agent of change
initial state
final state
action
function
input-output rule
unspecified input
execute
specific inputs

Chapter 4

Addition & Subtraction

4.1 Real-World Actions

Collections usually do not remain unchanged for very long and, given some **agent of change**, collections will change from an **initial state** to a **final state**. Then, the **action** of an *agent of change* is:

$$\text{Collection in initial state} \xRightarrow{\textit{Agent of Change}} \text{Collection in final state}$$

EXAMPLE 4.1. The sun is the agent that changes ***apples*** from being in a *green* state to being in a *ripe* state. In other words, the *action* of the sun is:

$$\text{Collection of } \textit{green } \textbf{\textit{apples}} \xRightarrow{Sun} \text{Collection of } \textit{ripe } \textbf{\textit{apples}}$$

4.2 Paper-World Functions

Real world *agents of change* are represented on paper by **functions** which we specify with an **input-output rule** that consists of:

i. An **unspecified input** which, when we **execute** the function, will be replaced by **specific inputs**, that is by the number phrases that represent the collections in their *initial state*.

function name
output specifying code
specify
specific outputs
attachment
attach
tack-on collection

ii. A **function name** for the function that represents the *agent of change*

iii. An **output specifying code** to **specify** the *output* of the function in terms of the *input*. The **specific outputs** are the number phrases that represent the collections in their *final state*.

Thus, the real world *action* of a given agent of change

$$\text{Initial collection} \xRightarrow{\textit{Agent of Change}} \text{Final collection}$$

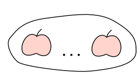
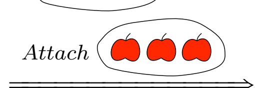
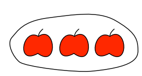
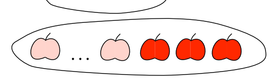
is represented on paper by the *input-output rule* of a function:

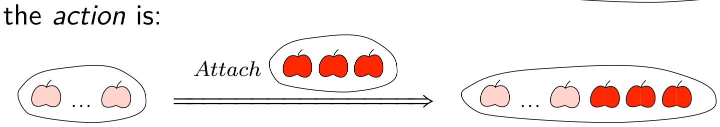
$$\underbrace{\text{Unspecified input}}_{\text{To be replaced by specific inputs}} \xrightarrow{\text{Function Name}} \underbrace{\text{Output-specifying code}}_{\text{Specifies what to do with the input to get the output}}$$

4.3 Attaching A Collection

Given a collection in an *initial* state, an **attachment** is an agent of change which **attaches** a given **tack-on collection** *to* the collection in the initial state to put the collection in a *final* state. However, whether we are dealing with collections of *plain* items or with collections of *oriented* items makes a *huge* difference.

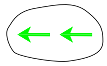
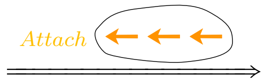
1. With collections of *plain* items, things are pretty straightforward.

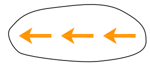
EXAMPLE 4.1. Let  be the *initial state* of a collection. After using the *attachment* , where  is the *tack-on collection*, the *final state* of the collection will be .
In short, the *action* is:

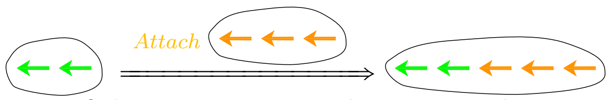


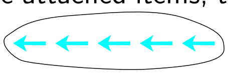
2. In the case of collections of *oriented* items, we may have to *cancel* items of *opposite* orientation.

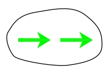
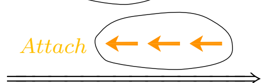
adding function
addition
amount of an addition
add-on number-phrase

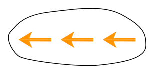
Example 4.2. Let  be the *initial state* of a collection and let the attachment be 

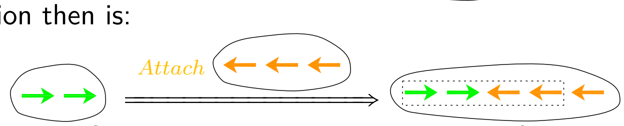
, where  is the *tack-on collection*. The action then is:

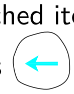


and since the orientation of the initial items is *the same as* the orientation of the attached items, there is no cancellation and the *final state* of the collection is .

