7 H	6 H	5 H	4 H	3 H	2 H	H	0	1	2	3	4	5	6	7	8	9	10

7.5. Place value and the tables

We first teach ToSingleDigits and then ToSingleDigitsTable, and subsequently the tables of addition and subtraction.

7.5.1. Grade 1

For Grade 1 and numbers in [0, 20], the routines ToSingleDigits (-Table) have little use. The routines basically concatenate the digits here, and the real exercise is above TableOfBasicAddition[0, 20]. However, we can these routines now to highlight a conceptual issue.

Above we already met the CCSS “0 tens” that CCSS does not explain as a multiplication but something magically different. The proper term is “0 of 10” and “0 count (weight) for 10”.

```
ToSingleDigits["countfor10", "countfor1"] (*highest place value first*)
```

countfor1 + 10 countfor10

```
Evaluate[above] // Inactivate
```

countfor1 + 10∗countfor10

In the current CCSS setup for Grade 1, we cannot speak about 10 × countfor10 when we haven't had multiplication. *My inference is that the relevant parts of the table of 10 must be discussed together with the introduction of the place value system.*

Conventionally it requires new words like twenty and thirty, but if kids can distinguish 2 apples and 3 apples, then please don't treat numbers differently, and they should be able to distinguish two·ten and three·ten. A grid of marbles should clarify it.

I suppose that grouping might be restricted to the TOBG[10] and the table of ten for 0, 10, 20, and that the burden will fall on the proper pronunciation of the numbers in full place value manner. (My hope is that kids have also time to sing and dance.)

7.5.2. Grade 2

For Grade 2, our routines become relevant. The reason: the TableOfBasicAddition[0, 100] is too large, and we use an algorithm now, called "place value system".

```
ToSingleDigits["countfor100", "countfor10", "countfor1"]
```

countfor1 + 10 countfor10 + 100 countfor100

7.5.3. Example for teachers and researchers, not for kids now

NB. The following is not an addition table but its second part.

The following example is not intended to be used in class as such. It is only intended to show what the routine can do, and to highlight the didactic issues. We want kids to learn these manipulations in the place value system. Let 10 have count 9 and 1 have count 12.

```
{9, 12} → ToSingleDigits[9, 12]
```

{9, 12} → 102

We can see the steps by using ToSingleDigitsTable. For the next table, the number -10 is a negative number. Thus, negative numbers really exist in everyday life. (The meaning of a word is its use.) Who wants to avoid negative numbers might tell kids that writing -10 on line 2 below 12 is a subtraction, with the outcome 2. Then 1 must be recorded in the following column (right to left) (so that we do not change the sum). How complex is this ? There are only 2 steps per column, and quite openly: (i) insert {1, -10} if needed, (ii) determine the column sum.

Again: we cannot speak at this level about 1 × 10 - 10 × 1 = 0 when we haven't had multiplication.

```
ToSingleDigitsTable[9, 12] (*default differences → True*)
```

	10^2	10^1	10^0
1	0	9	12
2		1	−10
3	1	−10	
4	+	+	+
5	1	0	2

(PM. Row 4 reads "+" and thus we have 12 + (-10) = 2, while 12 - 10 = 2 rather has "=". But "+" can be defended by reference to the middle column and correctness overall.)

7.5.4. The current method hides the differences

The current convention is to hide the differences. The following hides the subtraction 12 - 10. Kids must do 3 steps *at the same time* per column: (i) 12 - 10 = 2, and record 2 at the bottom, (ii) move 10 to the following column and turn it into 1, (iii) 9 + 1 = 10, and record this outcome at the current line. Now do these three steps again for the 2nd column.

```
ToSingleDigitsTable[9, 12, Differences → False]
```

	10^2	10^1	10^0
1	0	9	12
2	0	10	2
3	1	0	2

7.5.5. A sequence for teaching and learning

The suggestion is the following sequence for teaching and learning:

1. Kids better *start* with differences, because it has fewer steps and shows them all.
2. Then proceed with levels, because this involves *internalisation* of the intermediate steps (and *less writing*). (Tell kids that the objective of the lesson is not that they understand the place value system (because the would by now), but that you want them to write less and still get the right answer.)
3. Eventually there will be *mental automation* with a higher level script: write down the units, remember the rest, add to the count of ten, write down the last digit of the outcome, etcetera.

When the only goal is to drill the latter automation, then we might forget about the first two steps. When the goal is to better grasp the place value system, then these two earlier steps are important, to also understand what the drilling is about.

7.6. Tables of addition and subtraction

We already discussed the table of addition (in differences and levels) for the didactics of the negative numbers. It is useful to focus now on Grade 1 and 2, and wonder whether the following tables would be useful for them. This section only lists these tables.

? AdditionTable

AdditionTable[n__Integer] gives a table that adds the integers,
that must be positive, using the place value system. The integers are added
per digit position, and each overflow generates a new line in the table
AdditionTable[{n__Integer}] is another input format
Options are passed on to ToSingleDigitsTable.
Default there is Differences –> True (default); otherwise levels
When all integers are in [0, 9] then there is a warning that the routine is overly complex for such a case

? SubtractionTable

SubtractionTable[n, m] gives a table for n – m, for n > m > 0, using the place value system. The integers
are subtracted per digit position, and each underflow generates a new line in the table. If
n < m then the routine switches the order. When subtracting a block of numbers, perhaps
first add the numbers that must be subtracted, so that subtraction only applies to n – m
SubtractionTable[n__Integer] (for more than 2) or SubtractionTable[{n__Integer}]
cannot switch the order though. Integers must be positive
Options are passed on to ToSingleDigitsTable. Default there is
Differences –> True (default); otherwise levels
When all integers are in [0, 9] then there is a warning that the routine is overly complex for such a case

? BothSubtractionTables

BothSubtractionTables[n__Integer] or [{n__Integer}]
jointly shows with Differences –> True and Differences –> False

? MixedAdditionTable

MixedAdditionTable[n__Integer] (for more than 2) or MixedAdditionTable[{n__Integer}]
adds positive or negative integers. Options are passed on to ToSingleDigitsTable.
Default there is Differences –> True (default); otherwise levels
When all integers are in [0, 9] then there is a warning that the routine is overly complex for such a case

? BothMixedAdditionTables

BothMixedAdditionTables[n__Integer] or [{n__Integer}]
jointly shows with Differences –> True and Differences –> False

7.7. Grade 1 & 2: Tables of addition and subtraction

Grade 1 would count and Grade 2 would add, but they might use the same tables.

7.7.1. Grade 1: Table of addition in differences or levels

Given the restriction to the numbers 0-20 these tables have little use, except that they show in our format that addition in the place value system works on the separate digits. Only the RHS in levels is useful now.

```
BothMixedAdditionTables[11, 7]
```

	10^1	10^0
1	1	1
2	0	7
3	+	+
4	1	8
5	+	+
6	1	8

	10^1	10^0
1	1	1
2	0	7
3	+	+
4	1	8

7.7.2. Grade 1: Table of addition with outcome 20

The outcome 20 would be the first serious type of application of the addition table method that kids would see. See the discussion above about the distinctions w.r.t. the LHS (differences) and RHS (levels). For the LHS (differences) in row 5 , there is a negative number, but we can tell the kids that we insert a subtraction in this column (10 - 10) = 0. Also, to compensate in the row, we must insert a 1 in the column of ten, thus with 1 × 10 - 10 × 1 = 0, but this invokes multiplication, and the creation of 20 actually is the first step in presenting the table of 10 (for 1 of 10 is just 10).

```
BothMixedAdditionTables[11, 9]
```

	10^1	10^0
1	1	1
2	0	9
3	+	+
4	1	10
5	1	−10
6	+	+
7	2	0

	10^1	10^0
1	1	1
2	0	9
3	+	+
4	1	10
5	2	0

7.7.3. Grade 1 & 2: Table of addition getting into 11-20

Addition in [0, 20] better uses the abacus, and certainly when crossing 10. See Colignatus (2018a) for counting from 8 to 13 in full place value manner. When we do an addition with numbers in [0, 10] so that the result is in [11, 20] then the addition routine gives a warning that it is overly complex. The outcome in the first column on the RHS is already in [11, 20] and there is no need to transfer digits merely because this is how we have structured our table. But it is interesting to observe what the computer produces by sticking to the algorithm. Perhaps, though, the minds of the kids are as logical as the computer, and they might agree that 10 is transferred from the right column to the left column. Indeed, some kids might perform this method, and then get stuck for not knowing how to proceed. My guess though is that this would be self-evident from the earlier training on the TableOfBasicAddition and the abacus, so that it would be wasted teaching time and actually counterproductive to spend attention to it. It might be something for remedial teaching for kids who get stuck on this indeed. (Perhaps this explains why kids find it hard to pass 10.)

```
BothMixedAdditionTables[6, 7]
```

MixedAdditionTable: Numbers all in [–9, 9]: routine is overly complex

MixedAdditionTable: Numbers all in [–9, 9]: routine is overly complex

	10^1	10^0
1	0	6
2	0	7
3	+	+
4	0	13
5	1	−10
6	+	+
7	1	3

	10^1	10^0
1	0	6
2	0	7
3	+	+
4	0	13
5	1	3

7.7.4. Grade 1 & 2: Table of subtraction, for big minus small, without crossing 10

For subtraction and outcomes $a - b \geq 0$, the routine can generate negative numbers in single digit comparisons. In Grade 1 & 2 we thus use only big - small for *all* digits. There is little use of the place value system, except that these tables show that we work per digit position indeed, so that kids can get used to this table format, preparing them for more complex cases.

```
SubtractionTable[18, 2, 5, Differences → False]
```

	10^1	10^0
1	1	8
2	0	2
3	0	5
4	−	−
5	1	1

7.7.5. Grade 1 & 2: Table of subtraction from above to below 10

The situation is somewhat different when we start above 10 and subtract to below 10. In this case, the first column on the RHS must “borrow” (and never give back) from the second column.

