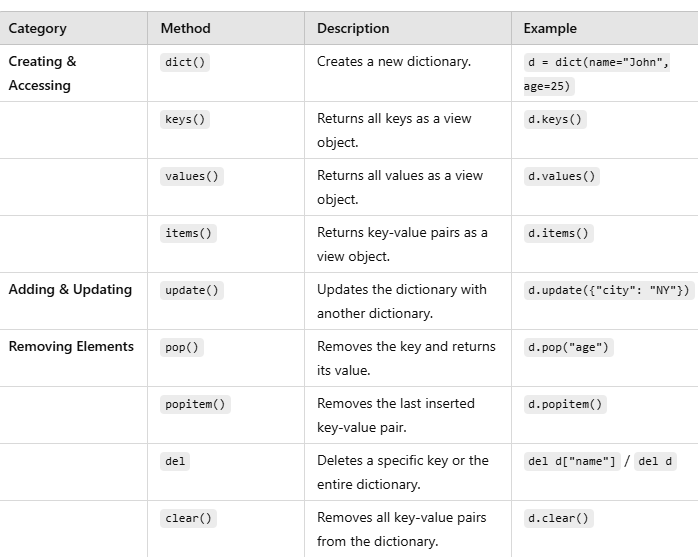
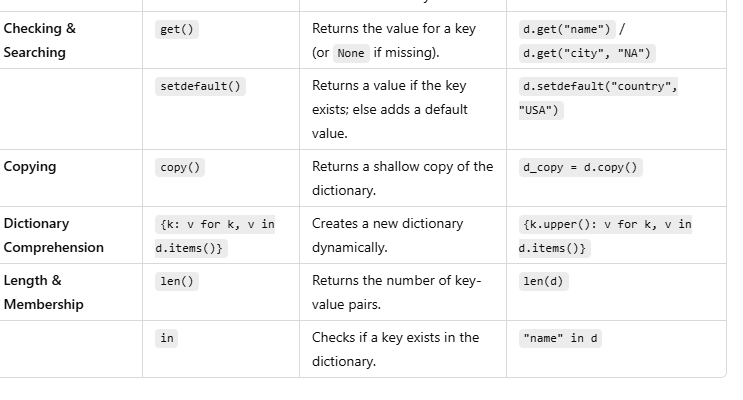
1. **Basic Program in Python**
2. Program or script which demonstrates the use of **different data types**
3. Program to make use of **following operators**
   1. Arithmetic Operations b. Logical Operations c. Relation operations
4. Program to make use **of If-else, el-if and nested if else loops** 
   1. Finding biggest of 3 numbers
   2. Even or odd
   3. Prime number
   4. Finding the grades of student
5. Program to make use **of while loop**
   1. Print the series from 1 to n
   2. Print the even and odd series
   3. Sum of natural numbers
   4. Armstrong number
   5. Palindrome
6. Program to make use of **for loop**
   1. Print the series from 1 to n
   2. Print the even and odd series
   3. Sum of natural numbers
   4. Armstrong number
   5. Palindrome
7. Program which demonstrates the use of function following
   1. No values passing and no parameters return
   2. Passing values and no parameters returns
   3. No passing values but return types
   4. Parameters passing and return types
8. Program to make use of **lists** and their operations or methods