Example 4.3. Let  be the *initial state* of a collection and let the attachment be 

, where  is the *tack-on collection*. The action then is:



but since the orientation of the initial items is *the opposite of* the orientation of the attached items, there will be two cancellations and the *final state* of the collection is .

4.4 Adding On Paper

An **adding function**, usually called **addition** for short[1], is a function that represents the *attachment* of a given tack-on collection to a collection in the initial state. The **amount of an addition** is the **add-on number-phrase** that represents the *tack-on collection*.

But it should not come as a surprise that addition of *plain* numerators and addition of *signed* numerators are completely different.

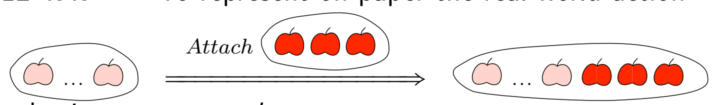
1. With *plain* numerators:

[1]As Educologists well know, addition really is a *unary* operator as the common language shows: add a tip to ..., how much did you add? etc. That group theoreticians found it preferable for *their* purpose to see it as a *binary* operator seems rather irrelevant here.

plain adding function
plain addition
+
context
carryover

- The name of a **plain adding function**, usually called **plain addition** for short, consists of the symbol + to represent *attaching* followed by the amount of the addition.

EXAMPLE 4.4. To represent on paper the real world *action*



we write the *input-output rule*

$$\underbrace{x\ \text{Apples}}_{\text{Unspecified input}} \xrightarrow{+\,3\ \text{Apples}} \underbrace{(x+3)\ \text{Apples}}_{\text{Output specifying code}}$$

where 3 **Apples** is the *amount* of the addition.

> **ALERT 4.1. Overuse of the symbol +.** We already used the symbol + to represent one of the two orientations an oriented collection can have. Using a single symbol to represent two different things can cause much trouble.

The way we decide what the symbol + stands for is by looking at the **context**, that is atthe surrounding symbols, in view of **??** and then by reasoning.

EXAMPLE 4.5. In $2+5$, the symbol + cannot be the sign of 5 because, by **??**, 2 has to be a *plain* numerator and what could a *plain* numerator followed by a *signed* numerator possibly represent? So 2 and 5 are *plain* numerators and the symbol + stands for *plain* addition.

- To execute a *plain addition*, we place the two numerators under a header as in **?? ??** and then, column by column, from right to left, we *count up* with possible "**carryovers**".

EXAMPLE 4.6. To execute

$$4\,627.47\ \text{Meters} \xrightarrow{+\,526.003\ \text{Meters}}$$

we place both numerators under a *header*:

THOUSAND	HUNDRED	TEN	SINGLE	TENTH	HUNDREDTH	THOUSANDTH
4	6	2	7	4	7	
	5	2	6	0	0	3

signed adding function
signed addition

and then we do the *addition* under the header:

THOUSAND	HUNDRED	TEN	SINGLE	TENTH	HUNDREDTH	THOUSANDTH
1		1				
4	6	2	7	4	7	
	5	2	6	0	0	3
5	1	5	3	4	7	3

rked

Which gives us the *decimal number-phrase*:

$5\,153.473$ **Meters**

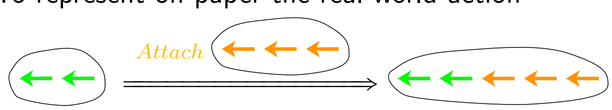
and we can write:

$$4\,627.47 \text{ Meters} \xrightarrow{+\,526.003 \text{ Meters}} [4\,627.47 + 526.003] \text{ Meters} = 5\,153.473 \text{ Meters}$$

2. With *signed* numerators:

- The name of a **signed adding function**, usually called **signed addition** for short, consists of the symbol $\oplus$ (to avoid even more overuse of the symbol $+$) to represent *attaching* followed by the *amount of the addition*.

