All functions in a finite set

The objective of this exercise is to find all distinct bijections between any two finite sets, but we'll explore related themes.

We'll use the following sets as examples.

$$A = \{1, 2, 3\}$$

$$B = \{a, b, c\}$$

Let's permute in A.

$$\{(1 \to a), (2 \to b), (3 \to c)\}$$
$$\{(2 \to a), (1 \to b), (3 \to c)\}$$
$$\{(3 \to a), (1 \to b), (2 \to c)\}$$
$$\{(3 \to a), (2 \to b), (1 \to c)\}$$
$$\{(1 \to a), (3 \to b), (2 \to c)\}$$
$$\{(2 \to a), (3 \to b), (1 \to c)\}$$

And permute in B.

$$\{(1 \to a), (2 \to b), (3 \to c)\}$$
$$\{(1 \to a), (2 \to c), (3 \to b)\}$$
$$\{(1 \to b), (2 \to a), (3 \to c)\}$$
$$\{(1 \to b), (2 \to c), (3 \to a)\}$$
$$\{(1 \to c), (2 \to a), (3 \to b)\}$$
$$\{(1 \to c), (2 \to b), (3 \to a)\}$$

```
(* The code will use the following shorthand notations:
   - (function/@argument) instead of Map[function,argument]
   - DoSomething[#]& instead of Function[{arg},DoSomething[arg]]
   - (x~Take~y) instead of Take[x,y]
   - //Column instead of Column@ *)
```

```
In[∘]:=  Clear[A];
         Clear[B];
         A:={1,2,3};
         B:={"a","b","c"};
         {
             Labeled[Permutations@A,"PermA",Left],
             Labeled[Permutations@B,"PermB",Left],
             Labeled[Subsets@A,"PowerA",Left],
             Labeled[Subsets@B,"PowerB",Left],
             Labeled[Tuples[A,2],"CombsA_2",Left],
             Labeled[Tuples[B,2],"CombsB_2",Left],
             Labeled[Tuples[{A,B}],"CombsA_B",Left]
         }//Column
```

```
         PermA  {{1, 2, 3}, {1, 3, 2}, {2, 1, 3}, {2, 3, 1}, {3, 1, 2}, {3, 2, 1}}
         PermB  {{a, b, c}, {a, c, b}, {b, a, c}, {b, c, a}, {c, a, b}, {c, b, a}}
         PowerA  {{}, {1}, {2}, {3}, {1, 2}, {1, 3}, {2, 3}, {1, 2, 3}}
Out[∘]=  PowerB  {{}, {a}, {b}, {c}, {a, b}, {a, c}, {b, c}, {a, b, c}}
         CombsA_2  {{1, 1}, {1, 2}, {1, 3}, {2, 1}, {2, 2}, {2, 3}, {3, 1}, {3, 2}, {3, 3}}
         CombsB_2  {{a, a}, {a, b}, {a, c}, {b, a}, {b, b}, {b, c}, {c, a}, {c, b}, {c, c}}
         CombsA_B  {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
```

Every possible combination between individual elements in A, B (the last list above) forms a relation.

Intensionally, the relation can be defined as every tuple of size n constructible between n sets.

```
In[∘]:=  Tuples@{A, B}

Out[∘]=  {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
```

A function adds a restriction: B's elements can not repeat.

Each of the below is a function.

```
In[∘]:=  Function[{a}, {a → #}] /@ A & /@ B // Column

         {{1 → a}, {2 → a}, {3 → a}}
Out[∘]=  {{1 → b}, {2 → b}, {3 → b}}
         {{1 → c}, {2 → c}, {3 → c}}
```

Let's find every function from A to $b \in B$.

We'll use sets to denote the domain in each function.

