FUNCTIONS

<u>**PRESENTED BY**</u>**:-**

ER. HANIT KARWAL

ASSISTANT PROFESSOR

INFORMATION TECHNOLOGY DEPT.

GNDEC, LUDHIANA

hanitgndec@gmail.com

FUNCTIONS

- Suppose that to each element of a set A *we assign a unique element of a set B; the collection of such*

  assignments is called *a function from A into B*.

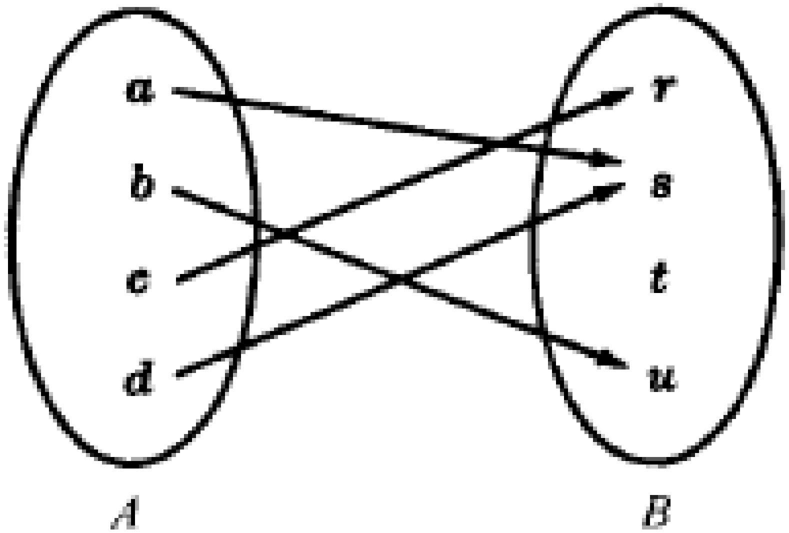
  The set A is called the domain of the function, and the set B is called the *target set or codomain*.
- Functions are ordinarily denoted by symbols. For example, let *f denote a function from A into B*.
- *Then we* write $f: A \rightarrow B$

  which is read: "*f is a function from A into B,*" *or "f takes (or maps) A into B.*"
- If $a \in A$, *then f (a) (read: "f of a")* denotes the unique element of *B which f assigns to a; it is called the image of a under f, or the value of f at a*.
- The set of all image values is called the *range or image of f*.

FUNCTIONS

- Let A and B be two finite sets having m and n elements respectively. Then total number of functions from A to B is n ^m.

DOMAIN – CODOMAIN - RANGE



DOMAIN: { a , b, c, d }
CODOMAIN : {r, s, t, u }
RANGE: { r, s , u}

f(a)= s
f(b)= u
f (c)= r
f(d)= s

Functions as Relations

- A function $f: A \rightarrow B$ *is a relation from A to B (i.e., a subset of A×B) such that each a* $\in$ *A belongs* to a unique ordered pair *(a, b) in f.*
- **Consider the following three relations on the set**
- ***A = {1, 2, 3}:***

  $f = \{(1, 3), (2, 3), (3, 1)\}$, $g = \{(1, 2), (3, 1)\}$,

  $h = \{(1, 3), (2, 1), (1, 2), (3, 1)\}$

- *f is a function from A into A since each member of A appears as the first coordinate in exactly one ordered* pair in *f; here*
- $f(1) = 3$, $f(2) = 3$, *and* $f(3) = 1$
- *g is not a function from A into A since 2* $\in$ *A is not the*

  first coordinate of any pair in *g and so g does not assign any image to 2.*
- *h is not a function from A into A since 1* $\in$ *A appears as the first coordinate of two distinct ordered pairs in h, (1, 3) and (1, 2). If h is to be* a function it cannot assign both 3 and 2 to the element 1 $\in$ *A.*

EXERCISE

- **Let $X = \{1, 2, 3, 4\}$. *Determine whether each relation on X is a function from X into X.***

  (a) $f = \{(2, 3), (1, 4), (2, 1), (3.2), (4, 4)\}$

  (b) $g = \{(3, 1), (4, 2), (1, 1)\}$

  (c) $h = \{(2, 1), (3, 4), (1, 4), (2, 1), (4, 4)\}$

  Recall that a subset *f of* $X \times X$ *is a function* $f: X \to X$ *if and only if each* $a \in X$ *appears as the first coordinate* in exactly one ordered pair in *f*.