EXAMPLE 4.7. To represent on paper the real world action

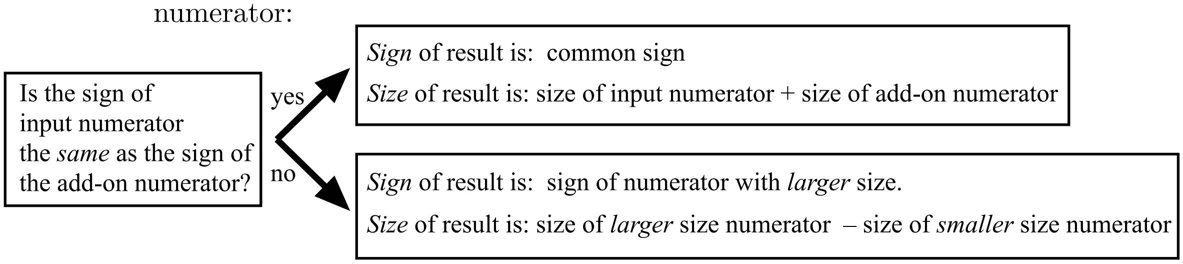


we write the input-output rule

$$-2 \text{ Arrows} \xrightarrow{\oplus -3 \text{ Arrows}} -2 \text{ Arrows} \oplus -3 \text{ Arrows}$$

where -3 **Arrows** is the *amount* of the addition.

- To execute a *signed addition*, the procedure is **forked**: Just like in the real world we must check if the *orientation* of the items in the add-on collection is the *same as* or is the *opposite of* the orientation of the items in the initial collection, on paper we must check if the *sign* of the add-on numerator is the *same as* or the *opposite of* the sign of the input numerator:



detachment
detach
take-from collection

EXAMPLE 4.8. Execute

$$-2\,\textsf{Arrows} \xrightarrow{\oplus -3\,\textsf{Arrows}} -\,2\,\textsf{Arrows} \oplus\ -3\,\textsf{Arrows}$$

Since the two numerators have the same *sign* namely $-$
– The *sign* of the result is the common sign $-$
– The *size* of the result is the size of -2 **Arrows** plus the size of -3 **Arrows**
So:

$$\begin{aligned} -2\,\textsf{Arrows} \xrightarrow{\oplus -3\,\textsf{Arrows}} & -\,2\,\textsf{Arrows} \oplus\ -3\,\textsf{Arrows} \\ & -\,(2+3)\,\textsf{Arrows} \\ & -\,5\,\textsf{Arrows} \end{aligned}$$

EXAMPLE 4.9. Execute

$$+2\,\textsf{Arrows} \xrightarrow{\oplus -3\,\textsf{Arrows}} +\,2\,\textsf{Arrows} \oplus\ -3\,\textsf{Arrows}$$

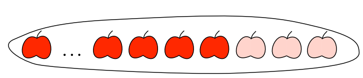
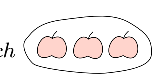
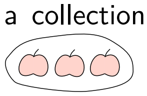
Since the two numerators have *opposite signs* we must *compare the sizes*.
– The *sign* of the result is the size of the numerator with larger size: $-$
– The *size* of the result is the size of -3 **Arrows** minus the size of $+2$ **Arrows**.
So:

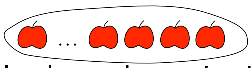
$$\begin{aligned} +2\,\textsf{Arrows} \xrightarrow{\oplus -3\,\textsf{Arrows}} & +\,2\,\textsf{Arrows} \oplus\ -3\,\textsf{Arrows} \\ & -\,(3-2)\,\textsf{Arrows} \\ & -\,1\,\textsf{Arrows} \end{aligned}$$

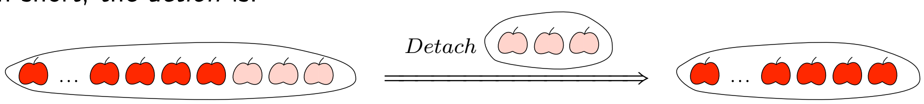
4.5 Detaching A Collection

Given a collection in an *initial* state, a **detachment** is an agent of change which **detaches** a given **take-from collection** *from* the collection in the initial state to get the *final* state of that collection. Again, whether we are dealing with collections of *plain* items or collections of *oriented* items makes a *huge* difference.

