Chapter 7-[Functions and onto](https://mfleck.cs.illinois.edu/building-blocks/version-1.3/functions-onto.pdf)

Friday, December 30, 2022

12:24 AM

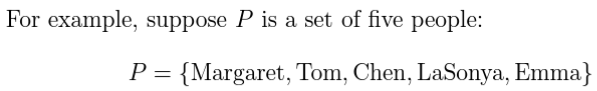
***Functions:***

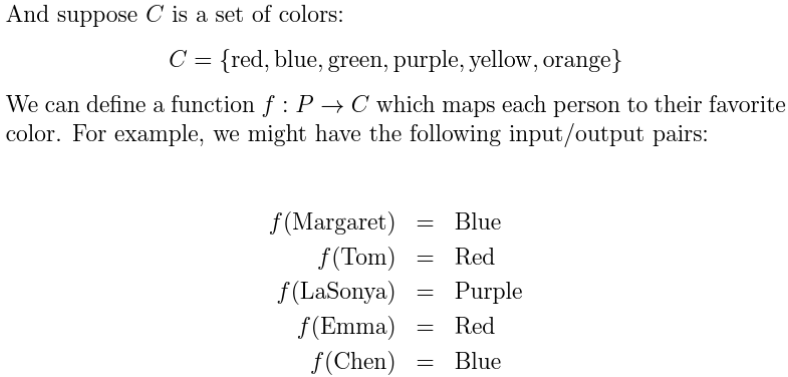
A Function from A to B is an assignment of exactly one element of B (i.e. the output value/*co-domain*) to each element of A (i.e. the input value/*domain*).

*(Note that every input has to have* ***1*** *output for a function to be valid)*

Type Signature:







Since the value of each element in P does NOT depend on the previous values, each element has |C| possibilities.

*(so if there are x elements in P and y elements in C, then there are xy possible ways to construct/define the function)*

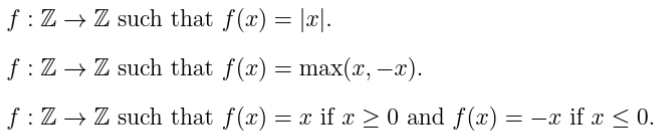
An ***Identity Function*** maps each value to itself in a set.



For two functions to be equal, they have to

1. Have the same output values for all inputs, and
2. Have the same *type signature (which means that the input and output sets have to be same, regardless of how the inputs map to the outputs)*

Examples of equal functions:



***Image:***

The image of a function is a set containing all the values produced by the function.



***Onto:***

A function is *onto* if the image of the function is its entire co-domain(output set).



***Not Onto:***

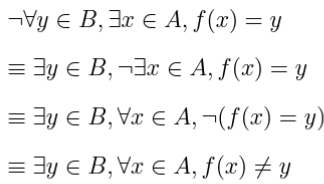
Negate the definition of Onto:



(Treat this as



So the definition of not onto is:



When using onto, the **order** of the **universal** and the **existential quantifiers** matters!

For example:



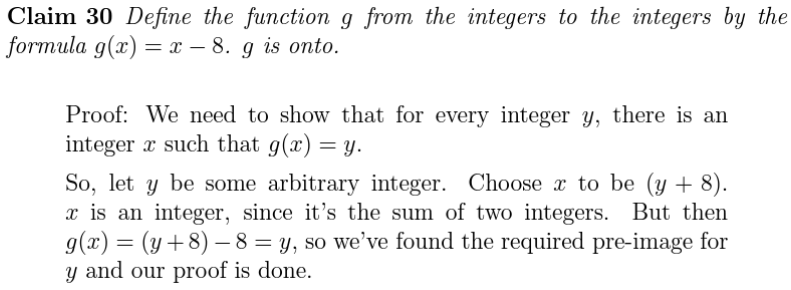
Is not the same as:



*(the quantifiers are flipped, and they mean different things-saying everyone uses a toothbrush vs everyone uses the same toothbrush)*

*(This reminded me that I need to buy a new toothbrush)*

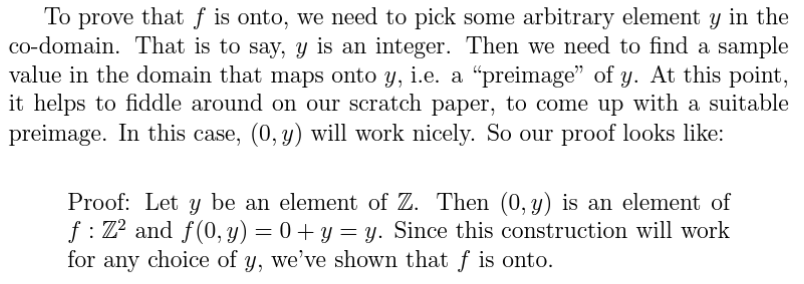
***Proving that a Function is Onto (1-D):***



The trick is to pick a good pre-image for y (in this case (y+8)).

***Proving that a Function is Onto (2-D):***





Trickier than 1-D, but you just need to find a **formula/pre-image** that can cover all possibilities of the **co-domain/output**. (note that not all of the *domain* needs to be covered)

***Composing functions:***





*(Make sure you know that the ordering matters and the domain can be different from the co-domain-input string and output integer, or input char and output real numbers, etc.)*

Avoid using the term "range" as it may refer to either the **image** (the set containing all possible outputs of a specific function), or the **co-domain** (the set containing all elements of a specific type, like reals or integers)