  (*a*) *No. Two different ordered pairs (2, 3) and (2, 1) in f have the same number 2 as their first coordinate.*

  (*b*) *No. The element* $2 \in X$ *does not appear as the first coordinate in any ordered pair in g.*

  (*c*) *Yes. Although* $2 \in X$ *appears as the first coordinate in two ordered pairs in h, these two ordered pairs are equal.*

Composition Function

- Consider functions $f: A \to B$ *and* $g: B \to C$; *that is, where the codomain of f is the domain of g. Then we* may define a new function from A *to* C, *called the composition of f and g and written* $g \circ f$, *as follows:*

$$\boldsymbol{(g \circ f)(a) \equiv g(f(a))}$$

*here we use the functional notation $g \circ f$ *for the composition of f and g instead of the* notation $f \circ g$ *which was used for relations.*

EXERCISE

- Let $A = \{a, b, c\}, B = \{x, y, z\}, C = \{r, s, t\}$.
  Let $f: A \to B$ *and* $g: B \to C$ *be defined by:*

$\boldsymbol{f = \{(a, y)(b, x), (c, y)\}}$ **;** $\boldsymbol{g = \{(x, s), (y, t), (z, r)\}}$**.**

Find: (*a*) *composition function* $g \circ f: A \to C$;

(*b*) $Im(f)$, $Im(g)$, $Im(g \circ f)$.

(a) Use the definition of the composition function to compute:

$(g \circ f)(a) = g(f(a)) = g(y) = t$

$(g \circ f)(b) = g(f(b)) = g(x) = s$

$(g \circ f)(c) = g(f(c)) = g(y) = t$

That is $g \circ f = \{(a, t), (b, s), (c, t)\}$.

(*b*) *Find the image points (or second coordinates):*

$\text{Im}(f) = \{x, y\}$, $Im(g) = \{r, s, t\}$, $Im(g \circ f) = \{s, t\}$

EXERCISE

- Let $f : \boldsymbol{R} \rightarrow \boldsymbol{R}$ ***and g:*** $\boldsymbol{R} \rightarrow \boldsymbol{R}$ ***be defined by***

$f(x) = 2x + 1$ and $g(x) = x^2 - 2$.

Find the formula for the composition function $g \circ f$.