These sets are, each, one of the subsets of A.

```
In[∘]:= {# → b} & /@ Subsets@A // Column

        {{} → b}
        {{1} → b}
        {{2} → b}
        {{3} → b}
Out[∘]= {{1, 2} → b}
        {{1, 3} → b}
        {{2, 3} → b}
        {{1, 2, 3} → b}
```

For subsets $s_1, s_2, \ldots$ in A and elements a, b, c in B, the formed functions are the sets

$$\{(s_1, a), (s_1, b), (s_1, c)\},$$

$$\{(s_1, a), (s_1, b), (s_2, c)\}, \ldots$$

These sets are $B \rightarrow A$ maps.

```
In[∘]:= Transpose@ (Function[{as}, {as → #}] /@ Subsets@A & /@ B) // Column

        {{{} → a}, {{} → b}, {{} → c}}
        {{{1} → a}, {{1} → b}, {{1} → c}}
        {{{2} → a}, {{2} → b}, {{2} → c}}
        {{{3} → a}, {{3} → b}, {{3} → c}}
Out[∘]= {{{1, 2} → a}, {{1, 2} → b}, {{1, 2} → c}}
        {{{1, 3} → a}, {{1, 3} → b}, {{1, 3} → c}}
        {{{2, 3} → a}, {{2, 3} → b}, {{2, 3} → c}}
        {{{1, 2, 3} → a}, {{1, 2, 3} → b}, {{1, 2, 3} → c}}
```

Let’s visualize it as one combination in A per element in B.

```
In[∘]:= Labeled[Function[{as}, {as → #}] /@ Subsets@A, #, Left] & /@ B // Column

        a {{{} → a}, {{1} → a}, {{2} → a}, {{3} → a},
            {{1, 2} → a}, {{1, 3} → a}, {{2, 3} → a}, {{1, 2, 3} → a}}
Out[∘]= b {{{} → b}, {{1} → b}, {{2} → b}, {{3} → b},
            {{1, 2} → b}, {{1, 3} → b}, {{2, 3} → b}, {{1, 2, 3} → b}}
        c {{{} → c}, {{1} → c}, {{2} → c}, {{3} → c},
            {{1, 2} → c}, {{1, 3} → c}, {{2, 3} → c}, {{1, 2, 3} → c}}
```

For each element in B, there is a set of combinations in A.

We’ll find the set of all functions by combining every element in each set of combinations.

For each set, an element is chosen. Then, they’re combined into a tuple. (In this case, a triple.)

Each tuple will be unique, of course.

Flattening into one set to do this doesn’ t work as each set’ s boundaries are lost:

```
In[∘]:= Flatten@ (Map[Function[{as}, {as → #}], Subsets@A] & /@ B)

Out[∘]= {{} → a, {1} → a, {2} → a, {3} → a, {1, 2} → a, {1, 3} → a, {2, 3} → a, {1, 2, 3} → a,
         {} → b, {1} → b, {2} → b, {3} → b, {1, 2} → b, {1, 3} → b, {2, 3} → b, {1, 2, 3} → b,
         {} → c, {1} → c, {2} → c, {3} → c, {1, 2} → c, {1, 3} → c, {2, 3} → c, {1, 2, 3} → c}
```

Here are the combinations in the domain of the function.

```
In[∘]:= Subsets@A

Out[∘]= {{}, {1}, {2}, {3}, {1, 2}, {1, 3}, {2, 3}, {1, 2, 3}}
```

Let’s map each such combination to each element in the codomain.

```
In[∘]:= Tuples@{Subsets@A, {#}} & /@ B // Column

        {{{}, a}, {{1}, a}, {{2}, a}, {{3}, a}, {{1, 2}, a}, {{1, 3}, a}, {{2, 3}, a}, {{1, 2, 3}, a}}
Out[∘]= {{{}, b}, {{1}, b}, {{2}, b}, {{3}, b}, {{1, 2}, b}, {{1, 3}, b}, {{2, 3}, b}, {{1, 2, 3}, b}}
        {{{}, c}, {{1}, c}, {{2}, c}, {{3}, c}, {{1, 2}, c}, {{1, 3}, c}, {{2, 3}, c}, {{1, 2, 3}, c}}
```

Let’s treat the outer set as a matrix and transpose it to visualize the codomain as columns and the domain as rows.

```
In[∘]:= Transpose@(Tuples@{Subsets@A, {#}} & /@ B) // Column

        {{{}, a}, {{}, b}, {{}, c}}
        {{{1}, a}, {{1}, b}, {{1}, c}}
        {{{2}, a}, {{2}, b}, {{2}, c}}
Out[∘]= {{{3}, a}, {{3}, b}, {{3}, c}}
        {{{1, 2}, a}, {{1, 2}, b}, {{1, 2}, c}}
        {{{1, 3}, a}, {{1, 3}, b}, {{1, 3}, c}}
        {{{2, 3}, a}, {{2, 3}, b}, {{2, 3}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1, 2, 3}, c}}
```

For each element in the codomain of the function B, we have a set of all functions with that element as image.
To create a combinative function which includes every element in the codomain, every such function must be considered in combination.
Each of these combinations is a 1:1 relation from a set of such functions to another set of such functions.

