Function- General Information

- Function is a relation between 2 sets. More exactly, it describes relation between elements of **input set (domain)** and **output set (codomain)** such that each **domain member is related exactly to single output** member.

Function- General Information

- Function is a relation between 2 sets. More exactly, it describes relation between elements of **input set** **(domain)** and **output set (codomain)** such that each **domain member is related exactly to single output** member.
- This relation is denoted as $\boldsymbol{F}: \boldsymbol{X} \to \boldsymbol{Y}$, where $\boldsymbol{F}$ – function name, $\boldsymbol{X}$ – is an **input set (domain)**, $\boldsymbol{Y}$ – **output set (codomain)**.

Function- General Information

- Function is a relation between 2 sets. More exactly, it describes relation between elements of **input set (domain)** and **output set (codomain)** such that each **domain member is related exactly to single output** member.
- This relation is denoted as $\boldsymbol{F}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$, where $\boldsymbol{F}$ – function name, $\boldsymbol{X}$ – is an **input set (domain)**, $\boldsymbol{Y}$ – **output set (codomain)**.
- Domain elements sometimes referred as **function arguments**.

Function- General Information

- Function is a relation between 2 sets. More exactly, it describes relation between elements of **input set (domain)** and **output set (codomain)** such that each **domain member is related exactly to single output** member.
- This relation is denoted as $\boldsymbol{F}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$, where $\boldsymbol{F}$ – function name, $\boldsymbol{X}$ – is an **input set (domain)**, $\boldsymbol{Y}$ – **output set (codomain)**.
- Domain elements sometimes referred as **function arguments**.
- Domain elements – exactly all elements that can be passed to a function; codomain – a set, containing a subset with all output values (codomain can contain additional elements).

Function- General Information

- Function is a relation between 2 sets. More exactly, it describes relation between elements of **input set** **(domain)** and **output set (codomain)** such that each **domain member is related exactly to single output** member.
- This relation is denoted as $\boldsymbol{F}$: $\boldsymbol{X} \rightarrow \boldsymbol{Y}$, where $\boldsymbol{F}$ – function name, $\boldsymbol{X}$ – is an **input set (domain)**, $\boldsymbol{Y}$ – **output set (codomain)**.
- Domain elements sometimes referred as **function arguments**.
- Domain elements – exactly all elements that can be passed to a function; codomain – a set, containing a subset with all output values (codomain can contain additional elements).
- E.g. $F(x) = x^2$: $R \rightarrow R$ has domain equals to set real numbers, and codomain $\boldsymbol{R}$, although it does not contain negative elements. Codomain $F[X]$ is called **set image** ***iff*** it contains elements $\boldsymbol{F}(\boldsymbol{x})$ produced by function $\boldsymbol{F}$ for all possible domain $\boldsymbol{X}$ members. Image is denoted by square brackets.

Function- General Information

- Function is a relation between 2 sets. More exactly, it describes relation between elements of **input set (domain)** and **output set (codomain)** such that each **domain member is related exactly to single output** member.
- This relation is denoted as $\boldsymbol{F}: \boldsymbol{X} \rightarrow \boldsymbol{Y}$, where $\boldsymbol{F}$ – function name, $\boldsymbol{X}$ – is an **input set (domain)**, $\boldsymbol{Y}$ – **output set (codomain)**.
- Domain elements sometimes referred as **function arguments**.
- Domain elements – exactly all elements that can be passed to a function; codomain – a set, containing a subset with all output values (codomain can contain additional elements).
- E.g. $F(x) = x^2 \colon R \rightarrow R$ has domain equals to set real numbers, and codomain $\boldsymbol{R}$, although it does not contain negative elements. Codomain $F[X]$ is called **set image** ***iff*** it contains elements $\boldsymbol{F}(\boldsymbol{x})$ produced by function $\boldsymbol{F}$ for all possible domain $\boldsymbol{X}$ members. Image is denoted by square brackets.
- For function $\boldsymbol{F}: \boldsymbol{X} \rightarrow \boldsymbol{Y}, \boldsymbol{F}[\boldsymbol{X}] \subseteq \boldsymbol{Y}$. There's also reverse operation: finding preimage by image set. Operation of finding preimage is what usually done when you are brute forcing password given a hash values (image).