$$(g \circ f)(x) = g(f(x))$$

$$= g(2x + 1)$$

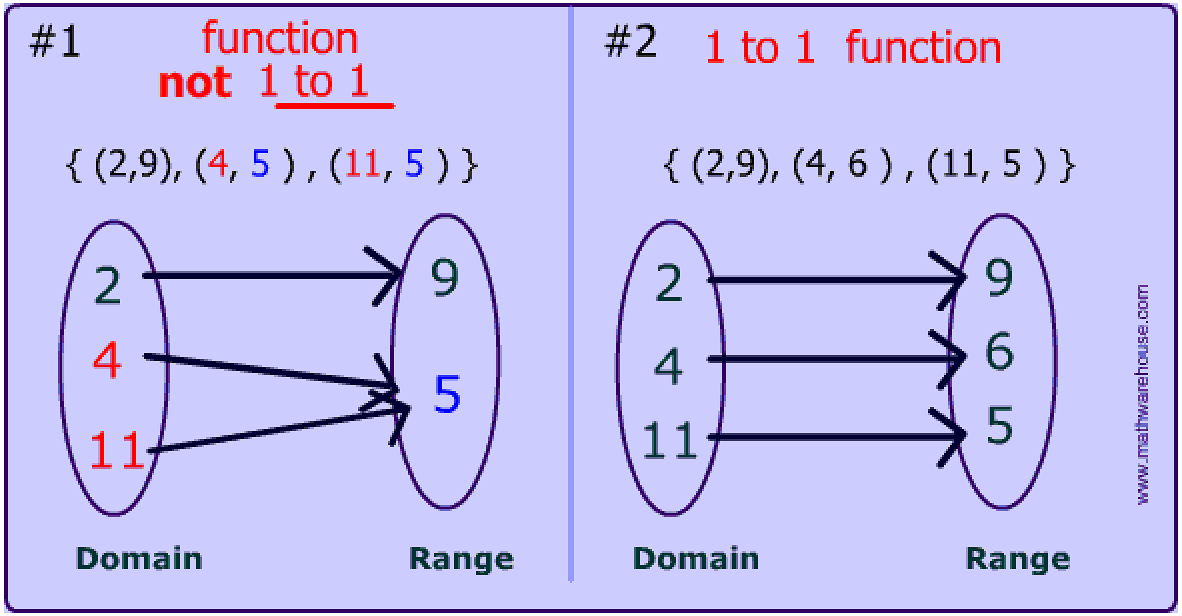
$$= (2x + 1)^2 - 2$$

$$= 4x^2 + 4x + 1 - 2$$

$$= 4x^2 + 4x - 1.$$

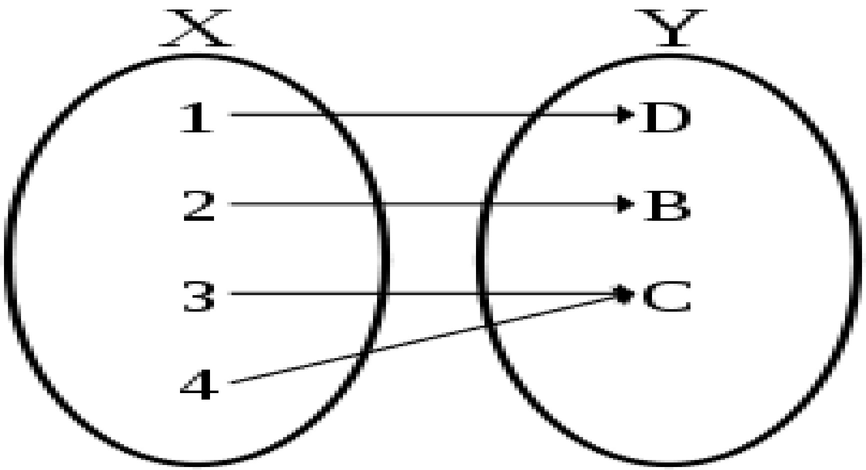
Types of functions

- **ONE-TO-ONE (Injective) Function:**
- A function *f : A → B is said to be one-to-one (written 1-1) if different elements in the domain A have* distinct images. Another way of saying the same thing is that *f is one-to-one if f (a) = f (a') implies a = a'*



