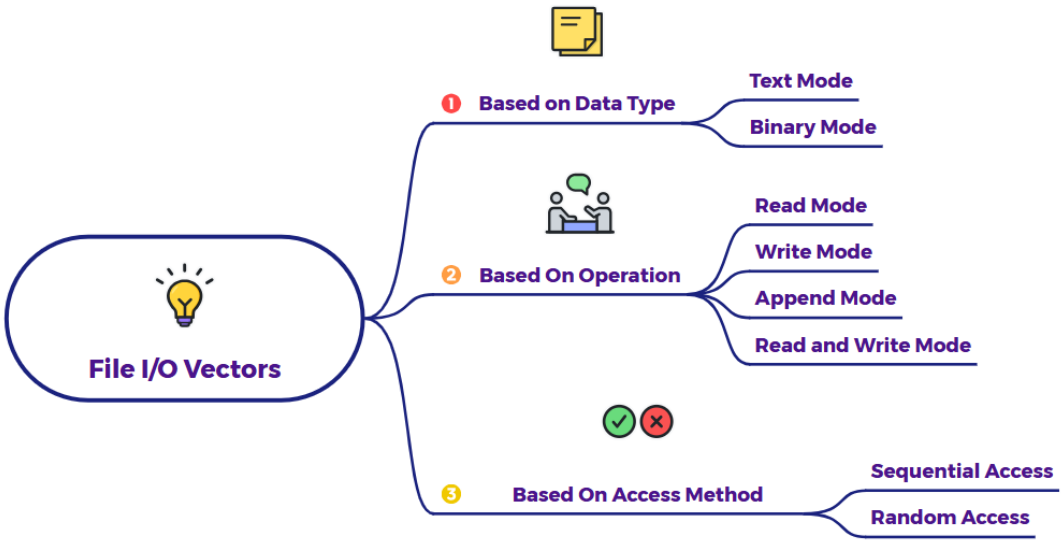
What are different file I/O vector of analysis ?



In Python, different types of file I/O primarily refer to **how you open and interact with the file**, which determines the mode of operation and how data is handled.

Here are the main distinctions and types:

1. **Based on Data Type (Text vs. Binary):**
    - **Text Mode ('t'):**
        - This is the **default mode** when you open a file.
        - Used for reading and writing **text data** (strings).
        - Python handles encoding and decoding of characters (e.g., UTF-8) automatically, converting raw bytes from the file into Python strings and vice-versa.
        - **Example:** open('my_document.txt', 'r') or open('log.csv', 'w')
    - **Binary Mode ('b'):**
        - Used for reading and writing **non-text (raw byte) data**.

- No encoding/decoding is performed. You work directly with bytes objects.
- Essential for files like images (.jpg, .png), audio (.mp3), video (.mp4), executables, or compressed files.
- **Example:** open ('my_image.jpg', 'rb') or open('new_data.bin', 'wb')

2. **Based on Operation (Read, Write, Append, etc.):**

These are specified by the **mode argument** when you call open ():

- **Read Mode ('r')**:
  - Opens the file for **reading only**.
  - The file pointer is placed at the beginning of the file.
  - Raises a FileNotFoundError if the file doesn't exist.
  - **Example:** f = open('data.txt', 'r')
- **Write Mode ('w')**:
  - Opens the file for **writing only**.
  - **Creates a new file** if it doesn't exist.
  - **Truncates (empties) the file** if it already exists (deletes all previous content).
  - The file pointer is placed at the beginning.
  - **Example:** f = open('output.txt', 'w')
- **Append Mode ('a')**:
  - Opens the file for **writing only**.
  - **Creates a new file** if it doesn't exist.
  - If the file exists, the file pointer is placed at the **end of the file**, so new content is added to the existing content.
  - **Example:** f = open('log.txt', 'a')
- **Read and Write Modes ('+')**:

- You can combine '+' with other modes to allow both reading and writing.
- **'r+' (Read and Write)**: Opens for both reading and writing. File pointer at the beginning. Does NOT create a new file if it doesn't exist.
- **'w+' (Write and Read)**: Opens for both reading and writing. **Creates a new file** or **truncates an existing one** first.
- **'a+' (Append and Read)**: Opens for both reading and writing. File pointer at the end for writing. Reading can occur from anywhere after seeking. Creates a new file if it doesn't exist.

3. **Based on Access Method (Sequential vs. Random):**

- **Sequential Access**: The most common way, where you read or write data from the beginning to the end of the file in order. This is the default behavior.
- **Random Access**: Using seek () and tell () methods on file objects, you can move the file pointer to any specific position within the file (for both reading and writing), allowing you to access or modify data at arbitrary locations.
    - f.tell(): Returns the current position of the file pointer.
    - f.seek(offset, whence): Changes the position of the file pointer.