Function

- **Task:** Find preimage:

$$sin^{-1}[\{0.7017, 0.52742\}]$$

- **Solution:**

$$\{\forall k \in Z \mid \pm\text{arcsin}(0.7017) + 2k\pi, \pm\text{arcsin}(0.52742) + 2k\pi\}$$

$$= \left\{\forall k \in Z \middle| \pm\frac{7}{9} + 2k\pi, \pm\frac{5}{9} + 2k\pi\right\}$$

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\boldsymbol{\sim}: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\boldsymbol{\sim} == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
    - You can draw arrow diagrams for explicit graphs.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
   - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \to \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
   - You can draw arrow diagrams for explicit graphs.
   - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim : \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
    - You can draw arrow diagrams for explicit graphs.
    - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**

2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \to \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
    - You can draw arrow diagrams for explicit graphs.
    - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**

2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
    - arithmetic is allowed on a set of real numbers,

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
   - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
   - You can draw arrow diagrams for explicit graphs.
   - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**

2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
   - arithmetic is allowed on a set of real numbers,
   - ***k-th*** derivative is allowed on a set of smooth functions,

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
   - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim : \boldsymbol{B} \to \boldsymbol{B}$ can be easily explicitly described as $\sim\ == \{(\mathbf{0},\mathbf{1}),(\mathbf{1},\mathbf{0})\}$.
   - You can draw arrow diagrams for explicit graphs.
   - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**

2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
   - arithmetic is allowed on a set of real numbers,
   - ***k-th*** derivative is allowed on a set of smooth functions,
   - “translation to Russian” is allowed on a set of English words, etc.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim : \boldsymbol{B} \to \boldsymbol{B}$ can be easily explicitly described as $\sim\ == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
    - You can draw arrow diagrams for explicit graphs.
    - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**
2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
    - arithmetic is allowed on a set of real numbers,
    - ***k-th*** derivative is allowed on a set of smooth functions,
    - "translation to Russian" is allowed on a set of English words, etc.
    - Moreover, function description can combine multiple formulas into algorithms. E.g.
    - we can say "first multiply by 2, then add 1" to describe a function for building odd numbers.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim\ == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
    - You can draw arrow diagrams for explicit graphs.
    - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**

2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
    - arithmetic is allowed on a set of real numbers,
    - ***k-th*** derivative is allowed on a set of smooth functions,
    - "translation to Russian" is allowed on a set of English words, etc.
    - Moreover, function description can combine multiple formulas into algorithms. E.g.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim\ == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
    - You can draw arrow diagrams for explicit graphs.
    - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**

2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
    - arithmetic is allowed on a set of real numbers,
    - ***k-th*** derivative is allowed on a set of smooth functions,
    - "translation to Russian" is allowed on a set of English words, etc.
    - Moreover, function description can combine multiple formulas into algorithms. E.g.
    - we can say "first multiply by 2, then add 1" to describe a function for building odd numbers.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim\ == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$**.**
    - You can draw arrow diagrams for explicit graphs.
    - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**
2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
    - arithmetic is allowed on a set of real numbers,
    - ***k-th*** derivative is allowed on a set of smooth functions,
    - "translation to Russian" is allowed on a set of English words, etc.
    - Moreover, function description can combine multiple formulas into algorithms. E.g.
    - we can say "first multiply by 2, then add 1" to describe a function for building odd numbers.
3. We can use piecewise definition in cases, when it is hard to enumerate elements:

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
    - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \to \boldsymbol{B}$ can be easily explicitly described as $\sim\ == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
    - You can draw arrow diagrams for explicit graphs.
    - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**
2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
    - arithmetic is allowed on a set of real numbers,
    - ***k-th*** derivative is allowed on a set of smooth functions,
    - "translation to Russian" is allowed on a set of English words, etc.
    - Moreover, function description can combine multiple formulas into algorithms. E.g.
    - we can say "first multiply by 2, then add 1" to describe a function for building odd numbers.
3. We can use piecewise definition in cases, when it is hard to enumerate elements:

$$sign(x) = \begin{cases} 1, x > 0 \mid same\ as\ \forall x > 0, (x, 1) \\ -1, x < 0 \mid same\ as\ \forall x < 0, (x, -1) \\ 0, x = 0 \mid same\ as\ (0, 0) \end{cases}$$

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
   - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
   - You can draw arrow diagrams for explicit graphs.
   - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**
2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
   - arithmetic is allowed on a set of real numbers,
   - ***k-th*** derivative is allowed on a set of smooth functions,
   - "translation to Russian" is allowed on a set of English words, etc.
   - Moreover, function description can combine multiple formulas into algorithms. E.g.
   - we can say "first multiply by 2, then add 1" to describe a function for building odd numbers.
3. We can use piecewise definition in cases, when it is hard to enumerate elements:

$$sign(x) = \begin{cases} 1, x > 0 \mid same\ as\ \forall x > 0, (x, 1) \\ -1, x < 0 \mid same\ as\ \forall x < 0, (x, -1) \\ 0, x = 0 \mid same\ as\ (0, 0) \end{cases}$$

- Function can accept multiple attributes. To unify approach of describing functions, we can use Cartesian product of input sets to find domain – these are all possible tuples of input set members. Cartesian product is calculated when you implement cross join (inner join without constraint) in SQL statement.

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
   - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \to \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0}, \mathbf{1}), (\mathbf{1}, \mathbf{0})\}$.
   - You can draw arrow diagrams for explicit graphs.
   - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**
2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
   - arithmetic is allowed on a set of real numbers,
   - ***k-th*** derivative is allowed on a set of smooth functions,
   - "translation to Russian" is allowed on a set of English words, etc.
   - Moreover, function description can combine multiple formulas into algorithms. E.g.
   - we can say "first multiply by 2, then add 1" to describe a function for building odd numbers.
3. We can use piecewise definition in cases, when it is hard to enumerate elements:

$$sign(x) = \begin{cases} 1, x > 0 \mid same\ as\ \forall x > 0, (x, 1) \\ -1, x < 0 \mid same\ as\ \forall x < 0, (x, -1) \\ 0, x = 0 \mid same\ as\ (0, 0) \end{cases}$$

   - Function can accept multiple attributes. To unify approach of describing functions, we can use Cartesian product of input sets to find domain – these are all possible tuples of input set members. Cartesian product is calculated when you implement cross join (inner join without constraint) in SQL statement.
   - **SELECT * FROM** Person, City

Function- Definition

- Function can be **described** in multiple ways.

1. We can describe function explicitly using ordered pairs, where first element is from domain set, and second corresponding codomain member: $(x, F(x))$.
   - Set of all possible pairs is called **graph of a function**. For example, logical negation $\sim: \boldsymbol{B} \rightarrow \boldsymbol{B}$ can be easily explicitly described as $\sim == \{(\mathbf{0},\mathbf{1}),(\mathbf{1},\mathbf{0})\}$.
   - You can draw arrow diagrams for explicit graphs.
   - If both domain and codomain are real, ordered pairs are called **Cartesian coordinates.**
2. We can use operations, combined into formulas, allowed on our domain set for building a graph set. E.g.,
   - arithmetic is allowed on a set of real numbers,
   - ***k-th*** derivative is allowed on a set of smooth functions,
   - “translation to Russian” is allowed on a set of English words, etc.
   - Moreover, function description can combine multiple formulas into algorithms. E.g.
   - we can say “first multiply by 2, then add 1” to describe a function for building odd numbers.
3. We can use piecewise definition in cases, when it is hard to enumerate elements:

$$sign(x) = \begin{cases} 1, x > 0 \mid same\ as\ \forall x > 0, (x, 1) \\ -1, x < 0 \mid same\ as\ \forall x < 0, (x, -1) \\ 0, x = 0 \mid same\ as\ (0, 0) \end{cases}$$