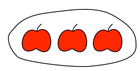
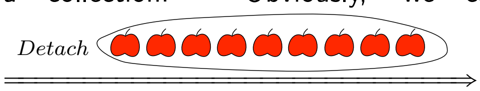
1. With collections of *plain* items, things are pretty straightforward.

EXAMPLE 4.10. Let  be the *initial state* of a collection. After using the agent of change *Detach*  $\Longrightarrow$, where  is the *take-from collection*, the *final state* of the collection will be

.
In short, the *action* is:

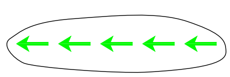
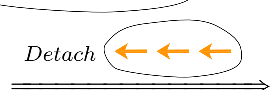
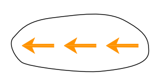


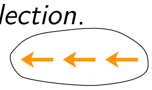
Of course, contrary to what happens with *attaching*, we cannot always *detach* because we cannot detach items that are not already there in the initial state of the collection.

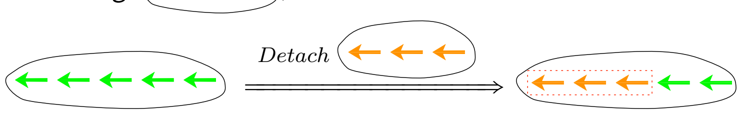
EXAMPLE 4.11. Let  be the *initial state* of a collection. Obviously, we cannot use the agent of change .

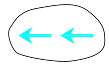
2. In the case of collections of *oriented* items, and contrary to what happened with plain items, we can always *detach* a take-from collection and there are two ways get the *final state*:

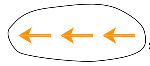
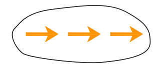
- We can *detach* the take-from collection directly but, depending on the size of the subtract-from collection compared to the size of the initial state of the collection, we may have to imagine the initial collection *as it might have been before cancellations*—which might not be immediately obvious but which can be easily figured out.
- We can *attach the opposite* of the take-from collection and the worst that can happen is that we may then have to cancel items with opposite orientations

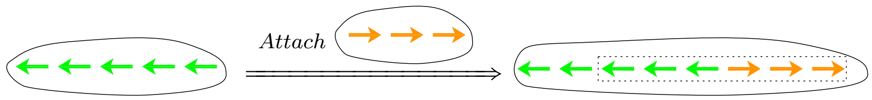
EXAMPLE 4.12. Let  be the *initial state* of a collection and let the detachment be  where  is the *take-from collection*.

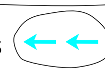
- *Detaching* , the action is

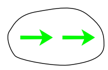
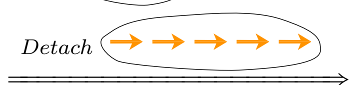
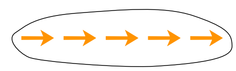


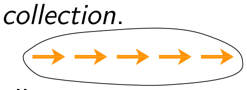
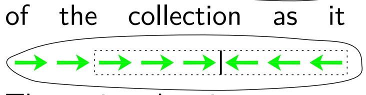
and the final state of the collection is 

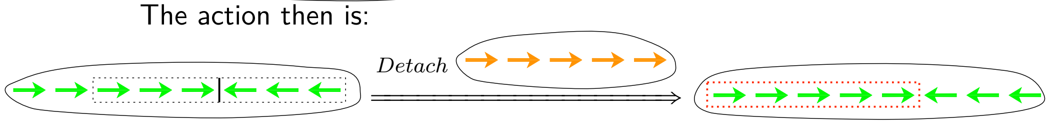
- *Attaching the opposite of* , namely *attaching* , the action is

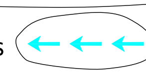


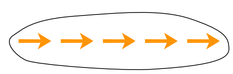
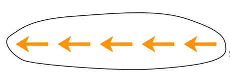
and, after cancellation the final state of the collection is .