This is done best by using an abacus (with vector addition) and rehearsing the Table of Basic Addition for [0, 20]. Obviously 18 - 9 = 9 + 9 - 9 = 9 is a strategy.

Alternatively, the method in the subtraction table is sound and will be used for higher figures, and it would represent a case to argue that negative numbers have a natural place in Grade 1. Or, simply do not do such subtractions until the negative numbers have been introduced in Grade 2.

```
BothSubtractionTables[18, 9]
```

	10^1	10^0
1	1	8
2	0	9
3	–	–
4	1	−1
5	−1	10
6	+	+
7	0	9

	10^1	10^0
1	1	8
2	0	9
3	–	–
4	1	−1
5	0	9

7.8. Grade 2: Tables of addition and subtraction

Using numbers in [0-100] with outcomes in [0, 100] can no longer by done (quite) by internalising a table, and internalisation is actually superfluous since we have a place value system that allows an algorithm. This algorithm has been put into the tables of addition and subtraction, which explains their genesis and use. Obviously we might also use the abacus but that would change the subject. See Colignatus (2018b) for further discussion of using the tables without *H*.

8. Stage 3. Grade 2. Introduction of the negative integers, i.e. *H*

The following is only a definition. This can be done in 15 minutes but requires accuracy. The introduction here concerns *H,* and the use of *-x* is only done at Stage 6.

```
TextRotate[][10]
```

{10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

```
TextReflect[10]
```

{10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

```
IntegerLineH[10]
```

{10 *H*, 9 *H*, 8 *H*, 7 *H*, 6 *H*, 5 *H*, 4 *H*, 3 *H*, 2 *H*, *H*, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

The nonnegative half-line can be turned around the circle. One might ask a pupil to stand upon a line, make some steps, let another pupil instruct *H* and another pupil instruct the number of steps, and see where the walker lands.

This completes the introduction, since it is only a definition. We quickly proceed with the next Stage 4, namely the rule that $x + H\,x \to 0$.

```
? IntegerLineH
```

IntegerLineH[n, m} gives a list of integers, from n to the left of 0, to m to right of 0. Input of negative numbers like −n and −m is allowed

IntegerLineH[n] is IntegerLineH[n, n]

```
? TextRotate
```

TextRotate[theta:(1/2)][string] shows string in rotation by theta * 2 Pi
TextRotate[theta:(1/2)][n] shows the list of
integers 0, ..., n (standing for the number line) rotated around 0
TextRotate[theta:Pi][s, m:1.0, w:12, h:12, opts] is the full call, with Magnification
-> m, FontSize -> w, and ImageSize height UpTo[m h]. Options
are passed onto BaseStyle of Graphics object Text, e.g. FontFamily

```
? TextReflect
```

TextReflect[string] shows string in reflection
TextReflect[n] shows the list of integers 0, ..., n (standing for the number line) reflected around 0
TextReflect[s, m:1.2, w:12, h:12, opts] is the full call, with
Magnification -> m, FontSize -> w, and ImageSize height UpTo[m h].
Options are passed onto BaseStyle of Graphics object Text, e.g. FontFamily

9. Stage 4. Grade 2 & 3. Addition with *H*, outcomes still nonnegative

We now consider steps that would be taken in education. We mimic what kids would do with pencil and paper. Where we use a variable, kids would use various numerical examples until they get the scheme (or we should have shown them wat a variable is).

We can deal with expressions with *H* via *algebra* or *tables of addition and subtraction*. Before we look at such tables, we better first master some algebra (for the tables rely upon this too, actually).

9.1. Working with H: half turn (rather than mirroring)

9.1.1. Meaning

Discuss the meaning.

```
a + H b
```

$a + b\,H$

Half turns around the circle. Expression $H\,b$ already requires a notion of multiplication: rotating from the right to the left. (And $H\,H\,b$ would rotate back to the original again, but we will see this later.)

9.1.2. Relation to 0, via RuleHToZero

The definition of *H* gives that $a + H\,a = 0$. Starting from 5 and seven steps to the right and seven steps back lands us in 5 again.

```
a + H a
```

$a\,H + a$

```
above      /.  RuleHToZero
```

0

There normal numbers give cookies and numbers with H give hungry cookie-monsters. When a hungry cookie-monster eats one cookie then its stomach is full. Thus $7 + 5\,H = 2 + 5 + 5\,H = 2 + 0 = 2$ or leaves 2 cookies. There are more ways to implement it.

9.1.3. Relation to 0, via AlgebraH

Simplify.

```
a + H a
```

$a\,H + a$

```
Collect[above, a]
```

$a\,(H + 1)$

```
above     // AlgebraH
```

0

```
First[RuleAlgebraH]
```

$H + 1 \to 0$

```
? AlgebraH
```

AlgebraH[expr] applies RuleAlgebraH to expr repeatedly, which is algebraic and tends to keep H. With HoldFirst and then ReleaseHold. Use FromH[expr] to fully eliminate H

AlgebraH[HoldForm[expr]] calls AlgebraHHelper[HoldForm[expr]], applies RuleAlgebraH repeatedly, and calls ReleaseHold

9.2. HoldForm for *H H*: rotate back to the original

When H is a half turn, then $H\,H$ is a full turn, and discussing this immediately may contribute to clarity what a half turn is. Perhaps mention this only, because the following requires algebra: "When you encounter a term $H\,H$ then this means a full turn, and you can replace it with 1." (And 2 turns still mean 1 turn ...)

In the main body of the paper: "When $a + H\,a = 0$ or that a finds an inverse or cancellation in $H\,a$, then an obvious question is what cancels $H\,a$ itself. Then we get $H\,a + H\,H\,a = 0$. What do you see ?"

PM. We can keep $H\,H$ without brackets or square H^2 by using HoldForm.

```
MatrixForm[{H # + # == 0, H # + HoldForm[H H #] == 0}] & /@ {2, 5, 9}
```

$$\left\{\begin{pmatrix} 2\,H + 2 = 0 \\ 2\,H + H\,H\,2 = 0 \end{pmatrix}, \begin{pmatrix} 5\,H + 5 = 0 \\ 5\,H + H\,H\,5 = 0 \end{pmatrix}, \begin{pmatrix} 9\,H + 9 = 0 \\ 9\,H + H\,H\,9 = 0 \end{pmatrix}\right\}$$

Discuss the meaning of the double H: rotate back to the original value.

```
a + H H b   // HoldForm
```

$a + H\,H\,b$

```
above /. RuleAlgebraH
```

$a + b\,1$

```
above  // AlgebraH
```

$a + b$

AlgebraH also applies a ReleaseHold. AlgebraH actually recognised the earlier HoldForm and then called the AlgebraHHelper to deal with that special format.

9.3. More terms in algebra

9.3.1. More terms, using RuleHToZero

Simplify.

```
HoldForm[2 + 3 H + 5 + 2 H]
```

$2 + 3\,H + 5 + 2\,H$

```
above   //  SortH
```

$2 + 5 + 2\,H + 3\,H$

```
above  /. RuleHToZero  (* recognises 2 + 2 H → 0 *)
```

$5 + 3\,H + 0$

```
above  /. RuleHToZero (* now uses stronger decomposition*)
```

$0 + 2$

```
above // ReleaseHold
```

2

9.3.2. More terms, using HToZero

```
2 + 3 H + 5 + 2 H  // HoldForm
```

$2 + 3\,H + 5 + 2\,H$

```
above // HToZero    (* collects terms, does not recognise 2 + 2 H *)
```

$$\begin{pmatrix} 2 + 5 + 2\,H + 3\,H \\ 7 + 5\,H \\ 2 + 5 + 5\,H \\ 2 \end{pmatrix}$$

9.3.3. More terms, using Inactivate

Use Inactive to show intermediate steps, but not so very enlightening.

```
Inactivate[2 + 3 H + 5 + 2 H]
```

$2 + 3 * H + 5 + 2 * H$

```
above // Sort
```

$2 + 5 + 2 * H + 3 * H$

```
above == Activate[above]   (* collects terms *)
```

$2 + 5 + 2 * H + 3 * H = 5 H + 7$

RuleHToZero cannot deal with Inactive[Plus] etc. and deals only with the RHS.

```
above  /. RuleHToZero
```

$2 + 5 + 2 * H + 3 * H = 2$

9.4. Place value using *H*

We had this table above but now use the variant with *H.*

Perhaps we should consider that kids in first grade already could handle $x + H x = 0$, so that the minus sign can be introduced later ? It is a cookie-monster eating a cookie, and would that be so hard to grasp if they already learn that 1 - 1 = 0 ?

```
ToSingleDigitsTableH[9, 12]   (* 2nd part of an addition table *)
```

	10^2	10^1	10^0
1	0	9	12
2		1	$10\,H$
3	1	$10\,H$	
4	+	+	+
5	1	0	2

When working with these differences has been internalised, then they need no longer be recorded, and kids can write the outcomes directly.

```
ToSingleDigitsTableH[9, 12, Differences → False]
```

	10^2	10^1	10^0
1	0	9	12
2	0	10	2
3	1	0	2

```
? ToSingleDigitsTableH
```

ToSingleDigitsTableH[n__?IntegerHQ] is ToSingleDigitsTable[n] with display of –x replaced by x H
ToSingleDigitsTableH[{n__?IntegerHQ}] works also

9.5. Tables for addition and subtraction using *H*

9.5.1. AdditionTableH

The following merely replaces -10 by 10 *H* for the internal steps.