   - Function can accept multiple attributes. To unify approach of describing functions, we can use Cartesian product of input sets to find domain – these are all possible tuples of input set members. Cartesian product is calculated when you implement cross join (inner join without constraint) in SQL statement.
   - **SELECT * FROM** Person, City
   - Important property of Cartesian product is $|A * B| = |A| * |B|$. That means that if you write unconstrained SQL join for two 1000-record tables, your statement would have to process 1M lines

Injective Function

- **Injective function** mean that your function never produce similar output for different domain values. E.g. x^3, $\log_2 x$, *inverse(word)* are injective. x^2, *count Letters(word)* are non-injective.

Injective Function

- **Injective function** mean that your function never produce similar output for different domain values. E.g. $x^3, \log_2 x$, *inverse(word)* are injective. x^2, *count Letters(word)* are non-injective.
   1. Are hash functions injective? (no, that's the main purpose of hash function)

Injective Function

- **Injective function** mean that your function never produce similar output for different domain values. E.g. $x^3, \log_2 x$, *inverse(word)* are injective. x^2, *count Letters(word)* are non-injective.
  1. Are hash functions injective? (no, that's the main purpose of hash function)
  2. Prove or disprove that $F(x) = (3x + 2) \% 11 : [0..10] \rightarrow Z$ is injective. (Either code or modular arithmetic). $F(x): [0..11] \rightarrow Z$?

Surjective Function

- **Surjective functions** are functions where codomain is exactly an image of the domain – each element in codomain has at least one related element in a domain.

1. Non-injective and surjective:

$$\mathbf{R} \to \mathbf{R} : x \mapsto (x-1)x(x+1) = x^3 - x$$

$$\mathbf{R} \to [-1, 1] : x \mapsto \sin(x)$$

Bijective Function

- **Bijective** functions are both injective and surjective.

For every set A the **identity function** $\boldsymbol{id_A}$ and thus specifically $\mathbf{R} \rightarrow \mathbf{R} : x \mapsto x$

$\mathbf{R}^+ \rightarrow \mathbf{R}^+ : x \mapsto x^2$ and thus also its inverse $\mathbf{R}^+ \rightarrow \mathbf{R}^+ : x \mapsto \sqrt{x}$

$\exp : \mathbf{R} \rightarrow \mathbf{R}^+ : x \mapsto \mathrm{e}^x$ $\qquad \ln : \mathbf{R}^+ \rightarrow \mathbf{R} : x \mapsto \ln x$

Function Composition

- We often use function composition, when write something like this: $\sin(x^2)$ or this e^{x+y}. The former can be generalized to $t(x) = g(f(x))$, the latter to $t(x, y) = g(f(x, y))$. This relation is called composition, and if you want to write it without variables, in will be $t = g \circ f$. For composition it is very important, that co-domain of $\boldsymbol{f}$ is a subset of domain of $\boldsymbol{g}$.

Function Composition

- Try the following example in your browser consoles (F12) together with students. Refresh that in JavaScript functions are first-class citizens and can be stored in variables. While typing, ask them to give definition to functions: $g:R \to R$, $f:R \to R$, i: words→words, h: words→Z^+
- g = function(x) { return x*x; }
- f = function(x) { return Math.sqrt(x); }
- f(g(100))
- g(f(100))
- // after these lines ask about the difference and ask to write formulas at the whiteboard
- h = function(x) { return x.split('').reverse().join(''); }
- i = function(x) { return x.length; }
- h('word')
- g(h('word'))
- // what's going wrong here? – domain of g and codomain of h don't match
- g(i(h('word')))
- // why it works now? – Z^+ is a subset of R, so it's ok
- Let's return to our functions g() and f(). Write
- t1 = function(x) { return g(f(x)); }
- t2 = function(x) { return f(g(x)); }
- // try these functions t1(x) and t2(x) with different params. Why they are giving the same result? Why this result is equal to param?