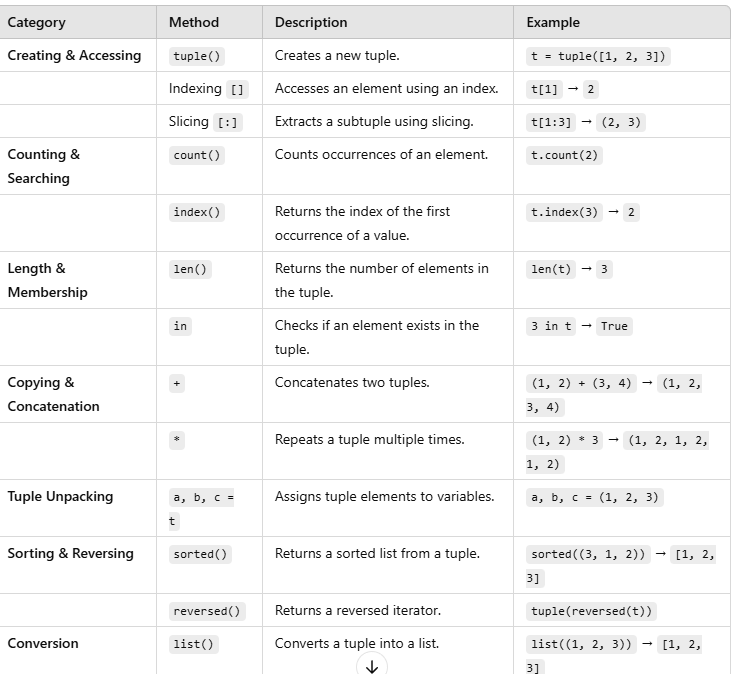


1. Program to make use of **dictionaries and their methods**

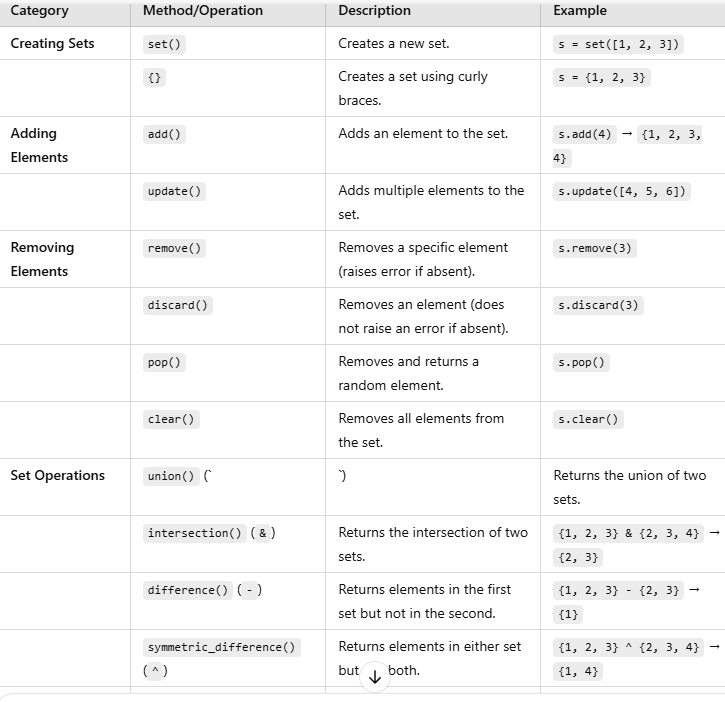
****

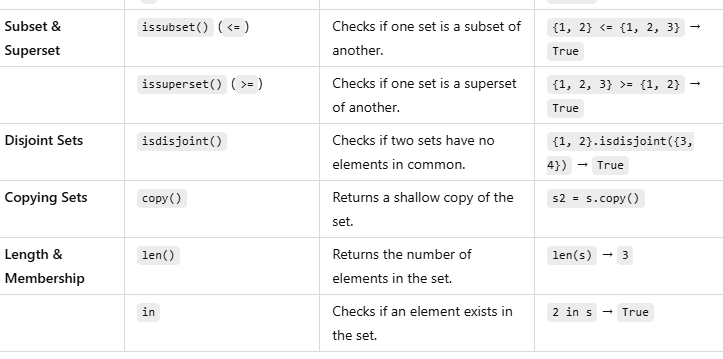
****

1. Program to make use **of tuples and their methods**



1. Program to work with **sets and their operations**

****

****

**24.02.2025**

**1. Array Creation & Properties**

* np.array() → Creates a NumPy array (1D, 2D, 3D)
* .shape → Returns shape (dimensions) of an array
* .size → Returns total number of elements
* .dtype → Returns data type of elements
* .ndim → Returns number of dimensions
* np.identity(n) → Creates an identity matrix of size n × n
* np.eye(n, m) → Creates a diagonal identity-like matrix

**2. Array Indexing & Slicing**

* arr[index] → Accesses element at a given index
* arr[row, col] → Accesses element in a 2D array
* arr[start:stop] → Slices a 1D array
* arr[row, :] → Slices entire row in 2D array
* arr[:, col] → Slices entire column in 2D array

**3. Reshaping & Flattening**

* .reshape(new\_shape) → Changes shape of an array
* .flatten() → Converts a multi-dimensional array into 1D