AdditionTableH[35, 18]

	10^1	10^0
1	3	5
2	1	8
3	+	+
4	4	13
5	1	10 H
6	+	+
7	5	3

It is a simple alteration, but it can be quite useful. Observe that the table uses these identities:

- In the 10^0 column: 13 + 10 H = 3 + 10 + 10 H = 3, using the rule that $a + a\,H = 0$.
- 1n row 5: 10 + 10 H = 0.

Thus:

- If kids had been introduced to AdditionTable with the non-default option Differences → False, then they now have a stepping stone still without the minus sign, so that they can see how the place value system works. The hidden subtraction is brought out into the open as an addition with *H,* and there is the advantage that only 2 sequential steps per column are required (instead of 3 simultaneously).
- If kids had been introduced to AdditionTable with the default option Differences → True, but they were still troubled by the subtraction (in a table of addition), then they now have a format without the minus sign, and 13 + 10 H = 3 + 10 + 10 H = 3, using the rule that $a + a\,H = 0$.

? AdditionTableH

AdditionTableH[n__Integer] is AdditionTable[n] with display of –10 replaced by 10 H
AdditionTableH[{n__Integer}] works also

9.5.2. MixedAdditionTableH

It may need explaining that 18 H = (10 + 8) H = 10 H + 8 H.

MixedAdditionTableH[30, 9 H, 8, 18 H]

	10^1	10^0
1	3	0
2	0	9 H
3	0	8
4	H	8 H
5	+	+
6	2	9 H
7	H	10
8	+	+
9	1	1

```
? MixedAdditionTableH
```

MixedAdditionTableH[n__?IntegerHQ] is MixedAdditionTable[n] with display of –x replaced by x H
MixedAdditionTableH[{n__?IntegerHQ}] works aloso

10. Stage 5. Grade 3. Extension with $a + H\, b = H\, c < 0$

10.1. Algebraic, using HToZero

Simplify.

```
HoldForm[3 + 4 H + H + 6 + 5 H]
```

$3 + 4\,H + H + 6 + 5\,H$

```
above // HToZero
```

$$\begin{pmatrix} 3 + 6 + H + 4\,H + 5\,H \\ 9 + 10\,H \\ 9 + H + 9\,H \\ H \end{pmatrix}$$

The importance here is that the outcome can also be a negative number. Kids have been using minus only for $a - b \geq 0$, and $H\,x$ is the only expression they have for a negative outcome.

10.2. Rather MixedAdditionTableH than SubtractionTableH

The SubtractionTableH has a row with the “-“ sign and thus we rather use the following.

```
MixedAdditionTableH[35, 18 H, 7 H , 12 H]
```

	10^1	10^0
1	3	5
2	H	$8\,H$
3	0	$7\,H$
4	H	$2\,H$
5	+	+
6	1	$12\,H$
7	H	10
8	+	+
9	0	$2\,H$

11. Stage 6. Grade 3. Writing -c as alternative to $H\,c$

11.1. Introducing the minus sign for a negative number, extending $a - b$

The principles are easy to show by means of vector addition and subtraction, but the following concerns notation.

Given:
(1) Kids have been using minus only for $a - b \geq 0$.

(2) Negative numbers have been introduced by $H x$, as the mirror of the positive numbers, and $x + H x = 0$ (with $x > 0$).
(3) Kids already have seen outcomes like $5 + 7 H = 2 H$.

Call into memory that $5 - 3 = 5 + 3 H = 2$ and more cases, so that $y - x = y + H x$, but only for nonnegative outcomes.

Explain that this lesson is not about something new, but only about writing in a shorter way what they already know. Show that it is shorter indeed.

The news 1: $5 - 7 = 5 + 7 H$. Thus $5 - 7$ is a shorter way of writing the same.

The news 2: Introduce $-x$. The definition: $-x = H x$.

The news from 1 and 2: $5 - 7 = 5 + 7 H = 2 H = -2$.

```
Range[-10, 10]
```

{−10, −9, −8, −7, −6, −5, −4, −3, −2, −1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

The definition: $y + H x = y + (-x) = y - x$.

Thus also $5 - 7 = 5 - 5 - 2 = -2$ because $5 - 5 = 0$.

Discussion: Discuss the meaning for the single number. Clarify that it is a *shorter notation only,* but that it will be used to extend $a - b$ also for outcomes lower or less than 0. (Do not say “smaller”, for that applies to the absolute values.)

Let pupils check what the new notation means in terms what they already know.

```
-b        // TimesToH
```

$b H$

Discuss the meaning for the operation on two numbers. (Arrow 10 + Arrow 2 H.)

```
a - b     // TimesToH
```

$a + b H$

Thus the same minus sign is used both for a single number and for the operation on two numbers.

When a pupil says that “this is not simpler at all” then explain that this is how the numbers are used by grown-ups, and that they are supposed to learn this method too.

11.2. New application of the subtraction table

The use of the minus sign in the row with “-“ is used in Grade 1 when outcomes are nonnegative. It can be used again now when pupils are making the transition $-x = H x$.

This is how it looks when negative numbers arise.

```
SubtractionTable[35, 18]
```

	10^1	10^0
1	3	5
2	1	8
3	–	–
4	2	−3
5	−1	10
6	+	+
7	1	7

Kids wanting to check, could use this, but row 3 still contains “-“.

```
SubtractionTableH[35, 18]
```

	10^1	10^0
1	3	5
2	1	8
3	–	–
4	2	3 *H*
5	*H*	10
6	+	+
7	1	7

Colignatus (2018b) clarifies that there can be a switch in the subtraction strategy when the outcome is negative.

```
SubtractionTable[18, 35]
```

... SubtractionTable: Order switched into 35 – 18

	10^1	10^0
1	3	5
2	1	8
3	–	–
4	2	−3
5	−1	10
6	+	+
7	1	7

```
SubtractionTable[11, 99, 78]
```

	10^2	10^1	10^0
1	0	1	1
2	0	9	9
3	0	7	8
4	–	–	–
5	0	−15	−16
6		−1	10
7	−1	10	
8	+	+	+
9	−1	−6	−6

```
? SubtractionTableH
```

SubtractionTableH[n__Integer] is SubtractionTable[n] with display of –10 replaced by 10 H
SubtractionTableH[{n__Integer}] works also

11.3. The tricky *a* - (-*b*)

Mathematica has a difference between number -4 and expression -*b*.

11.3.1. Creating an expression with *H H* for a number

Discuss the tricky case. It is the price paid for above easier notation. (Some kids might say that it is not easier at all though.)

The problematic expression can now be tackled by kids in steps. We have to do a lot of work to first keep the expression in HoldForm and then unpack it in steps. For teachers the steps are important, for programmers the different routines.

```
3 - -4 // TimesToHoldFormH
```

$3 + H\,(-4)$

```
above /. RuleMinusSignToH
```

$3 + H\,(4\,H)$

```
above // MinusSignToH  (*eliminate the brackets*)
```

••• ExpressionHQ: Warning: expression already contains H

$3 + 4\,H\,H$

```
above // AlgebraH  (*algebraic rule H H → 1*)
```

7

11.3.2. With HoldForm on a variable

Use the following routine to avoid a square.

```
tricky = HoldForm[a - (-b)]
```

$a - -b$

```
above  // TimesToHoldFormH
```

$a + b\,H\,H$

```
above    // AlgebraH (*algebraic rule H H → 1*)
```

$a + b$

```
RuleAlgebraH[[2]]
```

HoldPattern[$H\,H$] → 1

PM. This keeps a bracket for (-*b*).

```
Inactivate[a - (-b)]
```

$a + -1 * (-1 * b)$

```
above /. RuleAnyToH
```

$a + H * (H * b)$

11.3.3. From minus to *H H* becoming *H*^2

Simple substitution causes the appearance of the square. This might be no problem and even a natural environment to discuss it.

```
a - (-b)    // TimesToH
```

$a + b\,H^2$

```
above    // AlgebraH (*algebraic rule H H → 1*)
```

$a + b$

Without HoldForm, AlgebraH directly used the substitution rules.

```
RuleAlgebraH[[3]]
```

$H^2 \to 1$

11.4. More terms of a mixed variety, using HToZero

There may be an intermediate stage (up to Grade 6) with mixed use.

Simplify.

```
HoldForm[3 + 4 H + H - 6  - 2 H]
```

$3 + 4\,H + H - 6 - 2\,H$

```
above // HToZero
```

$$\begin{pmatrix} -6 + 3 - 2\,H + H + 4\,H \\ -3 + 3\,H \\ -3 - 3\,H + 6\,H \\ 6\,H \end{pmatrix}$$

Before, I hadn't thought about this possibility. At first the above comes across as a curious mixture of *H* and minus, and thus as rather Byzantine. However, the algebraic rules for a normal variable *x* apply also for *H.* Thus much of the deduction will already be an exercise for such. Of course, *H* comes with additional rules.

11.5. Tables of addition and subtraction without *H*

See Colignatus, Th. (2018b), *Tables for addition and subtraction with better use of the place value system.* The package there doesn't use *H.* It is included in the above however too.

The discussion of addition and subtraction finds a natural end here. We would pick it up for real numbers again though.

