CISC-102
Fall 2017
Week 4

Functions

We have already seen functions in this course. For example:

$$x^2 - 4x + 3$$

We could also write this function as an equation:

$$y = x^2 - 4x + 3$$

In this example you can think of plugging in a (Real) value for x and you will get a distinct value for y. So functions can be viewed as a *mapping* or a *transformation* or even some kind of *machine or algorithm* that takes an input an produces a distinct output.

Underlying every function are two sets (the two sets can be the same).

Let A and B these two sets. We define a function f from A into B as a mapping from <u>every element</u> of A to a <u>distinct element</u> of B. This can be written as:

$$f : A \rightarrow B$$

Vocabulary

Suppose f is a function from the set A to the set B. Then we say that A is the <u>*domain*</u> of f and B is the <u>*codomain*</u> of f. (Synonyms for codomain are: *target set* & *range*)

Notation

Let f denote a function from A to B, then we write:

$$f : \mathrm{A} \rightarrow \mathrm{B}$$

which is pronounced "f is a function from A to B", or "f maps A into B".

If $a \in$ A, and $b \in$ B we can write:

$$f(a) = b$$

to denote that the function *f* maps the element a to b.

More Vocabulary

We can say that *b* is the <u>*image*</u> of *a* under *f*.

More notation.

A function can be expressed by a formula (written as an equation, as illustrated by the following example:

$$f(x) = x^2 \text{ for } x \in \mathbb{R}$$

In this example *f* is the function and *x* is the variable.

Sometimes we can express the image of a variable (the *independent variable*) by a *dependent variable* as follows:

$$y = x^2$$

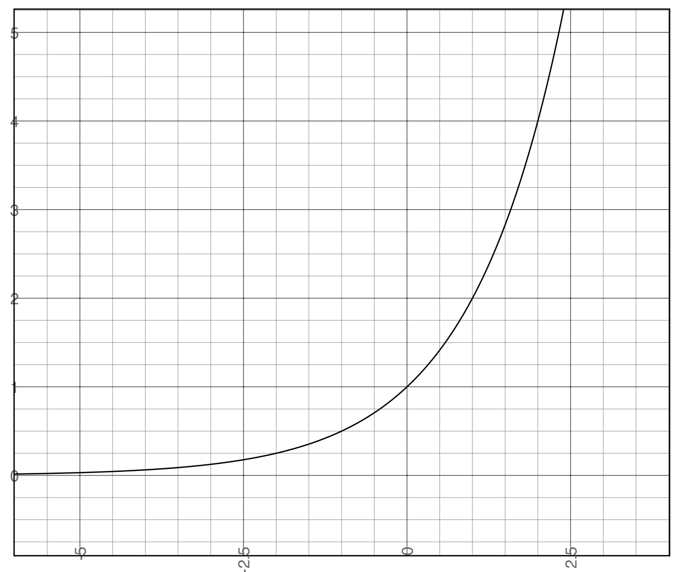
Injective(one-to-one), Surjective(onto), Bijective(one-to-one and onto) functions.

A function f: A → B is a *__one-to-one__* function if for every

a ∈ A there is a distinct image in B. A one-to-one function is also called an *injective function* or an *injection*.

Let $f : \mathbb{R} \to \mathbb{R}$ and $f(x) = 2^x$.

$f(x) = 2^x$ is one-to-one because there is a distinct image for every $x \in \mathbb{R}$, that is if $2^x = 2^y$ then $x = y$.

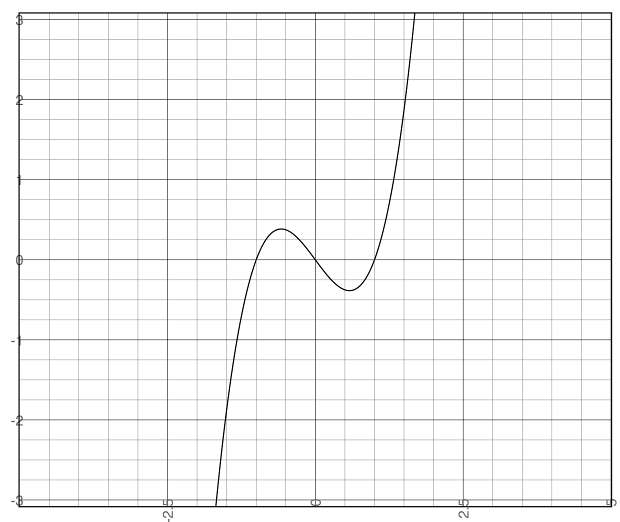


A function *f*: A → B is an <u>*onto*</u> function if

every $b \in B$ is an image. An onto function is also called a *surjective function* or a *surjection*.

Let $f : \mathbb{R} \rightarrow \mathbb{R}$ and $f(x) = x^3 - x$.

$f(x) = x^3 - x$ is onto because the pre-image of any real number y is the solution set of the cubic polynomial equation $x^3 - x - y = 0$ and every cubic polynomial with real coefficients has at least one real root.