**4. Mathematical Operations**

**Element-wise Operations**

* np.square(arr) → Squares each element
* np.power(arr, n) → Raises each element to power n
* np.sqrt(arr) → Computes square root element-wise
* arr + value → Adds a scalar to each element
* arr \* value → Multiplies each element by a scalar

**Statistical & Aggregation Functions**

* np.sum(arr) → Computes sum of elements
* np.mean(arr) → Computes mean (average)
* np.median(arr) → Computes median
* np.std(arr) → Computes standard deviation
* np.var(arr) → Computes variance
* np.max(arr) → Returns max value
* np.min(arr) → Returns min value
* np.prod(arr) → Computes product of all elements
* np.percentile(arr, q) → Computes qth percentile
* np.mode(arr) → Computes mode (not directly in NumPy, use scipy.stats.mode)

**5. Linear Algebra Functions**

* .T → Transpose of a matrix
* np.linalg.inv(A) → Inverse of a matrix A
* np.linalg.det(A) → Determinant of matrix A
* np.linalg.matrix\_rank(A) → Rank of matrix A
* np.dot(A, B) → Matrix multiplication
* np.linalg.eig(A) → Computes eigenvalues and eigenvectors
  + Returns (eigenvalues, eigenvectors)
* np.linalg.svd(A) → Singular Value Decomposition (SVD)
  + Returns (U, S, V^T)

**6. Stacking & Concatenation**

* np.vstack((A, B)) → Vertically stacks arrays
* np.hstack((A, B)) → Horizontally stacks arrays

**7. Broadcasting**

* arr + scalar → Adds scalar to entire array
* arr2D + arr1D → Adds 1D array to 2D array (broadcasting)

**8. Universal Functions (ufuncs)**

**Arithmetic Functions**

* np.add(arr1, arr2) → Adds element-wise
* np.subtract(arr1, arr2) → Subtracts element-wise
* np.multiply(arr1, arr2) → Multiplies element-wise
* np.divide(arr1, arr2) → Divides element-wise

**Trigonometric Functions**

* np.sin(arr) → Sine of elements
* np.cos(arr) → Cosine of elements
* np.tan(arr) → Tangent of elements
* np.radians(degrees) → Converts degrees to radians

**Exponential & Logarithmic Functions**

* np.exp(arr) → Computes e^x
* np.log(arr) → Computes natural logarithm (ln)
* np.log10(arr) → Computes log base 10

**Rounding Functions**

* np.floor(arr) → Rounds down to nearest integer
* np.ceil(arr) → Rounds up to nearest integer
* np.round(arr, decimals=n) → Rounds to n decimal places

**9. Comparison Functions**

* np.greater(arr1, arr2) → Element-wise greater-than comparison
* np.less(arr1, arr2) → Element-wise less-than comparison
* np.equal(arr1, arr2) → Element-wise equality comparison

**10. Bitwise Operations**

* np.bitwise\_and(arr1, arr2) → Bitwise AND operation
* np.bitwise\_or(arr1, arr2) → Bitwise OR operation
* np.bitwise\_xor(arr1, arr2) → Bitwise XOR operation

**11. File Reading & Writing**

* np.loadtxt(filename, delimiter=",") → Loads data from a text file
* np.savetxt(filename, arr, delimiter=",") → Saves array to a text file
* np.save(filename, arr) → Saves array to a binary .npy file
* np.load(filename) → Loads an array from a .npy file

12. **Perform the following Boolean functions**

* np.all(arr1)
* np.any(arr1)
* np.where()

13. **Perform the following sorting methods**

* np.sort(arr1)
  + np.argsort(arr1)
  + np.unique(arr1)

**Summary of New Additions:**

✔ Identity Matrix (np.identity, np.eye)  
✔ Inverse Matrix (np.linalg.inv)  
✔ Eigenvalues & Eigenvectors (np.linalg.eig)  
✔ Mean, Median, Mode (np.mean, np.median, scipy.stats.mode)  
✔ Standard Deviation (np.std)  
✔ Singular Value Decomposition (SVD) (np.linalg.svd)  
✔ Rank of a Matrix (np.linalg.matrix\_rank)  
✔ Determinant (np.linalg.det)  
✔ Transpose (.T)  
✔ Sum, Square, Power, Square Root (np.sum, np.square, np.power, np.sqrt)  
✔ Dot Product (np.dot)  
✔ File Read/Write (np.loadtxt, np.savetxt, np.save, np.load)  
✔ Aggregation & Universal Functions