12. Stage 7. Grade 4, 5, 6. Exponent *H:* multiplicative inverse, rational number, mixed number

12.1. The US Common Core

In terms of abstraction, it seems to me (but I am not competent in this) that it might be quite feasible that kids have digested the negative numbers by Grade 3, including the basic algebra involved in *H,* so that Grade 4, 5 and 6 are available for the exponent *H.*

The Common Core for Grade 4 e.g. uses the confused notation $5\,\frac{2}{3}$. We can save a lot on this, both the crooked structure and the later required *unlearning* on the notation of multiplication.

```
CCSSMathContent["Grade 4", "NF", "B.3"]
```

Build fractions from unit fractions.
CCSS.Math.Content.4.NF.B.3
Understand a fraction a/b with a > 1 as a sum of fractions 1/b.

```
CCSSMathContent["Grade 4", "NF", "B.3.A"]
```

CCSS.Math.Content.4.NF.B.3.A
Understand addition and subtraction of fractions as joining and separating parts referring to the same whole.

```
CCSSMathContent["Grade 4", "NF", "B.3.B"]
```

CCSS.Math.Content.4.NF.B.3.B
Decompose a fraction into a sum of fractions with the same denominator in more than one way, recording each decomposition by an equation. Justify decompositions, e.g., by using a visual fraction model. Examples: 3/8 = 1/8 + 1/8 + 1/8 ; 3/8 = 1/8 + 2/8 ; 2 1/8 = 1 + 1 + 1/8 = 8/8 + 8/8 + 1/8.

```
CCSSMathContent["Grade 4", "NF", "B.3.C"]
```

CCSS.Math.Content.4.NF.B.3.c
Add and subtract mixed numbers with like denominators, e.g., by replacing each mixed number with an equivalent fraction, and/or by using properties of operations and the relationship between addition and subtraction.

Grade 8 has (http://www.corestandards.org/Math/Content/8/EE/):

```
CCSSMathContent["Grade 8", "EE", "A.1"]
```

Expressions and Equations Work with radicals and integer exponents.
CCSS.Math.Content.8.EE.A.1
Know and apply the properties of integer exponents to generate equivalent numerical expressions.For example, 3^2 × 3^−5 = 3^−3 = 1/3^3 = 1/27

In the alternative approach kids would know use 3^H. Having learned more about exponents in Grade 7 an 8, they would be able to find that 3^2 × 3^-5 = 3^-3 = 3^(3*H*) = (3^3)^H = 27^*H* in accuracy without any need of the fraction bar.

12.2. The multiplicative inverse

The extension of the integers with the inverses of the integers sets us upon the path to create the rational numbers.

- Every nonzero integer *n* has an inverse number n^H such that their grouping is 1.
- The inverse number n^H can be pronounced as "per *n*". We better do not pronounce this as "one per *n*", since in n^H there is no mention of 1. Thus 2^H = "per two" = half, and 4^H = "per 4" = quarter.
- Show that $H^H = H$. It is easy to show that the latter is sufficient, for multiply both sides by *H*. Yet, now show that it is also necessary.
- We better do not use the words third, fourth, etcetera for “fractions”, because these are rank numbers.

Cake division is less relevant than the number line. Yet, a cake divided in 5 parts of equal size would cause that each part has the size 5^H (“per 5”), with the unit of measurement given by the cake. Thus 5 of those parts give 1 cake again.

2^H on the number line follows from:

```
2^H == 2^H × 10 × 10^H == 2^H × 2 × 5 × 10^H == 5 × 10^H
```

In general n^H can be found on the number line by above method or creating the decimal expansion, and always use this rule:

$n\, n^H = 1$

```
above // AlgebraH
```

True

```
RuleAlgebraH[[6]]
```

HoldPattern[x_ x_^H] :→ 1

```
?? RuleAlgebraH
```

RuleAlgebraH gives algebraic rules that tend to keep H. Used in AlgebraH[expr]

```
Attributes[RuleAlgebraH] = {Protected}

RuleAlgebraH = {1 + H → 0, HoldPattern[H H] → 1, H^2 → 1, H^H → H,
  0^H → Indeterminate, HoldPattern[x_ x_^H] :→ 1, x_^(1+H) :→ 1, (x_^H)^H :→ x, (x_ y__)^H :→ y^H x^H}
```

12.3. The meaning of these subsections

The following subsections might seems somewhat tortuous. Obviously we have quick solutions but we are interested now in what routine could mimic which step that a student could take. The following is obvious but mimicking it is not quite simple, e.g. when *Mathematica* wants to simplify into powers of 2. The routine SimplifyH provides some stability.

```
1/2 + 1/4   // SimplifyH
```

3×4^H

12.4. The inverse element with ReleaseHold

At some point pupils will learn about addition of exponents.

$n\, n^H = 1$

$n^{H+1} = 1$

```
above // AlgebraH
```

True

12.5. Prove that 2^H == 2 4^H using multiplication

Prove:

$2^H = 2 \times 4^H$

Proof:

Multiply both sides with 2:

$2 \times 2^H = 2 \times 2 \times 4^H$

$2 \times 2^H = 4 \times 4^H$

$1 = 1$

12.6. Rational numbers

When *n* is a natural number excluding zero then *n^H* are the *basic* rational numbers. Integer multiples of basic rational numbers would be rational numbers too. The natural numbers (likely including 0, depending upon the application) thus are rational numbers too.

$y\, x^H$

Formally we distinguish these *rational numbers* from the *quotient form* with numerator and denominator. Simplification of these rational numbers would tend to involve the prime numbers. Simplification with our routines is not so easy, but there is always SimplifyH.

4×2^H

PM. Using 4 = 2 2 gets interpreted as $4 = 2^2$.

```
above // ReleaseHold
```

2^{H+2}

Use H + 1 = 0 but this requires algebra on the exponent.

```
above // SimplifyH
```

2

For addition and *H* we had a routine HToZero that decomposed expressions into forms $a + a\,H = 0$. Now we might create a routine HToOne that factors integers into the prime numbers, and then applies $a\, a^H = 1$, to avoid such exponents. This leads too far now.

12.7. Mixed numbers

The rational numbers thus are: (i) integers, (ii) the basic rational numbers $0 < x < 1$, (iii) the negative basic rational numbers, and (iv) the remainder are the mixed numbers: sums of the first three.

An expression like 5/2 better is written as a mixed number with *H,* so that it is clearer where it is on the number line. We do not call SortH for the *standard form* because this would put the expression into HoldForm.

```
proper = 5 / 2    // ToH
```

$2^H + 2$

We already discussed this with some repetition. The comment in this subsection is that further arithmetic with these terms causes higher values of the exponents. This makes sense, given above comment on simplification.

The use of the fraction bar causes more complexities, while we better teach students how to work with exponents since they must do so anyway. It is not relevant now to develop a course in dealing with exponents in *Mathematica* now. But a little bit can be mentioned. Let us square and expand the above. This immediately gives higher values for the exponents. The resulting first term might better be 4^*H.*

```
above^2 // Expand // PowerExpand
```

$2^{2H} + 2^{H+2} + 4$

We can always convert back, by SimplifyH, FromH (that generates the fraction bar) or to decimal numbers.

```
p2 = proper^2;
{Evaluate[p2 // SimplifyH] // SortH,
 p2 // FromH,
 p2 // NH}
```

$\left\{6 + 4^H, \frac{25}{4}, 6.25\right\}$

PM. The following uses “Inactive” and unfortunately displays with the fraction bar.

```
inactive = 5 / 2 // RationalInactive
```

$2 + \frac{1}{2}$

Unfortunately, the latter doesn’t expand like (a + b)^2.

```
above^2 // Expand // PowerExpand
```

$\left(2 + \frac{1}{2}\right)^2$

```
? RationalInactive
```

RationalInactive[expr] puts all Rational[x, y]
 in expr into Inactive[Plus][IntegerPart[expr], FractionalPart[expr]]

12.8. Multiplication

Multiplication consists of scalar multiples (like the mirroring above) and grouping. A case of “5 of 4” would be a grouping.

Kids would start with $a + H\, b$ rather than with a - b. Thus grouping would start with H as well.

$a + b H$

An example multiplication with this term:

$(a + b) (a + b H)$

Not having squares yet, pupils could write *a a* and *b b* without problem. Over time the use of exponent 2 will be discovered as a neat trick.

$a^2 + a b H + a b + b^2 H$

$a^2 + a b (H + 1) + b^2 H$

```
above // AlgebraH
```

$a^2 + b^2 H$

```
RuleAlgebraH[[1]]
```

$H + 1 \to 0$

A geometric exposition might need some different labels.

https://en.wikipedia.org/wiki/Difference_of_two_squares

12.9. Simplifying expressions with unevaluated H

The following transforms in the mixed number format.

4×3^H

$3^H + 1$

```
above // SortH
```

$1 + 3^H$

This would tend to work always. First remove *H* and then build up again. SimplifyH might be sufficient, without the need to adapt Simplify to include rules on *H*.

```
4 × 3^H // SimplifyH
```

$3^H + 1$

PM. Above expression already contains *H.* ToH has no algorithm on simplification. It stops at a form that has introduced *H.* Remarkably, another round of application can also introduce a minus sign. Some routines like ToH thus give a warning when *H* is already present.

```
4 × 3^H
```

4×3^H

```
above // ToH
```

ExpressionHQ: Warning: expression already contains H

$4 \times 3^{-H^2}$

12.10. Beware of H^H

Correct is:

```
H^H // AlgebraH
```
H

Correct:

$\left(2^H\right)^H$

2

```
(2^H)^H  // AlgebraH
```
2

Correct, but perhaps confusing at first:

2^{H^H}

$\frac{1}{2}$

```
2^H^H  // AlgebraH
```
2^H

12.11. Prove that 2^H == 2 4^H using exponents

Prove.

$2^H = 2 \times 4^H$

Use that 4 = 2 * 2 and that (x y)^*H* = x^*H* y^*H.*

$2^H = 2\left(2^H \times 2^H\right)$

$2^H = 2 \times 2^H \times 2^H$

$2^H = 2^H \times 1$

12.12. Multiples of a Lowest Common Denominator - Direct approach

The following is the direct approach, but doesn't display neatly.

Simplify this.