Note: $f(x) = x^3 - x = x(x^2 - 1)$ is **not** one-to-one because $f(x) = 0$ for $x = -1$, $x = +1$, $x = 0$

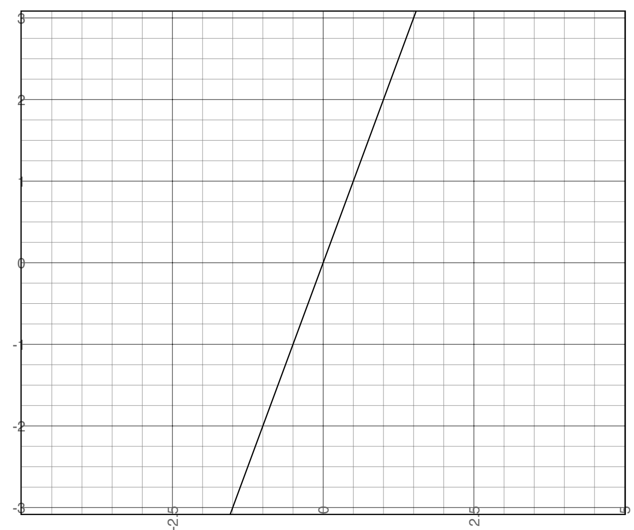
Note: $f(x) = 2^x$ is **not** onto because $2^x > 0$ for all $x \in \mathbb{R}$.

A function that is both one-to-one and onto is called a ***bijective function*** or a ***bijection***.

Let $f : \mathbb{R} \rightarrow \mathbb{R}$ and $f(x) = 2x$

$f(x) = 2x$ is one-to-one because we get a distinct image for every pre-image.

$f(x) = 2x$ is onto because every $y \in \mathbb{R}$ is an image. So $f(x) = 2x$ is a bijection.

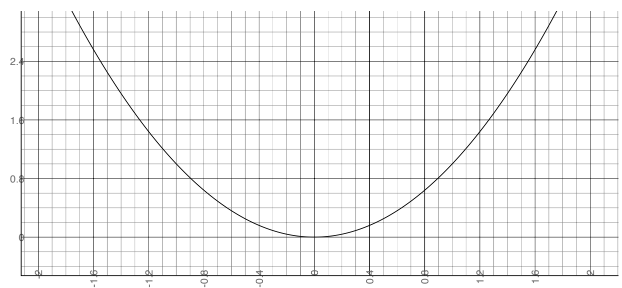


Bijective functions are also called *invertible* functions. That is suppose that f is a bijective function on the set A. Then f^{-1} denotes the inverse of the function f , meaning that whenever $f(x) = y$ we have $f^{-1}(y) = x$.

In our previous example we saw that function $f(x) = 2x$ is a bijective function. In this case we can define

$f^{-1}(x) = x/2$, so we get $f^{-1}(2x) = x$.

Let $f : \mathbb{R} \rightarrow \mathbb{R}$ and $f(x) = x^2$



Observe that $f(x) = x^2$ is a function because every $x \in \mathbb{R}$ has a distinct image. However, $f(x) = x^2$ is neither one-to-one (because $f(x) = f(-x)$) or onto ($f(x) \geq 0$).

Recursively Defined Functions

Recall the factorial function, $n!$. We can define n! and (n+1)! using these explicit iterative formulae:

$n!$ = $1 \times 2 \times 3 \times \cdots \times n$
$(n+1)!$ = $1 \times 2 \times 3 \times \cdots \times n \times (n+1)$

Notice how $(n+1)! = n! \times (n+1)$. This is a recursive definition of the factorial function. More formally we have the following definition.

The Factorial function is defined for non-negative integers, that is {0, 1, 2, 3, …} as follows:

(i) If $n = 0$ then $n! = 1$ (Base)
(ii) If $n > 0$ then $n! = n \times (n-1)!$ (Recursive definition)

Definition: (from SN) A function is said to be recursively defined if it has the following two properties:

i) There must be base values that are given and where the function does not refer to itself.
ii) Each time the function does refer to itself the referred function argument must be closer to the base that the referring function argument.

(In the factorial definition (n-1) is closer to 0, than n is.

We can use a recursive definition for the handshake problem.

Suppose that S is a set consisting of *n* elements, $n \geq 2$.
Q. How many two element subsets are there of the set S?

We need to come up with a base statement and a recursive definition.

The recursive definition is based on the observation, a set of *n* elements has *n*-1 more two element subsets than a set of *n-1* elements.

Let f be a function with domain {2,3,4, …} and range $\mathbb{N}$, such that:

i) $f(2) = 1$ (1 two element subset}
ii) $f(n) = f(n-1) + n-1$.

We can use a recursive definition for the number of values that can be stored in a binary string. The recursive definition is based on the observation that an n bit binary number stores twice as many bits as an (n-1) bit binary number.

Let f be a function on the the Natural numbers such that:

i) $f(1) = 2$ (2 values can be stored in one bit)
ii) $f(n) = f(n-1) \times 2$

We can show using mathematical induction that the closed form for the recursive function is 2^n.

Let P(n) be the proposition that $f(n) = 2^n$, where f(n) is recursively defined as:
i) $f(1) = 2$
ii) $f(n) = 2f(n-1)$

Theorem: $f(n) = 2^n$ for all natural numbers n.
Proof:
Base: $f(1) = 2^1$
Induction Hypothesis: $f(k) = 2^k$
Induction Step: $\mathbf{f(k+1) = 2}\, f(k)$

$$= 2 \times 2^k$$
$$= 2^{k+1}.$$

Therefore by the principle of mathematical induction we conclude that P(*n*) is true for all natural numbers *n*. □