Function Composition

- There special operation, called **inverse of the function**. This operation "switches sides" in ordered pairs: $f: X \rightarrow Y$; $f^{-1}: Y \rightarrow X$, defined by $f^{-1}\{\,(y, x) | y = f(x)\}$. If we want $f^{-1}(x)$ be a function, we have to ensure, f is bijective.

Function Composition

- There special operation, called **inverse of the function**. This operation "switches sides" in ordered pairs: $f{:}X\rightarrow Y$; $f^{-1}{:}Y\rightarrow X$, defined by $f^{-1}\{\,(y,x)|y=f(x)\}$. If we want $f^{-1}(x)$ be a function, we have to ensure, f is bijective.
- **Why $\boldsymbol{f(x)}$ cannot be**
  - Only Injective
  - Only Surjective

Function Composition

- There special operation, called **inverse of the function**. This operation "switches sides" in ordered pairs: *f:X⟶Y ; f⁻¹:Y⟶X,* defined by $f^{-1}\{(y,x)|y=f(x)\}$. If we want $f^{-1}(x)$ be a function, we have to ensure, f is bijective.
- **Why $\boldsymbol{f(x)}$ cannot be**
  - Only Injective
  - Only Surjective
    - If f is an invertible function with domain x and range y, then

Function Composition

- There special operation, called **inverse of the function**. This operation “switches sides” in ordered pairs: *f:X⟶Y ; f⁻¹:Y⟶X,* defined by $f^{-1}\{\,(y,x)|y=f(x)\}$. If we want $f^{-1}(x)$ be a function, we have to ensure, f is bijective.
- **Why $\boldsymbol{f(x)}$ cannot be**
  - Only Injective
  - Only Surjective
    - If f is an invertible function with domain x and range y, then
    - $f^{-1}\big(f(x)\big)=x$, for every $x\ \in X$

Function Composition

- There special operation, called **inverse of the function**. This operation "switches sides" in ordered pairs: $f{:}X \rightarrow Y$; $f^{-1}{:}Y \rightarrow X$, defined by $f^{-1}\{\,(y,x)|y = f(x)\}$. If we want $f^{-1}(x)$ be a function, we have to ensure, f is bijective.
- **Why $f(x)$ cannot be**
  - Only Injective
  - Only Surjective
    - If f is an invertible function with domain x and range y, then
    - $f^{-1}\big(f(x)\big) = x$, for every $x \in X$
    - Using the composition of functions we can rewrite this statement as follows: $f^{-1}\, o\, f = \, id_x$.

Function Composition

- There special operation, called **inverse of the function**. This operation "switches sides" in ordered pairs: $f{:}X\to Y$; $f^{-1}{:}Y\to X$, defined by $f^{-1}\{\,(y,x)|y=f(x)\}$. If we want $f^{-1}(x)$ be a function, we have to ensure, f is bijective.
- **Why $\boldsymbol{f(x)}$ cannot be**
  - Only Injective
  - Only Surjective
    - If f is an invertible function with domain x and range y, then
    - $f^{-1}\big(f(x)\big)=x$, for every $x\in X$
    - Using the composition of functions we can rewrite this statement as follows: $f^{-1}\,o\,f=id_x$.

- Knowing that $F=f(C)=\frac{9}{5}C+32$ implement F in any programming language. Evaluate and implement $F^{-1}(C)=C(F)$. Check combination is giving you identity function.