$2^H + 4^H$

```
above /. 2^H → 2×4^H  (* proven above *)
```
$2^{2H+1} + 4^H$

```
Collect[above, 4^H]  (* Mathematica's hidden steps *)
```
3×2^{2H}

```
above /. 2^(2H) → 4^H
```
3×4^H

12.13. Multiples of a Lowest Common Denominator - Inactive and HoldForm

The following is more involved, merely to display neatly. This shows that it would be possible to make a routine to clarify the steps.

Simplify this.

```
expr = 1 / 2 + 1 / 4    // ToH
  (*traditional teacher setting up a question with software*)
```

$2^H + 4^H$

```
above /. Plus → Inactive[Plus]
```

$2^H + 4^H$

The order of substitution matters.

$2^H + 4^H$

$2\,4^H + 4^H$

$3\,4^H$

```
above // ReleaseHold
```

3×4^H

Appendix F contains some other approaches that are less attractive.

12.14. Prime numbers

Students may now better appreciate the relevance of prime numbers for a simplied rational number.

```
999 / 456 → {FactorInteger[999] , FactorInteger[456]}
```

$$\frac{333}{152} \to \left\{\begin{pmatrix} 3 & 3 \\ 37 & 1 \end{pmatrix}, \begin{pmatrix} 2 & 3 \\ 3 & 1 \\ 19 & 1 \end{pmatrix}\right\}$$

```
HoldForm[ 3^3 × 37^1 (2^3 × 3^1 × 19^1) ^H]
```

$3^3 \times 37^1 \left(2^3 \times 3^1 \times 19^1\right)^H$

```
above // PowerExpand
```

$3^3 \times 37^1 \left(2^{3\,H} \left(3^{1\,H} \times 19^{1\,H}\right)\right)$

```
above // ReleaseHold
```

$37 \times 2^{3\,H} \times 3^{H+3} \times 19^H$

```
above // FromH
```

$$\frac{333}{152}$$

12.15. An exponent with H in the exponent itself

An exponent with *H* in the exponent itself indicates a root. We may be happy to get rid of the root sign (it is mainly useful as the square root for the Pythagorean Theorem in geometry), but, using exponents requires a fine distinction of the vertical distances, and it could be advisable to use ^ in exponents themselves.

```
expr = (5 / 2) ^ (8 / 7)  //  ToH   (* traditional teacher creating an expr *)
```

$(2^H + 2)^{7^H+1}$

We would rather see the following order within the mixed numbers, because of the number line, but this sorting has not been implemented as a general feature.

```
Evaluate[expr] // SortH
```

$(2 + 2^H)^{1+7^H}$

The standard form in *Mathematica*. Do not overlook the 7 in the root sign.

```
above // ReleaseHold //  FromH
```

$$\frac{5\sqrt[7]{\frac{5}{2}}}{2}$$

13. Appendix A. More details on using *Mathematica*

13.1. More details on using *Mathematica*

Mathematica already uses -1 mostly in the way that we want to. However, there are some holes w.r.t. the intended use in education.

Having these various routines may be confusing (at first), but we will employ them for the different purposes of the different stages, and then they might be understood within those contexts.

13.2. Universal constant

Mathematica doesn't use *H* yet as an unevaluated universal constant like *i*, π and *e*. Thus there is need for some additional routines.

13.3. HoldForm versus Inactivate

The MathEd`H` routines use HoldForm (without * and ^) rather than Inactivate (with * and ^). Rules working on the one form may not work on the other form. We are used to the shorter HoldForm and then can use the HoldFirst attribute. (Pupils will need to learn both notations though.)

At first the Inactive form seemed more relevant, since it also allows steps in Activate. Kids might be served with the * and ^ in the Inactive form. In my personal taste I found them distractive, and testing on Inactive[Plus] etcetera a bridge too far. While using HoldForm, the use of Inactive remains useful on occasion. When the makers of *Mathematica* consider implementing *H,* they best look at both HoldForm and Inactivate.

13.4. HoldForm on Plus

Even in HoldForm, 3 - 2 is coded as 3 + -2, and input 3 + (-2) is displayed as 3 - 2. It does read better (and the FullForm may be only a concern for programming).

```
3 - 2 // HoldForm
```

3 – 2

```
above // FullForm
```

HoldForm[Plus[3, −2]]

```
3 + -2 // HoldForm
```

3 – 2

```
above // FullForm
```

HoldForm[Plus[3, −2]]

13.5. Subtraction uses Times

Mathematica deals with subtraction via Times, and correctly so. The best way to show this is by calling Inactivate, though we will tend to use HoldForm.

```
Inactivate[a - (-b)]
```

$a + -1 * (-1 * b)$

With abstract *H* we better see the structure of the expression.

```
above    /.  RuleAnyToH
```

$a + H * (H * b)$

The bracket for (-1 * *b*) is correct, to identify the operation on the single variable. Because of the flatness of the product I prefer *a* + (-1)*(-1)**b* however.

Activation generates a square.

```
above    // Activate
```

$a + b\,H^2$

This might not fit the level of education here, but it might also be considered that introducing squares might actually be done before the introduction of negative numbers.

The routines TimesToH and TimesToHoldFormH only look at the expression Times[-1, *b*]. They use HoldForm. This also allows for mixtures of different formats (numbers with the minus sign and with *H*).

```
a - (-b)            // TimesToHoldFormH
```

$a + b\,H\,H$

```
2 - 3 + 6 - (-b) - 7 // TimesToHoldFormH
```

$2 - 3 + 6 + b\,H\,H - 7$

TimesToH has a ReleaseHold and thus also creates the square.

```
a - (-b)          // TimesToH
```

$a + b\,H^2$

There are occasions when we also want to show the fraction bar. (The latter is best postponed till when kids have good command of division by means of the expression with the exponent.)

```
-2 / (a - -b)   // TimesToH   (* only interested in H b, kids learning about H^2 *)
```

$$-\frac{2}{a + b\,H^2}$$

```
2 - 3 + 6 - (-b) - 7 // TimesToH
```

$b\,H^2 - 2$

MinusSignToH affects all numbers with the minus sign, including power terms.

```
2 - 3 + 6 - (-b) - 7 // MinusSignToH
```

$2 + 3\,H + 6 + b\,H\,H + 7\,H$

ToH only looks at the exponent (and *Mathematica* got rid of *minus negative*).

```
1 / (a - -b)   // ToH     (* learned how to get rid of H H *)
```

$(a + b)^H$

Remember that these routines would be for teachers and researchers to develop exercises and tests, and for pupils to check (with pencil and paper) what the expression with minus means in terms of *H* that they already have learned.

? above

above is the same as % or Out[$Line – 1]. It seems clearer for
education. Most routines in the MathEd`H` package have HoldFirst, but
recognise a func[above] case and then change to func[Evaluate[above]]

```
? RuleAnyToH
```

RuleAnyToH replaces –1 by H, using only –1 –> H. This substitutes
indiscriminately but leaves –2, –3, ... unaffected. See TimesToH and ToH for particular
patterns (not using Inactive), and RuleMinusSignToH for the negative integers

```
? TimesToHoldFormH
```

TimesToHoldFormH is TimesToH with HoldForm
and not ReleaseHold. It transforms a – (–b) into HoldForm[a + b H H]

```
? MinusSignToH
```

MinusSignToH[expr] applies RuleMinusSignToH, has HoldFirst, and leaves the
result into HoldForm. The occurence of H (H n) is turned into n H H instead of n H^2

13.6. Expressions with a fraction bar use Power, though with exceptions

Mathematica deals with expressions with the fraction bar with Power, and rightly so.

```
Inactivate[a / b]
```

$a * b \wedge (-1)$

```
above /. RuleAnyToH
```

$a * b \wedge H$

```
above // Activate
```

$a\, b^H$

However, *Mathematica* (i) has the exception of Rational[*n, m*], and (ii) neglects the mixed number. See **Appendix E** on this.

For mixed numbers, we would like to see this:

```
2 + 2^H
```

$2^H + 2$

Mathematica can also use the fraction bar format *m* / *n* that we want to avoid (also because it obscures the position on the number line).

```
above // FromH
```

$\frac{5}{2}$

```
? FromH
```

FromH[expr] replaces H by –1

Observe that RuleAnyToH replaces indiscriminately, doesn't recognise the integers with the minus sign, and doesn't have HoldForm. In this case it produces the relevant form with *H* only for the exponent (and *Mathematica* took care or *minus negative*).

```
-2 / (a - (-b)) /. RuleAnyToH
```

$-2\,(a + b)^H$

TimesToHoldFormH doesn't recognise the division and numbers with a minus sign, but has *H H* in HoldForm. This routine is suited for learning to work with *H* and subtraction, but not for division and the rational numbers.

```
-2 / (a - (-b)) // TimesToHoldFormH
```

$-\frac{2}{a + b\,H\,H}$

MinusSignToH is most comprehensive in this case, and possibly insightful for kids learning the ropes, but overly zealous w.r.t. the target of using *H* eventually only for the exponent.

```
-2 / (a - (-b)) // MinusSignToH  (*all aspects involved*)
```

$2\,H\,(a + b\,H\,H)^H$

```
above // AlgebraH
```

$2\,H\,(a + b)^H$

13.7. Sorting

Sorting provides another angle to consider.

The canonical order regards *H* as a letter in the alphabet.

For $2 + 2^H$ we rather see first 2 and secondly 2^H because of the position on the number line.