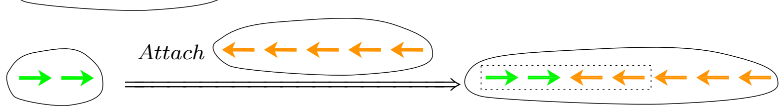
Example 4.13. Let  be the *initial state* of a collection and let the detachment be  where  is the *take-from collection*.

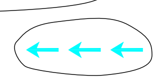
- *Detaching*  requires imagining the initial state of the collection as it might have been before cancellations: .
  The action then is:

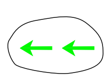


and, after detachment, the final state of the collection is 

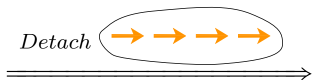
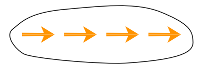
- *Attaching the opposite of*  namely attaching , the action is:

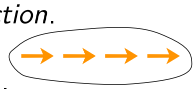
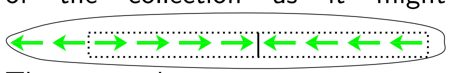


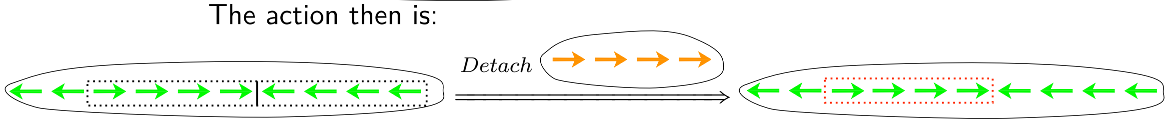
with, after cancellations, the same final state of the collection: 

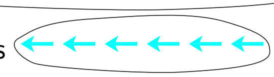
Example 4.14. Let  be the *initial state* of a collection and let

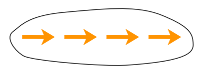
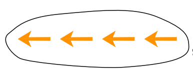
subtracting function
subtraction
amount of a subtraction
subtract-from number-phrase
plain subtracting function
plain subtraction
−

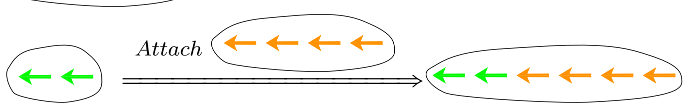
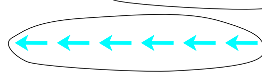
the detachment be  where  is the *take-from collection*.

- *Detaching*  requires imagining the initial state of the collection as it might have been before cancellations: .
  The action then is:



and after detachment the final state of the collection is 

- *Attaching the opposite of*  namely attaching , the action is:

 with the *same final* state of the collection: 

4.6 Subtracting On Paper

A **subtracting function**, usually called **subtraction** for short[2], is a function that represents the *detachment* of a given take-from collection from a collection in the initial state. The **amount of a subtraction** is the **subtract-from number-phrase** that represents the take-from collection.

But it should not come as a surprise that addition of *plain* numerators and addition of *signed* numerators are completely different.

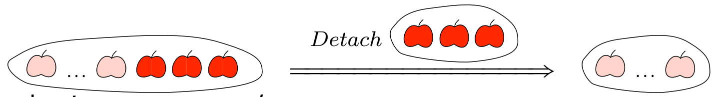
1. With *plain* numerators:

- The name of a **plain subtracting function**, usually called **plain subtraction** for short, consists of the symbol− to represent *detaching* followed by the amount of the subtraction.

[2]As Educologists well know, historically, subtraction was a *unary* operator as the common language still shows: we subtract from, etc. That group theoreticians found it preferable for *their* purpose to see it as a *binary* operator seems rather irrelevant here.

borrowing

EXAMPLE 4.15. To represent on paper the real world *action*



we write the *input-output rule*

$$\underbrace{x \text{ Apples}}_{\textit{Unspecified input}} \xrightarrow{-\,3 \text{ Apples}} \underbrace{x - 3 \text{ Apples}}_{\textit{Output specifying code}}$$

where 3 Apples is the *amount* of the subtraction.

> **ALERT 4.2. Overuse of the symbol $-$.** We already used the symbol $-$ to represent one of the two orientations an oriented collection can have. Using a single symbol to represent two different things can cause much trouble.