**PANDAS**

14. **Work with following functions in Series and Data frames using Pandas**

Function Description

df.sort\_values('col', ascending=True) Sorts by a column

df.sort\_values(['col1', 'col2']) Sorts by multiple columns

df.set\_index('col') Sets a column as index

df.reset\_index() Resets index

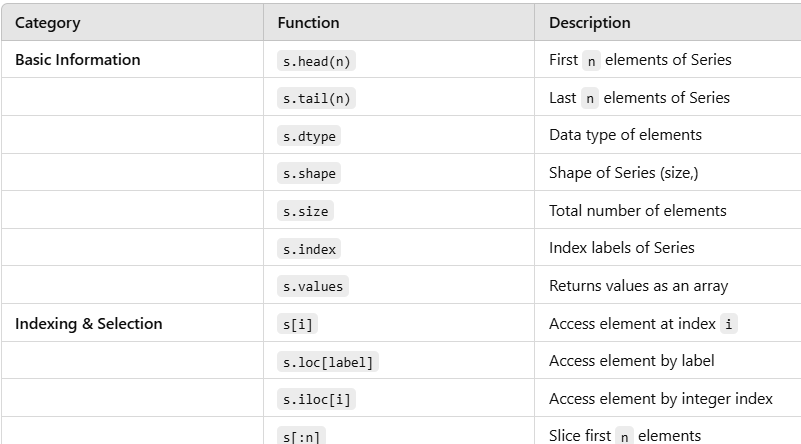
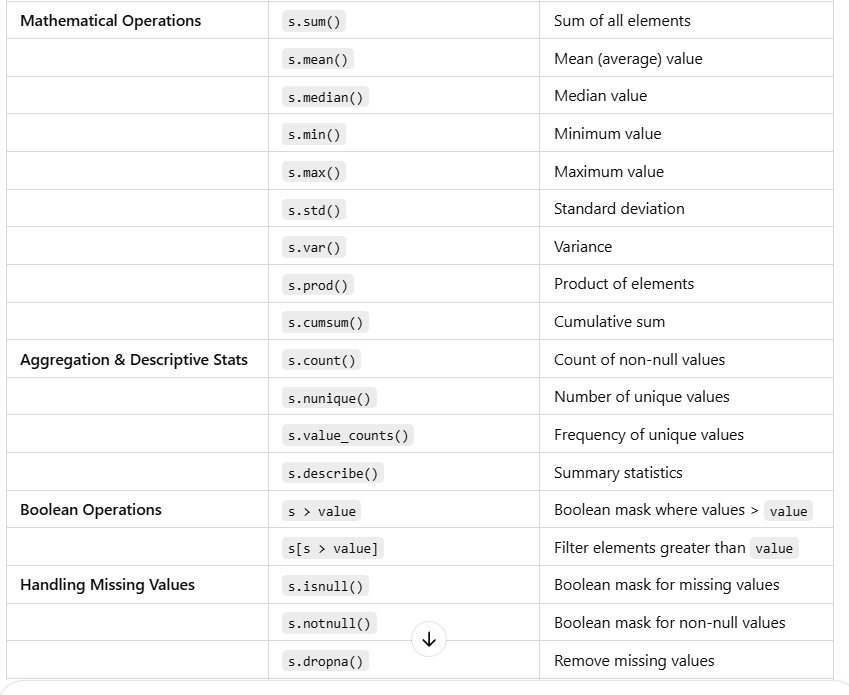
**15. Grouping & Aggregation**