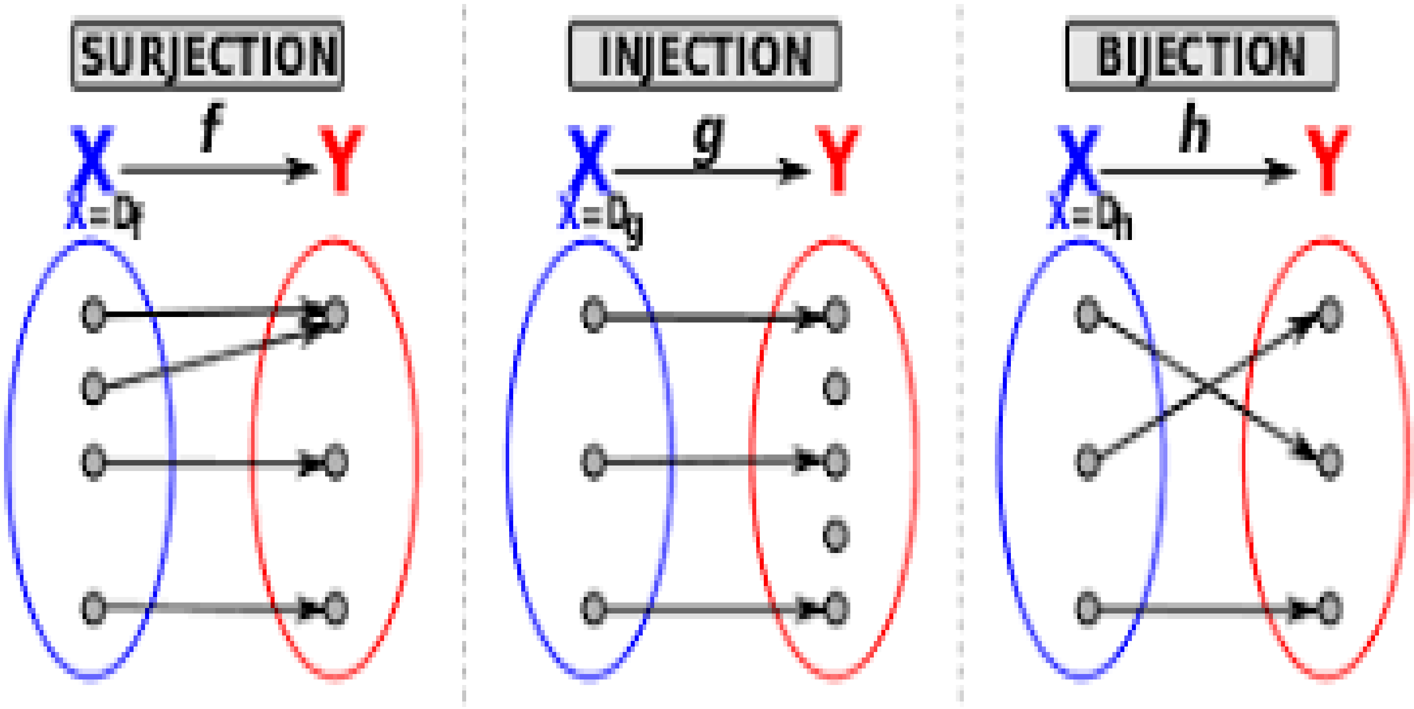
Types of functions

- **ONTO (Surjective) Function:**
- A function $f\colon X \rightarrow Y$ *is said to be an onto function if each element of* Y *is the image of some element of* X.
- In other words, $f : X \rightarrow Y$ *is onto if the image of* f *is the entire codomain, i.e., if* $f(X) = Y$.
- ***Range= Codomain***



Types of functions

- **1-1 AND ONTO (Bijective) Function:**
- A function which is both injective and bijective.