The way we decide what the symbol $-$ stands for is by looking at the *context* in view of **??** and then by reasoning.

EXAMPLE 4.16. In $2-5$, the symbol $-$ cannot be the sign of 5 because, by **??**, 2 has to be a *plain* numerator and what could a *plain* numerator followed by a *signed* numerator possibly represent? So 2 and 5 are *plain* numerators and the symbol $-$ stands for *plain* subtraction (which here cannot be done).

- To execute a *plain subtraction*, we place the two numerators under a header as in **??** **??** and then, column by column, from right to left, we *count down* with possible "**borrowings**".

EXAMPLE 4.17. To execute

$$4\,627.47 \text{ Meters} \xrightarrow{-\,926.253 \text{ Meters}}$$

we place both numerators under a *header*:

THOUSAND	HUNDRED	TEN	SINGLE	TENTH	HUNDREDTH	THOUSANDTH
4	6	2	7	4	7	
	9	2	6	2	5	3

and then we do the *subtraction* under the header:

signed subtracting function
gned subtraction

THOUSAND	HUNDRED	TEN	SINGLE	TENTH	HUNDREDTH	THOUSANDTH
	10					10
~~4~~ 3	6	2	7	4	~~7~~ 6	
	9	2	6	2	5	3
3	7	0	1	2	1	7

Which gives us the *decimal number-phrase*:

$$3\,701.217 \text{ Meters}$$

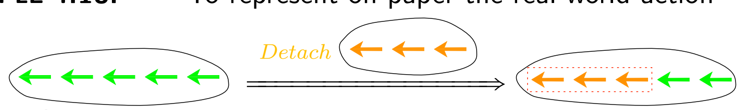
and we can write:

$$4\,627.47 \text{ Meters} \xrightarrow{-\ 926.253 \text{ Meters}} [4\,627.47 - 926.253] \text{ Meters} = 3\,701.217 \text{ Meters}$$

2. With *signed* numerators:

- The name of a **signed subtracting function**, usually called **signed subtraction** for short, consists of the symbol $\ominus$ (to avoid even more overuse of the symbol $-$) to represent *detaching* followed by the *amount* of the subtraction.

EXAMPLE 4.18. To represent on paper the real world action



we write the input-output rule

$$-5 \text{ Arrows} \xrightarrow{\ominus -3 \text{ Arrows}} -5 \text{ Arrows} \ominus -3 \text{ Arrows}$$

where -3 **Arrows** is the *amount* of the subtraction.

- To execute a *signed subtraction*, we will think of the second way of detaching a take-from collection, that is we will *oplus the opposite* of the numerator which represents the take-from collection. In other words:

> **THEOREM 4.1 (Ominus Theorem)** To *ominus* a given number phrase, *oplus* the *opposite* of the given number phrase

EXAMPLE 4.19. In order to execute

$$+3 \text{ Arrows} \xrightarrow{\ominus +5 \text{ Arrows}} +3 \text{ Arrows} \ominus +5 \text{ Arrows}$$

we execute

$$+3 \text{ Arrows} \oplus \text{Opposite } +5 \text{ Arrows}$$

that is

$$+\,3\,\text{Arrows} \oplus -\,5\,\text{Arrows}$$

that is

$$-\,2\,\text{Arrows}$$

Example 4.20. In order to execute

$$+3\,\text{Arrows} \xrightarrow{\ominus +5\,\text{Arrows}} +\,3\,\text{Arrows} \ominus -\,5\,\text{Arrows}$$

we execute

$$+\,3\,\text{Arrows} \oplus \text{Opposite} \ -\,5\,\text{Arrows}$$

that is

$$+\,3\,\text{Arrows} \oplus +\,5\,\text{Arrows}$$

that is

$$+\,8\,\text{Arrows}$$

Example 4.21. In order to execute

$$+3\,\text{Arrows} \xrightarrow{\ominus +5\,\text{Arrows}} -\,3\,\text{Arrows} \ominus +\,5\,\text{Arrows}$$