Mathematica has this order, and thus SortH imposes HoldForm.

```
2 + 2^H
```

$2^H + 2$

```
above // SortH
```

$2 + 2^H$

Sorting also affects lists.

```
IntegerLineH[3, 5]
```

$\{3\,H, 2\,H, H, 0, 1, 2, 3, 4, 5\}$

```
above // Sort
```

$\{0, 1, 2, 3, 4, 5, H, 2\,H, 3\,H\}$

We like to see the order of the number line though.

```
above // SortH
```

$\{3\,H, 2\,H, H, 0, 1, 2, 3, 4, 5\}$

```
? SortH
```

SortH[expr] sorts Lists using the values of
FromH, and sorts Plus in canonical order so that we get 2 + 2^H

14. Appendix B. The routines

14.1. Routines

14.1.1. Math`Ed`

```
? Cool`MathEd`H`*
```

▼ Cool`MathEd`H`

above	MinusSignIntegerQ	RuleNegativeToH
AlgebraH	MinusSignToH	RulePowerHeadToH
AlgebraHHelper	MixedNumberH	RulePowerHToMixedNumb-er
BothToH	NH	RulePowerToRational
ExpressionHQ	PowerHeadToH	RuleTimesToH
FlattenTimesH	PowerHToMixedNumber	ShowInversesH
FromH	PowerToH	SimplifyH
H	PowerToRational	SortH
HoldFirstH	RationalInactive	TextReflect
HToZero	RuleAlgebraH	TextRotate
IntegerHQ	RuleAnyToH	TimesToH
IntegerLineH	RuleHToZero	TimesToHoldFormH
IntegerLineHQ	RuleMinusSignToH	ToH

The descriptions of the main routines are (while the others follow along the way):

```
? H
```

H is the universal constant with value –1. Pronounce as "eta" or "symbolic negative one (or unit)". It represents a half turn along a circle, compared to the complex number I that gives a quarter turn. It is unevaluated and used symbolically like in 5 / 2 = 2 + 2^H. For evaluation, see symbolic FromH and numeric NH. See also TimesToH and ToH

```
? FromH
```

FromH[expr] replaces H by –1

```
? ToH
```

ToH[expr] checks whether some elements can be turned into a Rational number with a MixedNumber format, then turns all Power[x, –1] expressions into x^H. It has property HoldFirst, applies the replacement routines to the unevaluated expression, and then has ReleaseHold. ToH works like a "PowerToH" (deliberately not defined here), and internally handles RationalToH and PowerHeadToH

```
? MixedNumberH
```

MixedNumberH[expr] puts all Rational[n, m] or n
Power[m, −1] in expr into IntegerPart[expr] + ToH[FractionalPart[expr]]
MixedNumberH[x_Rational] is a helper routine
MixedNumberH["Example"] gives an example comparison of using H and the fraction bar

```
? NH
```

NH[expr (, n)] is N[FromH[expr] (, n)]

14.1.2. Tables of addition and subtraction without *H*

See Colignatus (2018b) for this.

```
? Cool`MathEd`AdditionTable`*
```

▼ Cool`MathEd`AdditionTable`

AdditionTable	SquareTableOfAddition	ToSingleDigits
BothMixedAdditionTables	SubtractionTable	ToSingleDigitsTable
BothSubtractionTables	TableOfBasicAddition	
MixedAdditionTable	TableOfBasicGrouping	

14.1.3. Tables of addition and subtraction with *H*

These are versions of Colignatus (2018b) now with the use of $H = -1$.

```
? Cool`MathEd`AdditionTableH`*
```

▼ Cool`MathEd`AdditionTableH`

AdditionTableH	MixedAdditionTableH	TableOfBasicAdditionH
BothMixedAdditionTablesH	SubtractionTableH	ToSingleDigitsTableH

14.1.4. Pronunciation using the full place value system

See Colignatus (2018a) for this.

```
? Cool`MathEd`PronounceInteger`*
```

▼ Cool`MathEd`PronounceInteger`

LatestPronunciation	PlaceValueDigits$Language	PronounceInteger
PartialPlaceValue	PlaceValueH	PronounceIntegerReadMe
PlaceValue	PlaceValueIntegerName	PronounceIntegers
PlaceValueBlock	PlaceValueRules	SpeakTransliteration
PlaceValueDigits	PlaceValueTable	

15. Appendix C. Basic properties of the routines

15.1. Testing for routines and answers to questions

For testing in routines, but also testing whether pupils have completed an exercise.

```
? IntegerLineHQ
```

IntegerLineHQ[expr] tests whether expr strictly
would be on the IntegerLineH, thus with expr one of ..., 2 H, H, 0, 1, 2, ...

```
? IntegerHQ
```

IntegerHQ[expr] is IntegerQ[expr] || IntegerLineHQ[expr], thus allowing integers with the minus sign and H, with the leniency of allowing also for –x with symbolic x (assumed to be an integer)

The following helps to identify the default numbers with the minus sign. We cannot call this routine "NegativeIntegerQ" since $H\,x$ is also a negative integer for $x > 0$.

```
? MinusSignIntegerQ
```

MinusSignIntegerQ[x] is IntegerQ[x] && Negative[x]

```
? ExpressionHQ
```

ExpressionHQ[expr] tests whether H occurs in expr

15.2. To and From, focussed on exponents

```
5 / 2       // ToH
```

$2^H + 2$

```
-5 / 2       // ToH
```

$-2^H - 2$

```
above     // FromH
```

$-\frac{5}{2}$

15.3. Holdfirst

Routines have HoldFirst so that e.g. input of 1/2 + 1/4 does not first simplify to 3/4.

```
1 / 2 + 1 / 4     // ToH
```

$2^H + 4^H$

```
above         // FromH
```

$\frac{3}{4}$

```
above      // ToH
```

3×4^H

The routines for *H* tend to recognise the term *above* and then translate *func*[*above*] into *func*[Evaluate[*above*]]. Some also recognise a HoldForm input. In other cases we would have to call Evaluate ourselves.

15.4. TimesToH for Times. ToH for Power and Rational. BothToH for both

TimesToH assumes a format Times[-1, b] while the numbers with the minus sign do not have that format. Thus TimesToH cannot transform those numbers. Use MinusSignToH for this. However, the latter may also transform Power[x, -1] into Power[x, H], which may not be the intention at this moment.

```
(a - b)  // TimesToH
```

$a + b\,H$

```
1 / (a - b)  // TimesToH
```

$\frac{1}{a + b\,H}$

```
1 / (a - b)   // ToH
```

$(a - b)^H$

```
1 / (a - b)  // BothToH
```

$(a + b\,H)^H$

```
? BothToH
```

BothToH[expr] is TimesToH[ToH[expr]], with HoldFirst. The idea is that education proceeds in steps, such that TimesToH is important first, BothToH intermediately, and later only ToH

```
? TimesToH
```

TimesToH[expr] replaces variables –x by H x. Has HoldFirst, applies RuleTimesToH, and then ReleaseHold. For the negative integers use RuleNegativeToH, though this might affect Power[x, –1] too

```
? TimesToHoldFormH
```

TimesToHoldFormH is TimesToH with HoldForm and not ReleaseHold. It transforms a – (–b) into HoldForm[a + b H H]

15.5. Numbers with the minus sign and MinusSignToH

In the above, we looked mostly at variables but now we look at numbers specifically.

Mathematica has the numbers with the minus sign in its hard core programming.

```
-3 // FullForm
```

-3

The MathEd`H` routines have the following assumptions:

- At a first stage, pupils work only with nonnegative numbers, and thus there are no numbers with a minus sign in them. Then we have only minus in *a* - *b*. Variablles differ from numbers though. **TimesToH** suffices for immediate ReleaseHold, or use MinusSignToH to keep HoldForm.

```
a - b // TimesToH  (*variables*)
```

$a + b\,H$

```
4 - 3 // TimesToH  (*numbers*)
```

1

```
4 - 3  // MinusSignToH (*HoldForm, useful for more numbers*)
```

$4 + 3\,H$

This is useful for the first stage.

```
(2 - 5 + 6 - 7)  // MinusSignToH  (*traditional teacher designing a question*)
```

$2 + 5\,H + 6 + 7\,H$

```
above // ReleaseHold
```

$12\,H + 8$

This only puts the expression into HoldForm. For a first stage, this is useless. It however would be useful at the stage when both *H* and the numbers with the minus sign are being used.

```
(2 - 5 + 6 - 7)  // TimesToHoldFormH
```

$2 - 5 + 6 - 7$

At the first stage we would not use the following, for it doesn't change the numbers with the minus sign and it has releasehold. The reason that it didn't evaluate above is because of the variables *a* and *b*.

```
(2 - 5 + 6 - 7)  // TimesToH
```

-4

This is already more interesting. Observe that the routine does not distinguish between the type of integer. The number -4 is not the same as Times[-1, 4].

```
(8 - 1 + 3 - -4)  // TimesToHoldFormH
```

$8 - 1 + 3 + H\,(-4)$

This recognises -4 too but evaluates, turning the negative number into a binary operation.

```
(8 - 1 + 3 - -4) // TimesToH
```

$10 - 4\,H$

- At a second stage there might be numbers with the minus sign, and attention is moving towards using *H* only for the exponent. In that case **TimesToH** respects also numbers with the minus sign while MinusSignToH would be overkill.