Types of functions

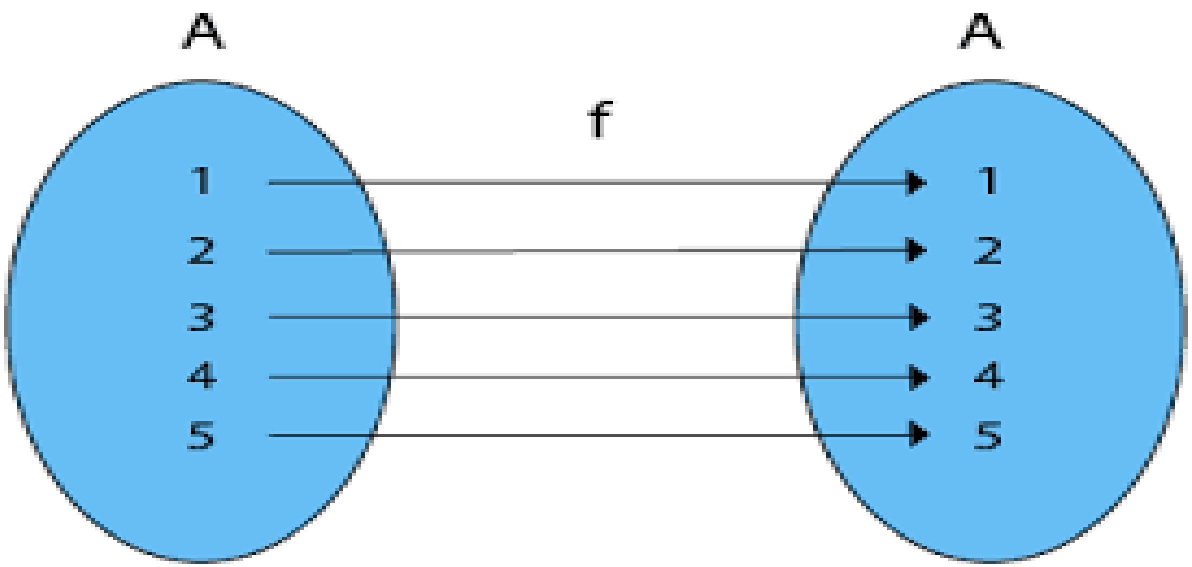
- **Invertible (Inverse) Function:**

- A function $f: A \to B$ *is invertible if its inverse relation* f ^(−1) is a function from B *to* A.
- *In general, the inverse* relation f^(−1) may not be a function.
- **A function *f: A → B is invertible if and only if f is both one-to-one and onto.***

TYPES OF FUNCTIONS

- **Identity Function:**

- A function f: $A \rightarrow A$ *is identity relation if* *$f(a) = a$ for all a belonging to A.*



RECURSIVELY DEFINED FUNCTIONS

- A function is said to be *recursively defined if the function definition refers to itself*.
- The function definition must have the following two properties:

  (1) There must be certain arguments, called *base values, for which the function does not refer to itself.*

  (2) Each time the function does refer to itself, the argument of the function must be closer to a base value

RECURSIVELY DEFINED FUNCTIONS

- **Factorial Function**
- The product of the positive integers from 1 to *n*, *inclusive, is called "n factorial" and is usually denoted by n!*.

  That is, $n! = n(n-1)(n-2) \cdot \cdot \cdot 3 \cdot 2 \cdot 1$

  - It is also convenient to define 0! = 1, so that the function is defined for all nonnegative integers.

  $0! = 1$, $1! = 1$, $2! = 2 \cdot 1 = 2$, $3! = 3 \cdot 2 \cdot 1 = 6$

  $4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$ $5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120$,

  $6! = 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 720$ *and so on..*
- This is true for every positive integer *n; that is,*

  $$\boldsymbol{n! = n \cdot (n-1)!}$$

RECURSIVELY DEFINED FUNCTIONS

- **(Factorial Function):**

(a) If $n = 0$, then $n! = 1$.

(b) If $n > 0$, then $n! = n \cdot (n - 1)!$

- the above definition of *$n!$ is recursive, since it refers to itself when it uses $(n - 1)!$. However:*
  - (1) The value of *$n!$ is explicitly given when $n = 0$ (thus 0 is a base value).*
  - (2) The value of *$n!$ for arbitrary n is defined in terms of a smaller value of n which is closer to the base* value 0.
- Accordingly, the definition is not circular, or, in other words, the function is well-defined.

RECURSIVELY DEFINED FUNCTIONS

- **Fibonacci Sequence**
- The celebrated Fibonacci sequence (usually denoted by $F_0, F_1, F_2, \ldots$) *is as follows:*

  $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$

  That is, $F_0 = 0$ *and* $F_1 = 1$ *and each succeeding term is the sum of the two preceding terms.*

  For example, the next two terms of the sequence are

  $34 + 55 = 89$ and $55 + 89 = 144$

RECURSIVELY DEFINED FUNCTIONS

- **(Fibonacci Sequence):**

  (a) If $n = 0$, or $n = 1$, then $F_n = n$.

  (b) If $n > 1$, then $F_n = F_{n-2} + F_{n-1}$.

- This is another example of a recursive definition, since the definition refers to itself when it uses F_{n-2} *and* F_{n-1}
- *However:*

  (1) The base values are 0 and 1.