In this example, there are three function sets. We need to construct every triple between the three sets.

```
In[∘]:= Tuples@(Tuples@{Subsets@A, {#}} & /@ B)

        {{{{}, a}, {{}, b}, {{}, c}}, {{{}, a}, {{}, b}, {{1}, c}}, {{{}, a}, {{}, b}, {{2}, c}},
         {{{}, a}, {{}, b}, {{3}, c}}, {{{}, a}, {{}, b}, {{1, 2}, c}},
         {{{}, a}, {{}, b}, {{1, 3}, c}}, {{{}, a}, {{}, b}, {{2, 3}, c}}, ⋯ 498 ⋯ ,
         {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1}, c}}, {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{2}, c}},
Out[∘]=  {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{3}, c}}, {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1, 2}, c}},
         {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1, 3}, c}},
         {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{2, 3}, c}},
         {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1, 2, 3}, c}}}

        large output | show less | show more | show all | set size limit...
```

```
In[∘]:= Length@Tuples@(Tuples@{Subsets@A, {#}} & /@ B)

Out[∘]= 512
```

There are 512 functions from *A* to *B*.

Let’s examine the tail of the triple set.

```
In[∘]:= Reverse@Tuples@(Tuples@{Subsets@A, {#}} & /@ B) ~Take~ 20 // Column
```

```
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1, 2, 3}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{2, 3}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1, 3}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1, 2}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{3}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{2}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{1}, c}}
        {{{1, 2, 3}, a}, {{1, 2, 3}, b}, {{}, c}}
        {{{1, 2, 3}, a}, {{2, 3}, b}, {{1, 2, 3}, c}}
Out[∘]= {{{1, 2, 3}, a}, {{2, 3}, b}, {{2, 3}, c}}
        {{{1, 2, 3}, a}, {{2, 3}, b}, {{1, 3}, c}}
        {{{1, 2, 3}, a}, {{2, 3}, b}, {{1, 2}, c}}
        {{{1, 2, 3}, a}, {{2, 3}, b}, {{3}, c}}
        {{{1, 2, 3}, a}, {{2, 3}, b}, {{2}, c}}
        {{{1, 2, 3}, a}, {{2, 3}, b}, {{1}, c}}
        {{{1, 2, 3}, a}, {{2, 3}, b}, {{}, c}}
        {{{1, 2, 3}, a}, {{1, 3}, b}, {{1, 2, 3}, c}}
        {{{1, 2, 3}, a}, {{1, 3}, b}, {{2, 3}, c}}
        {{{1, 2, 3}, a}, {{1, 3}, b}, {{1, 3}, c}}
        {{{1, 2, 3}, a}, {{1, 3}, b}, {{1, 2}, c}}
```

Let’s consider for a moment the set of all relations.

This demands the requirement that elements of *B* do not repeat in the image sets to be lifted.

A relation can be defined as a set of tuples, with each tuple containing elements from all related sets. To find all relations, there must first be found all possible tuples.

```
In[∘]:= Tuples@{A, B}

Out[∘]= {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
```

Now, we combine in any quantity these tuples.

```
In[∘]:= Subsets@Tuples@{A, B}
```

```
        {{}, {{1, a}}, {{1, b}}, {{1, c}}, {{2, a}}, {{2, b}}, ⋯ 500 ⋯ ,
         {{1, a}, {1, b}, {1, c}, {2, a}, {2, c}, {3, a}, {3, b}, {3, c}},
         {{1, a}, {1, b}, {1, c}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}},
         {{1, a}, {1, b}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}},
Out[∘]=  {{1, a}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}},
         {{1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}},
         {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}}

        large output | show less | show more | show all | set size limit...
```

```
In[∘]:= Length@Subsets@Tuples@{A, B}

Out[∘]= 512
```