16. Appendix D. SimplifyH

Thes routines in **Appendix A** can do a little only. Our few basic algebraic rules do not include all possible variation. The following introduces a single obstructive variable. Not all our routines find a neat solution.

```
expr = a (2 H + 3)
```

$a\,(2\,H+3)$

```
above  /. RuleHToZero
```

a

```
Evaluate[expr]   // HToZero
```

$$\begin{pmatrix} a\,(3+2\,H) \\ a\,(1+2+2\,H) \\ a \end{pmatrix}$$

AlgebraH is intended to contain only the main rules. It is not intended to store a multitude of variations that will always find a solution.

```
Evaluate[expr]   // AlgebraH
```

$a\,(2\,H+3)$

If we would get stuck (or would not find the proper substitution routine) there is always the straight-forward SimplifyH.

```
Evaluate[expr]   // SimplifyH
```

a

```
? SimplifyH
```

SimplifyH[expr, f:ToH] is f[Evaluate[Simplify[FromH[expr]]]]

When we use variables or HoldForm then the numbers do not multiply, and we can do algebra.

```
(a b) ^H   /. RuleAlgebraH
```

$a^H\,b^H$

```
HoldForm[(3 × 4) ^H]
```

$(3\times 4)^H$

```
above /. RuleAlgebraH
```

$4^H\times 3^H$

```
?? RuleAlgebraH
```

RuleAlgebraH gives algebraic rules that tend to keep H. Used in AlgebraH[expr]

```
Attributes[RuleAlgebraH] = {Protected}
```

```
RuleAlgebraH = {1 + H → 0, HoldPattern[H H] → 1, H^2 → 1, H^H → H,
  0^H → Indeterminate, HoldPattern[x_ x_^H] :→ 1, x_^(1+H) :→ 1, (x_^H)^H :→ x, (x_ y__)^H :→ y^H x^H}
```

17. Appendix E. *Mathematica*: Rational[*n, m*] and *n* Power[*m*, -1]

17.1. How ToH deals with this

For compexer input, we meet with some particular properties.

Rational[*n, m*] is the form of the Rationals, but the standard form often is *n* Power[*m*, -1].

The *n* Power[*m,* -1] format seems standard, also in input, but sometimes *Mathematica* creates the Rational format in output. Calling FullForm for input is not enough, and we may have to use Hold.

```
FullForm[5 / 2]
```

Rational[5, 2]

```
FullForm[Hold[5 / 2]]
```

Hold[Times[5, Power[2, −1]]]

```
FullForm[2 + 1 / 2]
```

Rational[5, 2]

```
FullForm[Hold[2 + 1 / 2]]
```

Hold[Plus[2, Times[1, Power[2, −1]]]]

This makes programming a bit awkward.

- For Rational[5, 2] it is easy to transform this into 2 + 2^H, with no need of HoldForm.
- For Times[5, Power[2, -1], ...] it tends to be more difficult when there are other factors.

```
case = 2 / (5 x)
```

$$\frac{2}{5\,x}$$

```
case    // FullForm
```

Times[Rational[2, 5], Power[*x*, −1]]

For us this becomes (do not forget that ToH has attribute HoldFirst, so that Evaluate is needed):

```
case    // Evaluate // ToH
```

$2 \times 5^H\, x^H$

17.2. Subroutines

ToH calls these (sub-) routines. Deliberately called PowerHeadToH instead of PowerToH.

```
(case // Evaluate) // PowerHeadToH
```

$$\frac{2\,x^H}{5}$$

```
case // Evaluate // MixedNumberH
```

$$\frac{2\times 5^H}{x}$$

17.3. RuleAnyToH, ToH, TimesToH, BothToH

RuleAnyToH replaces -1 by *H* indiscriminately. We might say that it provides a literal translation. In the following example, the 1 remains standing, and 5 4^H is not developed into a proper mixed number. One might say that these steps belong to the realm of "simplification". One might also say that it forms an important element in translation to properly deal with this. The 1 prefex and the hidden whole in 5/4 are products of an inferior notation only.

```
HoldForm[1 / (2 + 5 / 4)] /. RuleAnyToH
```

$$1\left(2+5\times 4^H\right)^H$$

```
above // AlgebraH
```

$$\left(5\times 4^H+2\right)^H$$

```
above // SortH
```

$$\left(2+5\times 4^H\right)^H$$

ToH is more sophisticated. Writing 5 / 4 as 1 + 4^H also causes the addition of the wholes 2 + 1 = 3.

```
HoldForm[1 / (2 + 5 / 4)] // ToH
```

$$\left(4^H+3\right)^H$$

TimesToH has no business here, and applies ReleaseHold.

```
HoldForm[1 / (2 + 5 / 4)] // TimesToH
```

$$\frac{4}{13}$$

The following applies first ToH and then TimesToH, but in this case ToH takes the brunt.

```
HoldForm[1 / (2 + 5 / 4)] // BothToH
```

$$\left(4^H+3\right)^H$$

17.4. Seemingly not 100% obeying HoldFirst

Above expression 1 / (2 + 5/4) may appear also in a larger expression.

The following uses ToH and thus transforms 2 + 5/4 into 2 + 1 + 4^H and this becomes 3 + 4^H.

```
(2 + 1 / 2 ) / (2 + 5 / 4) x^-2  // ToH
```

$(2^H + 2)(4^H + 3)^H x^{2H}$

ToH has attribute HoldFirst. Not having this would cause a more drastic change.

```
(2 + 1 / 2 ) / (2 + 5 / 4) x^-2
```

$$\frac{10}{13\,x^2}$$

```
above  // ToH
```

$10 \times 13^H x^{2H}$

18. Appendix F. Lowest Common Denominator

18.1. Comment

This concerns adding 1/2 + 1/4. This is simpler with H than using the traditional fraction bar. The current question is how we can reproduce that simplicity within the environment of *Mathematica.* The best approach is in the body of the text, and the following are some alternatives.

Obviously we have a quick solution but we are interested now in what routine could mimic which step that a student could take. *Mathematica* complicates the steps by creating powers of 2.

```
1 / 2 + 1 / 4   // SimplifyH
```

3×4^H

18.2. Multiples of a LCD - with a variable

The use of a variable is simplest, but variables are not (much) used in current curriculum.

Simplify this.

```
expr = 1 / 2 + 1 / 4   // ToH
  (*traditional teacher setting up a question with software*)
```

$2^H + 4^H$

```
subst = 4^H → c  (*commonfactor*)
```

$4^H \to c$

```
expr /. Plus → Inactive[Plus]
```

$2^H + 4^H$

```
above  /. subst
```

$2^H + c$

```
above /.  2^H → 2 c
```

$2\,c + c$

```
above // Activate
```

$3\,c$

```
above /. Reverse[subst]
```

3×4^{H}

18.3. Multiples of a LCD - without HoldForm and Activate

Simplify this.

```
1 / 2 + 1 / 4    // ToH  (* traditional teacher setting up a question with software *)
```

$2^{H}+4^{H}$

```
Collect[above, 4^H]  (* remarkably does not work, but changes question *)
```

$2^{H}+2^{2\,H}$

```
Collect[above, 2^H] (* remarkably does not work *)
```

$2^{H}+2^{2\,H}$

```
above /. 4^H → (2^H)^2 /. 2^H → 2 × 4^H  (* NB: 2 H + 1 == H *)
```

$2^{2\,H}+2^{2\,H+1}$

```
Collect[above, 2^H]  (* remarkably works *)
```

$3\times 2^{2\,H}$

```
above /. 2^(2 H) → 4^H
```

3×4^{H}

18.4. Multiples of a LCD - with HoldForm

Simplify this.

```
1 / 2 + 1 / 4    // ToH  (* traditional teacher setting up a question with software *)
```

$2^{H}+4^{H}$

```
above /. 2^H → HoldForm[2 × 4^H]
```

$2\times 4^{H}+4^{H}$

```
above // ReleaseHold
```

$2^{2\,H+1}+4^{H}$

```
Collect[above, 4^H]
```

$3\times 2^{2\,H}$

```
above /. 2^(2 H) → 4^H
```

3×4^{H}

19. Appendix G. Terminology of mathematics by computer

Mathematics concerns patterns and can involve anything, so that we need flexibility in our tools when we do or use mathematics. In the dawn of mankind we used stories. When writing was invented we used pen and paper. It is a revolution for mankind, comparable to the invention of the wheel and the alphabet, that we now can do mathematics using a computer. Many people focus on the computer and would say that it is a computer revolution, but computers might also generate chaos, which shows that the true relevance comes from structured use.

I regard mathematics by computer as a two-sided coin, that involves both human thought (supported by tools) and what technically happens within a computer. The computer language (software) is the interface between the human mind and the hardware with the flow of electrons, photons or whatever (I am no physicist). We might hold that thought is more fundamental, but this is of little consequence, since we still need consistency that 1+1 = 2 in math also is 1+1 = 2 in the computer, and properly interfaced by the language that would have 1+1 = 2 too. The clearest expression of mathematics by computer is in "computer algebra" languages, that understand what this revolution for mankind is about, and which were developed for the explicit support of doing mathematics by computer.

The makers of *Mathematica* (WRI) might be conceptually moving to regarding computation itself as a more fundamental notion than mathematics or the recognition and handling of patterns. Perhaps in their view there would be no such two-sided coin. The brain might be just computation, the computer would obviously be computation, and the language is only a translator of such computations. The idea that we are mainly interested in the structured products of the brain could be less relevant.

Stephen Wolfram by origin is a physicist and the name "*Mathematica*" comes from Newton's book and not from "mathematics" itself, though Newton made that reference. Stephen Wolfram obviously has a long involvement with cellular automata, culminating in his *New Kind of Science*. Wolfram (2013) distinguishes *Mathematica* as a computer program from the *language* that the program uses and is partially written in. Eventually he settled for the term "Wolfram language" for the computer language that he and WRI use, like "English" is the language used by the people in England (codified by their committees on the use of the English language).

My inclination however was to regard “*Mathematica”* primarily as the name of the language that happened to be evaluated by the program of the same name. I compared *Mathematica* to Algol and Fortran. I found Wolfram’s Addison-Wesley book title in 1991 & 1998 “*Mathematica*. A system for doing mathematics by computers” as quite apt. Obviously the *system* consists of the language and the software that runs it, but the latter might be provided by other providers too, like Fortran has different compilers. Every programmer knows that the devil is in the details, and that a language documentation on paper might not give the full details of actually running the software. Thus when there are not more software providers then it is only accurate to state the the present definition of the language is given precisely by the one program that runs it. This is only practical and not fundamental. In this situation there is no conflict in thinking of “*Mathematica* as the language of *Mathematica*”. Thus in my view there is no need to find a new name for the language. I thought that I was using a language but apparently in Wolfram's recent view the emphasis was on the computer program. I didn't read Wolfram's blog in 2013 and otherwise might have given this feedback.