we execute

$$-\,3\,\text{Arrows} \oplus \text{Opposite} \ +\,5\,\text{Arrows}$$

that is

$$-\,3\,\text{Arrows} \oplus -\,5\,\text{Arrows}$$

that is

$$-\,8\,\text{Arrows}$$

Example 4.22. In order to execute

$$+3\,\text{Arrows} \xrightarrow{\ominus +5\,\text{Arrows}} -\,3\,\text{Arrows} \ominus -\,5\,\text{Arrows}$$

we execute

$$-\,3\,\text{Arrows} \oplus \text{Opposite} \ -\,5\,\text{Arrows}$$

that is

$$-\,3\,\text{Arrows} \oplus +\,5\,\text{Arrows}$$

that is

$$+\,2\,\text{Arrows}$$

4.7 Change

change

When a collection changes from an *initial state* to a *final state*, we often want to know what the **change** is, that is what the *tack-on collection* or the *take-from collection* was. What we do on paper is to subtract the initial state from the final state. As always, whether we are dealing with collections of *plain* items or collections of *oriented* items makes a *huge* difference.

1. With collections of plain items, things are fairly straightforward:

EXAMPLE 4.23. Given that on Tuesday, Jill had a collection of FIFTEEN ***dollars*** and that on Thursday Jill had a collection of TWENTY ONE ***dollars***, what was the *change* from Tuesday to Thursday?
Since she had *more* on Thursday than she had on Tuesday, on Wednesday, Jill *earned*

$$\underbrace{\boxed{21}\ \textbf{Dollars}}_{\text{Thursday}} - \underbrace{\boxed{15}\ \textbf{Dollars}}_{\text{Tuesday}}$$

namely Jill *earned* 6 **Dollars**

EXAMPLE 4.24. Given that on Tuesday, Jack had a collection of THIRTY FOUR ***dollars*** and that on Thursday Jack had a collection of NINETEEN ***dollars***, what was the *change* from Tuesday to Thursday?
Since he had *less* on Thursday than he had on Tuesday,, on Wednesday, Jack *lost*

$$\underbrace{\boxed{34}\ \textbf{Dollars}}_{\text{Thursday}} - \underbrace{\boxed{19}\ \textbf{Dollars}}_{\text{Tuesday}}$$

namely Jack lost 15 **Dollars**

2. In matters of change, though, it is much more efficient to use *signed* numerators and always to *subtract* the *initial* numerator from the *final* numerator. Then, $\ominus$ and the signs automatically take care of the change.

EXAMPLE 4.25. Given that on Tuesday, Jill *had* a collection of FIFTEEN ***dollars*** and that on Thursday Jill *had* a collection of TWENTY ONE ***dollars***, what was the *change* from Tuesday to Thursday?
We represent collections being *had* by *positive* numerators and collections being

owed by *negative* numerators. The change then is:

$$\underbrace{+21\ \textsf{Dollars}}_{\text{Thursday}} \ominus \underbrace{+15\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$\underbrace{+21\ \textsf{Dollars}}_{\text{Thursday}} \oplus \underbrace{-15\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$+\ 6\ \textsf{Dollars}$$

that is Jill *made* 6 **Dollars**

EXAMPLE 4.26. Given that on Tuesday, Jack had a collection of THIRTY FOUR ***dollars*** and that on Thursday Jack had a collection of NINETEEN ***dollars***, what was the *change* from Tuesday to Thursday?
We represent collections being *had* by *positive* numerators and collections being *owed* by *negative* numerators. The change then is:

$$\underbrace{+19\ \textsf{Dollars}}_{\text{Thursday}} \ominus \underbrace{+34\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$\underbrace{+19\ \textsf{Dollars}}_{\text{Thursday}} \oplus \underbrace{-34\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$-\ 15\ \textsf{Dollars}$$

that is Jack *lost* 15 **Dollars**

3. A big advantage of using signs is that we can deal just as easily with *oriented* collections.

EXAMPLE 4.27. Given that on Tuesday, Jill *owed* a collection of FIFTEEN ***dollars*** and that on Thursday Jill *had* a collection of TWENTY ONE ***dollars***, what was the *change* from Tuesday to Thursday?
We represent collections being *had* by *positive* numerators and collections being *owed* by *negative* numerators. The change then is:

$$\underbrace{+21\ \textsf{Dollars}}_{\text{Thursday}} \ominus \underbrace{-15\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$\underbrace{+21\ \textsf{Dollars}}_{\text{Thursday}} \oplus \underbrace{+15\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$+\ 36\ \textsf{Dollars}$$

