Chapter 4

Addition & Subtraction

Real-World Actions, 1 • Paper-World Functions, 1 • Attaching A Collection, 2 • Adding On Paper, 3 • Detaching A Collection, 6 • Subtracting On Paper, 9 • Change, 13 • Translation Functions On Paper, 15.

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:

\[ \text{Collection in initial state} \xrightarrow{\text{Agent of Change}} \text{Collection in final state} \]

**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:

\[ \text{Collection of green apples} \xrightarrow{\text{Sun}} \text{Collection of ripe apples} \]

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.
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

\[
\begin{align*}
\text{Initial collection} & \xrightarrow{\text{Agent of Change}} \text{Final collection} \\
\end{align*}
\]

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

![Input-output rule diagram]

 specifies what to do with the input to get the output

\[\text{Unspecified input} \quad \text{Function Name} \quad \text{Output-specifying code}\]

To be replaced by specific inputs

\[\text{Initial collection} \quad \xrightarrow{\text{Agent of Change}} \quad \text{Final collection}\]

\[
\begin{align*}
\text{Unspecified input} & \quad \text{Function Name} \quad \text{Output-specifying code} \\
\end{align*}
\]

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 \(\ldots \) be the *initial state* of a collection. After using the **attachment** where \(\ldots \) is the **tack-on collection**, the *final state* of the collection will be \(\ldots \). In short, the *action* is:

![Attachment example diagram]

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

**Example 4.2.** Let \( \leftarrow \) be the *initial state* of a collection and let the attachment be \( \text{Attach} \), where \( \leftarrow \) is the *tack-on collection*. The action then is:

\[
\begin{array}{c}
\text{Attach} \\
\leftarrow \\
\text{Attach} \\
\end{array}
\]

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 \( \leftarrow \).

**Example 4.3.** Let \( \rightarrow \) be the *initial state* of a collection and let the attachment be \( \text{Attach} \), where \( \leftarrow \) is the *tack-on collection*. The action then is:

\[
\begin{array}{c}
\text{Attach} \\
\rightarrow \\
\text{Attach} \\
\end{array}
\]

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 \( \leftarrow \).

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.
• 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

![Attachment Diagram]

we write the input-output rule

\[
\begin{align*}
x \text{ Apples} + 3 \text{ Apples} & \rightarrow (x + 3) \text{ Apples} \\
\text{Unspecified input} & \rightarrow \text{Output specifying code}
\end{align*}
\]

where \( 3 \text{ Apples} \) 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, at the 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

\[
\begin{align*}
4027.47 \text{ Meters} + 526.003 \text{ Meters}
\end{align*}
\]

we place both numerators under a header:
and then we do the addition under the header:

<table>
<thead>
<tr>
<th>Thousand</th>
<th>Hundred</th>
<th>Ten</th>
<th>Single</th>
<th>Tenth</th>
<th>Hundredth</th>
<th>Thousandth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>2</td>
<td>7</td>
<td>4</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>7</td>
<td>3</td>
</tr>
</tbody>
</table>

Which gives us the decimal number-phrase:

\[5,153.473 \text{ Meters}\]

and we can write:

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

2. With signed numerators:

- The name of a signed adding function, usually called signed addition for short, consists of the symbol \(\mp\) (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

\[\text{Attach} \quad \text{Arrows} \quad \text{Arrows} \quad \text{Arrows}\]

we write the input-output rule

\[-2 \text{ Arrows} \mp -3 \text{ Arrows} = -2 \text{ Arrows} \mp -3 \text{ Arrows}\]

where \(-3 \text{ Arrows}\) 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:

| Is the sign of input numerator the same as the sign of the add-on numerator? |
|-------------------------|-------------------------|-------------------------|
| yes                     | Sign of result is: common sign |
|                         | Size of result is: size of input numerator + size of add-on numerator |
| no                      | Sign of result is: sign of numerator with larger size. |
|                         | Size of result is: size of larger size numerator – size of smaller size numerator |
**EXAMPLE 4.8.** Execute
\[ -2 \text{ Arrows} \oplus -3 \text{ Arrows} = -2 \text{ Arrows} + 3 \text{ Arrows} \]
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 \(-2 \text{ Arrows}\) plus the size of \(-3 \text{ Arrows}\)
So:
\[ -2 \text{ Arrows} \oplus -3 \text{ Arrows} = -2 \text{ Arrows} + -3 \text{ Arrows} \]
\[ = -2 \text{ Arrows} + (2 + 3) \text{ Arrows} \]
\[ = -2 \text{ Arrows} + 5 \text{ Arrows} \]
\[
\]
**EXAMPLE 4.9.** Execute
\[ +2 \text{ Arrows} \oplus -3 \text{ Arrows} = +2 \text{ Arrows} + -3 \text{ Arrows} \]
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 \(-3 \text{ Arrows}\) minus the size of \(+2 \text{ Arrows}\).
So:
\[ +2 \text{ Arrows} \oplus -3 \text{ Arrows} = +2 \text{ Arrows} \oplus -3 \text{ Arrows} \]
\[ = -3 \text{ Arrows} - (2 - 3) \text{ Arrows} \]
\[ = -3 \text{ Arrows} - 1 \text{ Arrows} \]

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 \(\cdots\) be the initial state of a collection. After using the agent of change \(\overset{\text{Detach}}{\text{\rightarrow}}\), where \(\overset{\text{Detach}}{\text{\rightarrow}}\) is the take-from collection, the final state of the collection will be
4.5. Detaching A Collection

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 \( \leftrightarrow \).

- **Attaching the opposite of** \( \leftrightarrow \), namely attaching \( \rightarrow \rightarrow \), the action is:

  \[ \text{Attach} \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow
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.
EXAMPLE 4.15. To represent on paper the real world action

\[
\begin{array}{ccc}
\text{Apples} & - \text{Apples} & \\
\text{Unspecified input} & \text{Output specifying code} & \\
\end{array}
\]

where \text{Apples} is the amount of the subtraction.

ALERT 4.2. Overuse of the symbol \(\ldots\). We already used the symbol \(\ldots\) 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 \(\ldots\) stands for is by looking at the context in view of ?? and then by reasoning.

EXAMPLE 4.16. In \(2 - 5\), the symbol \(\ldots\) 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 \(\ldots\) 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

\[
\begin{array}{c}
4627.47 \text{ Meters} - 926.253 \text{ Meters} \\
\end{array}
\]

we place both numerators under a header:

\[
\begin{array}{cccccccc}
\text{THOUSAND} & \text{HUNDRED} & \text{TEN} & \text{SINGLE} & \text{TENTH} & \text{HUNDREDTH} & \text{THOUSANDTH} \\
4 & 6 & 2 & 7 & 4 & 7 & 3 \\
9 & 2 & 6 & 2 & 5 & & \\
\end{array}
\]

and then we do the subtraction under the header:
4.6. Subtracting On Paper

Which gives us the decimal number-phrase:

\[ 3701.217 \text{ Meters} \]

and we can write:

\[ 4627.47 \text{ Meters} - 926.253 \text{ Meters} = 3701.217 \text{ Meters} \]

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

\[ \begin{array}{c}
\text{Detach} \\
\end{array} \]

\[ \begin{array}{c}
-5 \text{ Arrows} \ominus -3 \text{ Arrows} \\
-5 \text{ Arrows} \ominus -3 \text{ Arrows} \\
\end{array} \]

where \(-3 \text{ Arrows}\) 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 plus 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, plus the opposite of the given number phrase

\[ \begin{array}{c}
+3 \text{ Arrows} \oplus +5 \text{ Arrows} \\
+3 \text{ Arrows} \oplus +5 \text{ Arrows} \\
\end{array} \]

\[ \begin{array}{c}
+3 \text{ Arrows} \oplus \text{ Opposite} +5 \text{ Arrows} \\
\end{array} \]
that is

\[ + 3 \text{ Arrows} \oplus - 5 \text{ Arrows} \]

that is

\[ - 2 \text{ Arrows} \]

\textbf{Example 4.20.} In order to execute

\[ +3 \text{ Arrows} \oplus +5 \text{ Arrows} \]

we execute

\[ +3 \text{ Arrows} \oplus \text{ Opposite} - 5 \text{ Arrows} \]

that is

\[ +3 \text{ Arrows} \oplus + 5 \text{ Arrows} \]

that is

\[ +8 \text{ Arrows} \]

\textbf{Example 4.21.} In order to execute

\[ +3 \text{ Arrows} \oplus +5 \text{ Arrows} \]

we execute

\[ -3 \text{ Arrows} \oplus \text{ Opposite} + 5 \text{ Arrows} \]

that is

\[ -3 \text{ Arrows} \oplus - 5 \text{ Arrows} \]

that is

\[ -8 \text{ Arrows} \]