Let’s look at the tail.

```
In[∘]:= Reverse@Subsets@Tuples@{A, B}~Take~20 // Column

        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}}
Out[∘]= {{1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, b}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, b}, {1, c}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, b}, {1, c}, {2, a}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {3, b}, {3, c}}
        {{1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, b}, {3, c}}
        {{1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, c}}
        {{1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}}
        {{1, a}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, c}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
```

```
In[∘]:= Length@Permutations@Tuples@{A, B}

Out[∘]= 362 880
```

If the order of the tuples mattered, we' d have 362880 “permutative” relations.

Here’s a sample of such “permutative relations” between *A* and *B*:

```
In[∘]:= Permutations@Tuples@{A, B}~Take~20 // Column

        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, a}, {3, c}, {3, b}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, b}, {3, a}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, b}, {3, c}, {3, a}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, c}, {3, a}, {3, b}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {2, c}, {3, c}, {3, b}, {3, a}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {2, c}, {3, b}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {2, c}, {3, c}, {3, b}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {3, b}, {2, c}, {3, c}}
Out[∘]= {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {3, b}, {3, c}, {2, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {3, c}, {2, c}, {3, b}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, a}, {3, c}, {3, b}, {2, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, b}, {2, c}, {3, a}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, b}, {2, c}, {3, c}, {3, a}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, b}, {3, a}, {2, c}, {3, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, b}, {3, a}, {3, c}, {2, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, b}, {3, c}, {2, c}, {3, a}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, b}, {3, c}, {3, a}, {2, c}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, c}, {2, c}, {3, a}, {3, b}}
        {{1, a}, {1, b}, {1, c}, {2, a}, {2, b}, {3, c}, {2, c}, {3, b}, {3, a}}
```

Comparing relations and functions

In our previously assembled function list, the functions' domains were represented as sets.
To match the list of relations, let's decompose these sets into individual elements and form pairs.

Let's separate the tuples into elements.

```
In[∘]:= Thread@{#} & /@ Subsets@A

Out[∘]= {{}, {{1}}, {{2}}, {{3}}, {{1}, {2}}, {{1}, {3}}, {{2}, {3}}, {{1}, {2}, {3}}}
```

The domains decomposed into pairs:

```
In[∘]:= (Function[{as}, Thread@(as → x)] /@ # &) /@ Subsets@A

Out[∘]= {{}, {1 → x}, {2 → x}, {3 → x}, {1 → x, 2 → x},
          {1 → x, 3 → x}, {2 → x, 3 → x}, {1 → x, 2 → x, 3 → x}}
```

Let's substitute the function's images for X:

```
In[∘]:= (Column@Tuples@{Subsets@A, {#}} & /@ B // Column) ~Labeled~ "Before" →
          ((Column@(Function[{b}, Thread[{b, #}]] /@ Subsets@A) &) /@ B) ~Labeled~ "After"

        {{}, a}
        {{1}, a}
        {{2}, a}
        {{3}, a}
        {{1, 2}, a}
        {{1, 3}, a}
        {{2, 3}, a}
        {{1, 2, 3}, a}
        {{}, b}
        {{1}, b}
        {{2}, b}
        {{3}, b}
Out[∘]= {{1, 2}, b}          →
        {{1, 3}, b}
        {{2, 3}, b}
        {{1, 2, 3}, b}
        {{}, c}
        {{1}, c}
        {{2}, c}
        {{3}, c}
        {{1, 2}, c}
        {{1, 3}, c}
        {{2, 3}, c}
        {{1, 2, 3}, c}
            Before

           {}                            {}                           {}
           {{1, a}}                      {{1, b}}                     {{1, c}}
           {{2, a}}                      {{2, b}}                     {{2, c}}
         { {{3, a}}                    , {{3, b}}                   , {{3, c}}                    }
           {{1, a}, {2, a}}              {{1, b}, {2, b}}             {{1, c}, {2, c}}
           {{1, a}, {3, a}}              {{1, b}, {3, b}}             {{1, c}, {3, c}}
           {{2, a}, {3, a}}              {{2, b}, {3, b}}             {{2, c}, {3, c}}
           {{1, a}, {2, a}, {3, a}}      {{1, b}, {2, b}, {3, b}}     {{1, c}, {2, c}, {3, c}}
                                                    After
```