Function Description

df.groupby('col').mean() Groups by column and calculates mean

df.groupby('col').agg(['mean', 'sum']) Performs multiple aggregations

df.pivot\_table(values='col1', index='col2') Creates a pivot table



**24/3/2025**

**16. Perform the following Operations on Multiple arrays**

a. Stack two arrays vertically

b. Stack two arrays horizontally

c. Get the common items between two python numpy arrays

d. Remove from one array those items that exist in another

e. Get the positions where elements of two arrays match

**17. Statistical analysis**

a. Compute the mean, median, standard deviation of a numpy array

b. Find the percentile scores of a numpy array

c. Compute the Euclidean distance between two arrays

d. Find the correlation between two columns of a numpy array

e. Probabilistic sampling in numpy

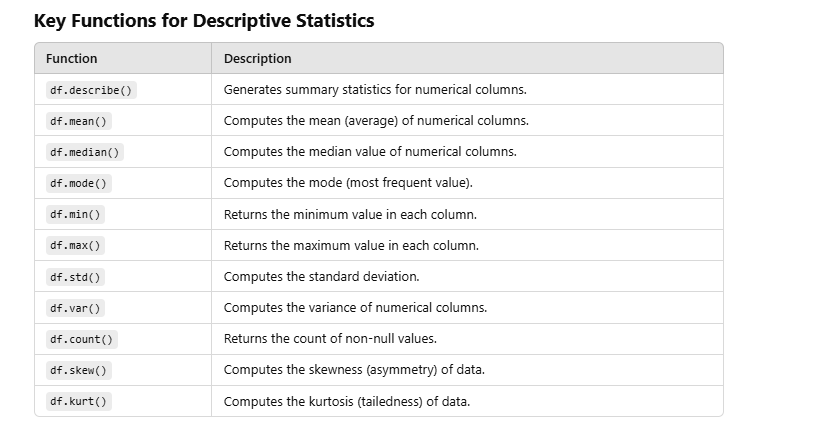
f. Compute the moving average of a numpy array

07.04.2025.

17 **Data Cleaning**

* 1. **Find the position of missing values in numpy array**
  2. **Drop rows that contain a missing value from a numpy array**
  3. **Replace all missing valueswith0inanumpyarray**
  4. **Drop all missing values from a numpy array**

**18. Perform the following descriptive functions**



19. Implement the following web API’s using post, get, put and update methods

20. Implement the following queries using SQLite

a. import library

b. connect to database

c. create a table

d. insert & display records

e. add a column

f. update, delete columns

g. drop the table and close

21. Implement the data transformation in python using Pandas

* Removing duplicates
* Adding a column
* Replacing values
* Renaming axis/index
* Discretization and binning
* Detecting and filtering outliers
* Permutation and random sampling

22. Implement String Manipulation Functions in Python

|  |  |  |
| --- | --- | --- |
| Method | Description | Example |
| upper() | Converts string to uppercase | 'hello'.upper() → 'HELLO' |
| lower() | Converts string to lowercase | 'HELLO'.lower() → 'hello' |
| title() | Capitalizes first letter of each word | 'hello world'.title() → 'Hello World' |
| strip() | Removes whitespace from both ends | ' text '.strip() → 'text' |
| replace() | Replaces parts of string | 'data'.replace('d', 'D') → 'Data' |
| split() | Splits string by separator into list | 'a,b,c'.split(',') → ['a', 'b', 'c'] |
| join() | Joins list elements with separator | '-'.join(['a','b']) → 'a-b' |
| find() | Returns index of first match or -1 | 'abc'.find('b') → 1 |
| count() | Counts occurrences of substring | 'banana'.count('a') → 3 |
| startswith() | Checks if string starts with prefix | 'hello'.startswith('he') → True |
| endswith() | Checks if string ends with suffix | 'hello.py'.endswith('.py') → True |
| zfill() | Pads string on the left with zeros | '42'.zfill(5) → '00042' |
| isalnum() | Checks if all characters are alphanumeric | 'abc123'.isalnum() → True |
| isspace() | Checks if string contains only whitespace | ' '.isspace() → True |
| format() | Inserts variables into string | 'Hello {}'.format('Alice') → 'Hello Alice' |
| f-string | Inline variable formatting | f'Hello {name}' → 'Hello Alice' |
| casefold() | Lowercase for caseless matching | 'Straße'.casefold() == 'strasse' → True |

23. D**ata Wra**ngling in Python: Join, Combine, and Reshape

# Hierarchical Indexing Combining and Merging Datasets Reshape

24. **Data Aggregation and Group Operations in Pandas**

25. Perform statistical analysis and visualize:

* Mean, median, standard deviation
* Histogram and box plot of the data