So, on Wednesday, Jill *made* 36 **Dollars**.

distance
undoes

Example 4.28. Given that on Tuesday, Jack *owed* a collection of TWENTY FOUR ***dollars*** and that on Thursday Jack *owed* a collection of FOURTEEN ***dollars***, what was the *change* from Tuesday to Thursday?
We represent collections being *had* by *positive* numerators and collections being *owed* by *negative* numerators. The change then is:

$$\underbrace{-14\ \textsf{Dollars}}_{\text{Thursday}} \ominus \underbrace{-24\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$\underbrace{-14\ \textsf{Dollars}}_{\text{Thursday}} \oplus \underbrace{+24\ \textsf{Dollars}}_{\text{Tuesday}}$$
$$+10\ \textsf{Dollars}$$

So, on Wednesday, *made* 10 **Dollars**.

4. The **distance** between two *signed* numerators is the *size* of the change from one numerator to the other. This is often a useful concept.

Example 4.29. Find the distance between -14 **Dollars** and -42 **Dollars**, This is the *size* of the change from either one to the other.

- If we compute the change *from* -42 **Dollars** *to* -14 **Dollars**
$$\begin{aligned} -14\ \textsf{Dollars} \ominus -42\ \textsf{Dollars} &= -14\ \textsf{Dollars} \oplus +42\ \textsf{Dollars} \\ &= +28\ \textsf{Dollars} \end{aligned}$$
We then get that the distance *between* -42 **Dollars** *and* -14 **Dollars** is:
Size of $+28$ **Dollars** which is 28 **Dollars**
- If we compute the change *from* -14 **Dollars** *to* -42 **Dollars**
$$\begin{aligned} -42\ \textsf{Dollars} \ominus -14\ \textsf{Dollars} &= -42\ \textsf{Dollars} \oplus +14\ \textsf{Dollars} \\ &= -28\ \textsf{Dollars} \end{aligned}$$
We then get that the distance *between* -42 **Dollars** *and* -14 **Dollars** is:
Size of -28 **Dollars** which is 28 **Dollars**

4.8 Translation Functions On Paper

1. We will say that a function **undoes** another function if, when we input the output of the first function into the second function, the output of the second function is the same as the input of the first function.
We then have:

translation functions
reverse translation

> **THEOREM 4.2** *Signed addition* and *signed subtraction* of the same *amount* undo each other.

EXAMPLE 4.30. The subtraction $\boxed{x} \xrightarrow{\ominus \boxed{+7}} \boxed{\boxed{x} \ominus \boxed{+7}}$ is undone by the addition $\boxed{x} \xrightarrow{\oplus \boxed{+7}} \boxed{\boxed{x} \oplus \boxed{+7}}$.
For instance, the *subtraction* of $\boxed{+7}$:

$$\boxed{-3} \xrightarrow{\ominus \boxed{+7}} \boxed{\boxed{-3} \ominus \boxed{+7}}$$

is undone by the *addition* of $\boxed{+7}$:

$$\boxed{-3 \ominus +7} \xrightarrow{\oplus \boxed{-7}} \boxed{-3 \ominus +7} \oplus \boxed{-7}$$
$$\boxed{-3}$$

2. An immediate consequence of Theorem 4.1 (Ominus Theorem) is that we really need only deal with *signed additions* but it is occasionally more intuitive to use *signed subtraction*. So, in order to be able to talk about *signed addition* and *signed subtraction* in general, we will use the term **translation functions** to include *signed addition* and *signed subtraction*.
Given a translation of a given amount, we will then say that the translation of the *opposite amount* is the **reverse translation** of the given translation. We can then rephrase Theorem 4.2 as:

> **THEOREM 4.2** *Translations* of opposite amounts undo each other.

Index