  (2) The value of F_n *is defined in terms of smaller values of n which are closer to the base values.*

- Accordingly, this function is well-defined.

HASHING FUNCTIONS

- Hashing can be used to build, search, or delete from a table.
- The basic idea behind hashing is to take a field in a record, known as the **key**, and convert it through some fixed process to a numeric value, known as the **hash key**, which represents the position to either store or find an item in the table. The numeric value will be in the range of 0 to n-1, where n is the maximum number of slots (or **buckets**) in the table.
- The fixed process to convert a key to a hash key is known as a **hash function**. This function will be used whenever access to the table is needed.

HASHING FUNCTIONS

- **What is meant by Good Hash Function?**

  A good hash function should have the following properties:

1. Efficiently computable. (Easy and quick to complete)
2. Should uniformly distribute the keys (Each table position equally likely for each key)

- **3 Methods of Hashing**:

1. Division (MOD) Method
2. Mid-square Method
3. Universal or Folding Method

HASHING FUNCTIONS

- **The MOD (Division) Method**
- In this method for creating hash functions, we map a key into one of the slots of table by taking the remainder of key divided by table size.

  That is, the hash function is

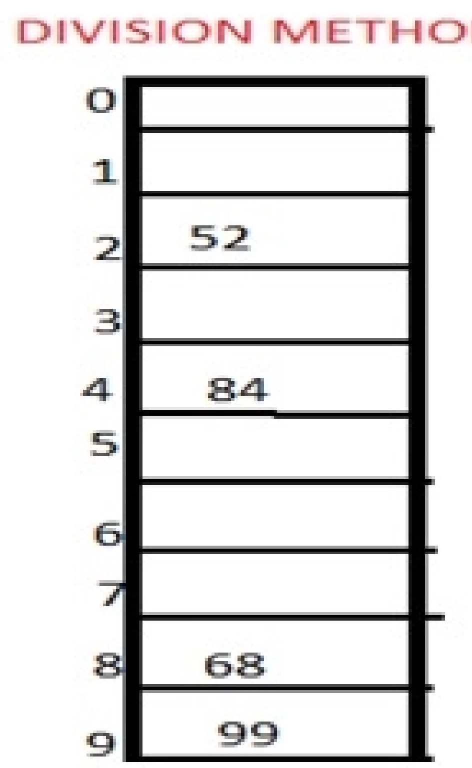
  h(key) = key **mod** table size = k (Mod M)

  i.e. key % table size = remainder when k is divided by M

  Using a prime number M reduces the number of collisions.

HASHING FUNCTIONS
THE MOD (DIVISION) METHOD

- For example:- If the records 52, 68, 99, 84 is to be placed in a hash table and let us take the table size is 10
- **Then:**
- h(key)=record% table size.
- 52%10 =2
- 68%10 =8
- 99%10 =9
- 84%10 =4



HASHING FUNCTIONS

- **Mid Square Method**

  In this method firstly key is squared and then mid part of the result is taken as the index.

  For example: consider that if we want to place a record of 3101 and the size of table is 1000.

  So 3101*3101=9616201

  i.e. **h (3101) = 162 (middle 3 digit)**

HASHING FUNCTIONS

- **Universal folding method**
- In this method the key is divided into separate parts and by using some simple operations these parts are combined to produce a hash key.
- For example: consider a record of 12465512 then it will be divided into parts i.e. 124, 655, 12.

  After dividing the parts combine these parts by adding it.

  H(key)=124+655+12 =791

HASHING FUNCTIONS
UNIVERSAL FOLDING METHOD

- The **folding method** for constructing hash functions begins by dividing the item into equal-size pieces (the last piece may not be of equal size).
- These pieces are then added together to give the resulting hash value.
- For example, if our item was the phone number 436-555-4601, we would take the digits and divide them into groups of 2 (43,65,55,46,01).

  After the addition, *[43+65+55+46+01]*, we get 210.

  If we assume our hash table has 11 slots, then we need to perform the extra step of dividing by 11 and keeping the remainder