Wolfram (2017) and (2018) uses the terms "computational essay" and "computational thinking" while the latter is used such that he apparently intends this to mean something like (my interpretation): programming in the Wolfram Language, using internet resources, e.g. the cloud and not necessarily the stand-alone version of *Mathematica* or now also Wolfram Desktop. My impression is that Wolfram indeed emphasizes *computation,* and that he perhaps also wants to get rid of a popular confusion of the name *"Mathematica"* with *mathematics* only. Apparently he doesn't want to get rid of that name altogether, likely given his involvement in its history and also its fine reputation.

A related website is https://www.computerbasedmath.org (CBM) by Conrad Wolfram. Most likely Conrad adopts Stephen's view on computation. It might also be that CBM finds the name "*Mathematica"* disinformative, as educators (i) may be unaware of what this language and program is, (ii) may associate mathematics with pen and paper, and (iii) would pay attention however at the word "computer". Perhaps CBM also thinks: You better adopt the language of your audience than teach them to understand your terminology on the history of *Mathematica.*

I am not convinced by these recent developments. I still think: (1) that this is a two-sided coin (but I am no physicist and do no know about electrons and such), (2) that it is advantageous to clarify to the world: (2a) that mathematics can be used for everything, and (2b) that doing mathematics by computer is a revolution for mankind, and (3) that one should beware of people without didactic training who want to ship computer technology into the classroom. My suggestion to Stephen Wolfram remains, as I did before in (2009, 2015a), that he turns WRI into a public utility like those that exist in Holland - while it already has many characteristics of this. It is curious to see the open source initiatives that apparently will not use the language of *Mathematica,* now by WRI (also) called the Wolfram Language, most likely because of copyright fears even while it is good mathematics.

Apparently there are legal concerns (but I am no lawyer) that issues like 1+1 = 2 or π are not under copyright, but that choices for software can be. For example the use of $h[x]$ with square brackets rather than parentheses $h(x)$, might be presented to the copyright courts as a copyright issue. This is awkward, because it is good didactics of mathematics to use the square brackets. Not only computers but also kids may get confused by expressions $a(2 + b)$ and $f(x + h) - f(x)$. Let me refer to my suggestion that each nation sets up its own National Center for Mathematics Education. Presently we have a jungle that is no good for WRI, no good for the open source movement (e.g. R or https://www.python.org or http://jupyter.org), and especially no good for the students. Everyone will be served by clear distinctions between (i) what is in the common domain for mathematics and education of mathematics (the language) and (ii) what would be subject to private property laws (programs in that language, interpreters and compilers for the language) (though such could also be placed into the common domain).

20. Literature

Thomas Colignatus is the scientific name of Thomas Cool, econometrician and teacher of mathematics, Scheveningen, Holland.

The author thanks Peter Harremöes (Denmark) for a discussion that helped to focus on this present proposal, and *H* may also be called the *Harremoës operator.*

Bellamy, A. (2015), *A Critical Analysis of Teaching and Learning Negative Numbers*, Philosophy of Mathematics Education Journal, Issue 29, http://socialsciences.exeter.ac.uk/education/research/-centres/stem/publications/pmej/pome29/index.html

Colignatus, Th. (2009, 2015a), *Elegance with Substance,*
(1) website: http://thomascool.eu/Papers/Math/Index.html
(2) PDF on Zenodo: https://zenodo.org/record/291974
(3) Publisher for a print: https://www.mijnbestseller.nl/shop/index.php/catalog/product/view/id/115148/s/elegance-with-substance-78023-www-mijnbestseller-nl/

Colignatus, Th. (2014a), *An Archi gif, compliments to Lucas V. Barbosa,* https://boycottholland.wordpress.com/2014/07/14/an-archi-gif-compliments-to-lucas-v-barbosa/

Colignatus, Th. (2014b), *Confusing math in elementary school*, https://boycottholland.wordpress.com/2014/08/25/confusing-math-in-elementary-school/

Colignatus, Th. (2014c), *Taking a loss*, https://boycottholland.wordpress.com/2014/08/30/taking-a-loss/

Colignatus, Th. (2014d), *With your undivided attention*, https://boycottholland.wordpress.com/2014/09/04/with-your-undivided-attention/

Colignatus, Th. (2015b), *A child wants nice and no mean numbers*,
(1) website: http://thomascool.eu/Papers/NiceNumbers/Index.html
(2) PDF on Zenodo: https://zenodo.org/record/291979
(3) Publisher for a print: https://www.mijnbestseller.nl/shop/index.php/catalog/product/view/id/118082/s/a-child-wants-nice-and-no-mean-numbers-79074-www-mijnbestseller-nl/

Colignatus, Th. (2015c), *Computer algebra is a revolution. But 21st century skills ?*, https://boycottholland.wordpress.com/2015/12/08/computer-algebra-is-a-revolution-but-21st-century-skills-q/

Colignatus, Th. (2015d), *Jan van de Craats tortures kids with fractions*, https://boycottholland.wordpress.com/2015/09/19/jan-van-de-craats-tortures-kids-with-fractions/

Colignatus, Th. (2015e), *Abstraction vs Eugene Wigner & Edward Frenkel,* https://boycottholland.wordpress.com/2015/05/23/abstraction-vs-eugene-wigner-edward-frenkel/

Colignatus, Th. (2015f, 2018c), *The need for a standard for the mathematical pronunciation of the natural numbers. Suggested principles of design. Implementation for English, German, French, Dutch and Danish*, https://zenodo.org/record/774866

Colignatus, Th. (2016), *Looking beyond the Common Core State Standards for Mathematics,* https://boycottholland.wordpress.com/2016/03/22/looking-beyond-the-ccss-m/

Colignatus, Th. (2016b), *What is the use of group theory ?,* https://boycottholland.wordpress.com/2016/12/29/what-is-the-use-of-group-theory-q/

Colignatus, Th. (2017a), *Education, group theory and division,* https://boycottholland.wordpress.-com/2017/01/03/education-group-theory-and-division/

Colignatus, Th. (2017b), *Teaching Simpson's paradox at elementary school – with H,* https://boy-cottholland.wordpress.com/2017/03/05/teaching-simpsons-paradox-at-elementary-school-with-h/

Colignatus, Th. (2017c), *Ook in België is de gemengde breuk een drama,* https://boycotholland.word-press.com/2017/06/13/ook-in-belgie-is-de-gemengde-breuk-een-drama/

Colignatus, Th. (2018a), *Pronunciation of the integers with full use of the place value system,* http-s://www.wolframcloud.com/objects/thomas-cool/MathEd/2018-05-09-Pronunciation-of-inte-gers.nb, or http://community.wolfram.com/groups/-/m/t/1334793, or PDF at https://-doi.org/10.5281/zenodo.1244063 or Mathematica notebook with package at https://-doi.org/10.5281/zenodo.1244008

Colignatus, Th. (2018b), *Tables for addition and subtraction with better use of the place value system*, update May 5 2018, https://www.wolframcloud.com/objects/thomas-cool/MathEd/2018-04-02-AdditionTable-and-SubtractionTable.nb or http://community.wolfram.com/groups/-/m/t/1313408 or PDF at https://zenodo.org/record/1241349 and Mathematica notebook with package at http-s://zenodo.org/record/1241404

Common Core State Standards (CCSS) (2018), "Mathematics Standards", http://www.corestandard-s.org/Math/

Everyday Mathematics (2018), *Partial quotients*, http://everydaymath.uchicago.edu/teaching-topics/computation/div-part-quot/

Harremoës, P. (2000), "Talnotation". May, MFK-bladet 22.5, http://www.harremoes.dk/Peter/, http://www.harremoes.dk/Peter/talnot.pdf

Hiele, P.M. van (1973), *Begrip en Inzicht,* Muusses

Hiele, P.M, van (1986), *Structure and Insight: A Theory of Mathematics Education,* Academic Press

Ministry of Education, Culture and Science (2016): "VO-docenten bevoegd voor basisschool: 'goede methode voor talentvolle kinderen'", https://www.delerarenagenda.nl/blog/weblog/we-blog/2016/20160902-vo-docenten-bevoegd-voor-basisschool-%E2%80%98goede-methode-voor-talentvolle-kinderen%E2%80%99

Pillsy (2010), *Contribution to: "Evaluate beyond one level within Hold in Mathematica"*, https://stack-overflow.com/questions/3136604/evaluate-beyond-one-level-within-hold-in-mathematica

Skemp, R. (1975), *The psychology of learning mathematics,* Penguin

Tall, D.O (2013), *How humans learn to think mathematically*, Cambridge

Vredenduin, P.G.J. (1991), *De geschiedenis van positief en negatief,* edited by Peter van 't Riet, Wolters-Noordhoff

Wolfram Research, Inc. (2017), *Mathematica*, Version 11.2, http://www.wolfram.com

Wolfram, S. (1991, 1998), *Mathematica. A system for doing mathematics by computer,* 2nd edition, Addison-Wesley

Wolfram, S. (2013), *What Should We Call the Language of Mathematica?*, http://blog.stephenwolfram.-com/2013/02/what-should-we-call-the-language-of-mathematica/

Wolfram, S. (2017), *What Is a Computational Essay?,* http://blog.stephenwolfram.com/2017/11/what-is-a-computational-essay/

Wolfram, S. (2018), *Launching the Wolfram Challenges Site,* http://blog.stephenwolfram.-com/2017/11/what-is-a-computational-essay/