We'll change the representation from sets of combinations of elements in the domain and elements in the image, to sets of combinations of elements in the image with elements in the domain.

```
In[∘]:= Column@Take[Reverse@Tuples@{Subsets@A, Subsets@B}, 20]

        {{1, 2, 3}, {a, b, c}}
        {{1, 2, 3}, {b, c}}
        {{1, 2, 3}, {a, c}}
        {{1, 2, 3}, {a, b}}
        {{1, 2, 3}, {c}}
        {{1, 2, 3}, {b}}
        {{1, 2, 3}, {a}}
        {{1, 2, 3}, {}}
        {{2, 3}, {a, b, c}}
Out[∘]= {{2, 3}, {b, c}}
        {{2, 3}, {a, c}}
        {{2, 3}, {a, b}}
        {{2, 3}, {c}}
        {{2, 3}, {b}}
        {{2, 3}, {a}}
        {{2, 3}, {}}
        {{1, 3}, {a, b, c}}
        {{1, 3}, {b, c}}
        {{1, 3}, {a, c}}
        {{1, 3}, {a, b}}
```

Now, we only need to generate every combination to compose every relation in A, B.

```
In[∘]:= Column@Subsets@Take[Reverse@Tuples@{Subsets@A, Subsets@B}, 3]

        {}
        {{{1, 2, 3}, {a, b, c}}}
        {{{1, 2, 3}, {b, c}}}
Out[∘]= {{{1, 2, 3}, {a, c}}}
        {{{1, 2, 3}, {a, b, c}}, {{1, 2, 3}, {b, c}}}
        {{{1, 2, 3}, {a, b, c}}, {{1, 2, 3}, {a, c}}}
        {{{1, 2, 3}, {b, c}}, {{1, 2, 3}, {a, c}}}
        {{{1, 2, 3}, {a, b, c}}, {{1, 2, 3}, {b, c}}, {{1, 2, 3}, {a, c}}}
```

```
In[∘]:= Length@Tuples[Tuples[{Subsets@A, Subsets@B}], 2]

Out[∘]= 4096
```

```
In[∘]:= {Mean@Map[Length, Tuples[Tuples[{Subsets@A, Subsets@B}], 2]],
          StandardDeviation@Map[Length, Tuples[Tuples[{Subsets@A, Subsets@B}], 2]]}

Out[∘]= {2, 0}
```

```
In[∘]:= Length@Tuples[Map[Function[{b}, Tuples[{Subsets@A, {b}}]], B]]

Out[∘]= 512
```

Function kinds

An injective function is 1:1 and there can be remaining image elements.
We’ll make a change from subsets to elements in the domain.

```
In[∘]:= Column@Map[Function[{b}, Labeled[Column@Map[Function[{as}, {as → b}], A],
            "Function " <> ToString@b, Right]], Take[B, Round@((Length@B) / 2)]]

        {1 → A}
        {2 → A}  Function A
        {3 → A}
Out[∘]= {1 → B}
        {2 → B}  Function B
        {3 → B}
```

A surjective function has no remaining image elements and elements may not be 1:1 mapped.

```
In[∘]:= Column@Map[Function[{b}, Labeled[Column@Map[Function[{as}, {as → b}], Subsets@A],
            "Function " <> ToString@b, Right]], B]

        {{} → a}
        {{1} → a}
        {{2} → a}
        {{3} → a}
        {{1, 2} → a}       Function a
        {{1, 3} → a}
        {{2, 3} → a}
        {{1, 2, 3} → a}
        {{} → b}
        {{1} → b}
        {{2} → b}
        {{3} → b}
Out[∘]= {{1, 2} → b}       Function b
        {{1, 3} → b}
        {{2, 3} → b}
        {{1, 2, 3} → b}
        {{} → c}
        {{1} → c}
        {{2} → c}
        {{3} → c}
        {{1, 2} → c}       Function c
        {{1, 3} → c}
        {{2, 3} → c}
        {{1, 2, 3} → c}
```

To be continued...