\textbf{Example 4.22.} In order to execute

\[ +3 \text{ Arrows} \oplus +5 \text{ Arrows} \]

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

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

\[
\begin{array}{c}
\text{21 Dollars} - \text{15 Dollars} \\
\text{Thursday} & \text{Tuesday}
\end{array}
\]

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

\[
\begin{array}{c}
\text{34 Dollars} - \text{19 Dollars} \\
\text{Thursday} & \text{Tuesday}
\end{array}
\]

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, ⊗ 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:

\[
\begin{align*}
\text{Thursday} & \quad +21 \text{ Dollars} & \quad +15 \text{ Dollars} \\
\text{Tuesday} & \quad -15 \text{ Dollars} & \quad -15 \text{ Dollars}
\end{align*}
\]

+ 6 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:

\[
\begin{align*}
\text{Thursday} & \quad +19 \text{ Dollars} & \quad +34 \text{ Dollars} \\
\text{Tuesday} & \quad -34 \text{ Dollars} & \quad -34 \text{ Dollars}
\end{align*}
\]

- 15 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:

\[
\begin{align*}
\text{Thursday} & \quad +21 \text{ Dollars} & \quad -15 \text{ Dollars} \\
\text{Tuesday} & \quad +15 \text{ Dollars} & \quad +15 \text{ Dollars}
\end{align*}
\]

+ 36 Dollars

So, on Wednesday, Jill made 36 Dollars.
4.8. Translation Functions On Paper

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:

\[
\begin{align*}
\text{Thursday} & : -14 \text{ Dollars} \\
\text{Tuesday} & : +24 \text{ Dollars}
\end{align*}
\]

\[+ 10 \text{ Dollars}\]

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 \(-14 \text{ Dollars}\) and \(-42 \text{ Dollars}\), This is the size of the change from either one to the other.

- If we compute the change from \(-42 \text{ Dollars}\) to \(-14 \text{ Dollars}\)
  \[ -14 \text{ Dollars} \oplus -42 \text{ Dollars} = -14 \text{ Dollars} \oplus +42 \text{ Dollars} = +28 \text{ Dollars} \]

  We then get that the distance between \(-42 \text{ Dollars}\) and \(-14 \text{ Dollars}\) is:
  
  Size of \(+28 \text{ Dollars}\) which is \(28 \text{ Dollars}\)

- If we compute the change from \(-14 \text{ Dollars}\) to \(-42 \text{ Dollars}\)
  \[ -42 \text{ Dollars} \oplus -14 \text{ Dollars} = -42 \text{ Dollars} \oplus +14 \text{ Dollars} = -28 \text{ Dollars} \]

  We then get that the distance between \(-42 \text{ Dollars}\) and \(-14 \text{ Dollars}\) is:

  Size of \(-28 \text{ Dollars}\) which is \(28 \text{ Dollars}\)

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:
Theorem 4.2  Signed addition and signed subtraction of the same amount undo each other.

Example 4.30. The subtraction $x - 7$ is undone by the addition $x + 7$.

For instance, the subtraction of $+7$:

$-3 \ominus +7 \rightarrow -3 + 7$ is undone by the addition of $+7$:

$-3 + 7 \oplus -7 \rightarrow -3$.

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

+, 4
-, 9
∅, 11
⊕, 5

action, 1
add-on number-phrase, 3
adding function, 3
addition, 3
agent of change, 1
amount of a subtraction, 9
amount of an addition, 3
attach, 2
attachment, 2

borrowing, 10
carryover, 4
change, 13
context, 4

detach, 6
detachment, 6

execute, 1

final state, 1
forked, 5
function, 1
function name, 2

initial state, 1
input-output rule, 1

output specifying code, 2

plain adding function, 4
plain addition, 4
plain subtracting function, 9
plain subtraction, 9

reverse translation, 16

signed adding function, 5
signed addition, 5
signed subtracting function, 11
signed subtraction, 11
specific inputs, 1
specific outputs, 2
specify, 2
subtract-from number-phrase, 9
subtracting function, 9
subtraction, 9

tack-on collection, 2
take-from collection, 6
translation functions, 16

undoes, 15
unspecified input, 1