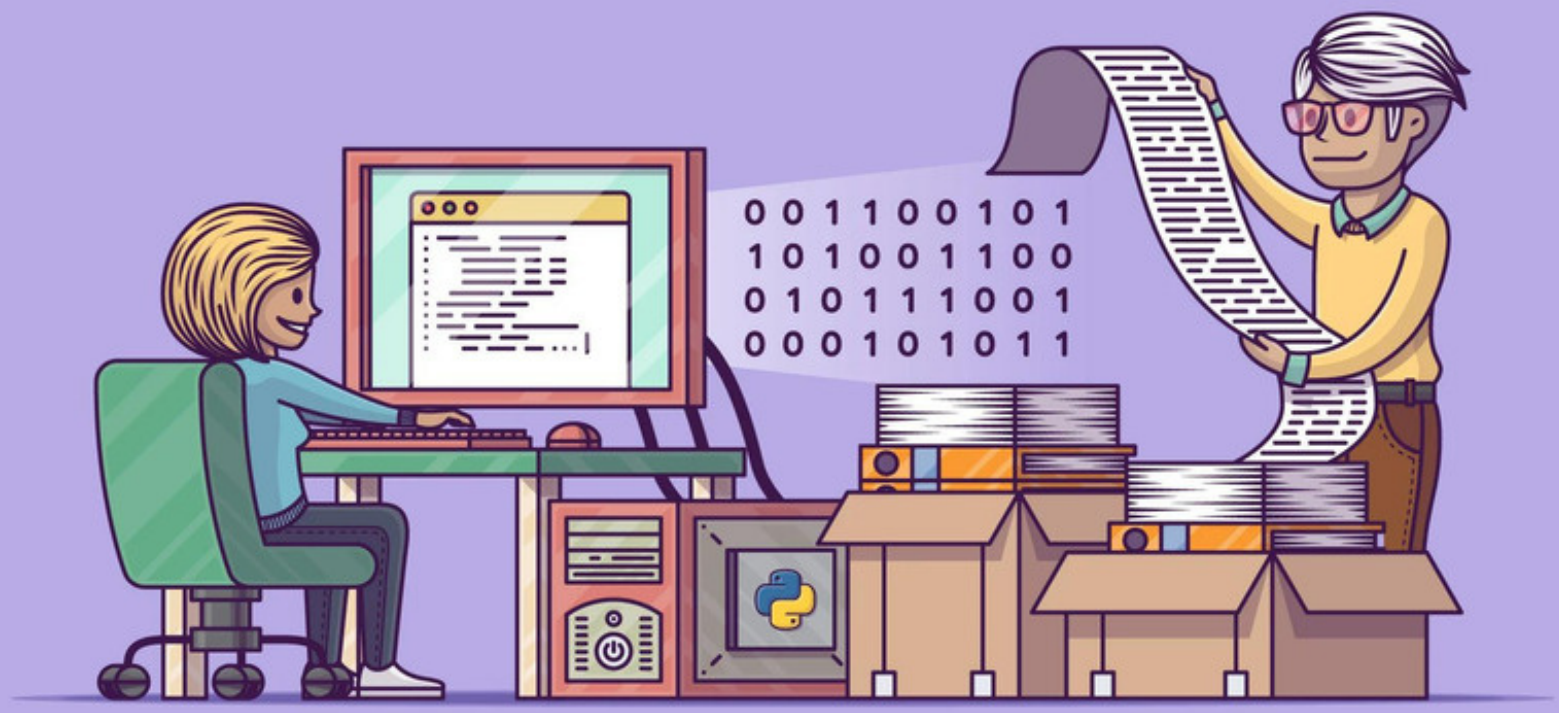
FILE READING AND WRITING



JSON METHOD

The json module in Python is used to work with JSON data. JSON stands for JavaScript Object Notation and is a popular data format used for representing structured data. It is commonly used to transmit and receive data between a server and web application in JSON format. It is also common to store a JSON object in a file.

The json module provides two methods, loads() and load(), that allow you to parse JSON strings and JSON files, respectively, to convert JSON into Python objects such as lists and dictionaries.

The json module and the open() function can be used together to read and write JSON data to files.

PYthon.py

```
import json

with open('data.json', 'r') as f:
        data = json.load(f)
```

CSV METHOD

The csv module in Python provides a number of methods for reading and writing CSV files. The most commonly used method is the reader() method, which returns a reader object that can be used to iterate over the lines in a CSV file. Each line in the CSV file is represented as a list of strings, where each string represents a field in the record.

Python.py

```
import csv

with open('data.csv', 'r') as csvfile:
    reader = csv.reader(csvfile)
    for row in reader:
        print(row)
```

IO METHOD

The io module in Python provides the main facilities for dealing with various types of I/O. There are three main types of I/O: text I/O, binary I/O, and raw I/O. These are generic categories, and various backing stores can be used for each of them.

The io module also provides a number of features that make it easier to perform I/O operations. Some of these features include:

- **Buffering: The io module provides buffering support, which can improve the performance of I/O operations.**
- **Encoding: The io module provides support for encoding and decoding text data.**
- **Error handling: The io module provides a number of error handling features that can be used to handle errors that occur during I/O operations.**
- **The flush() function in Python is used to flush the write buffer of a file object.**

Python.py

```
import io
with open("test1.txt" ,"wb") as f :
  file = io.BufferedWriter(f)
file.write(b"this is my second line\n")
  file.flush()
```

ANALYSIS

Analysis

Problem (P1)
- S1 — 100ms
- S2 — 50ms
- S3 — 30ms ⇐

1) Time Complexity
2) Space Complexity

Asymptotic Notations

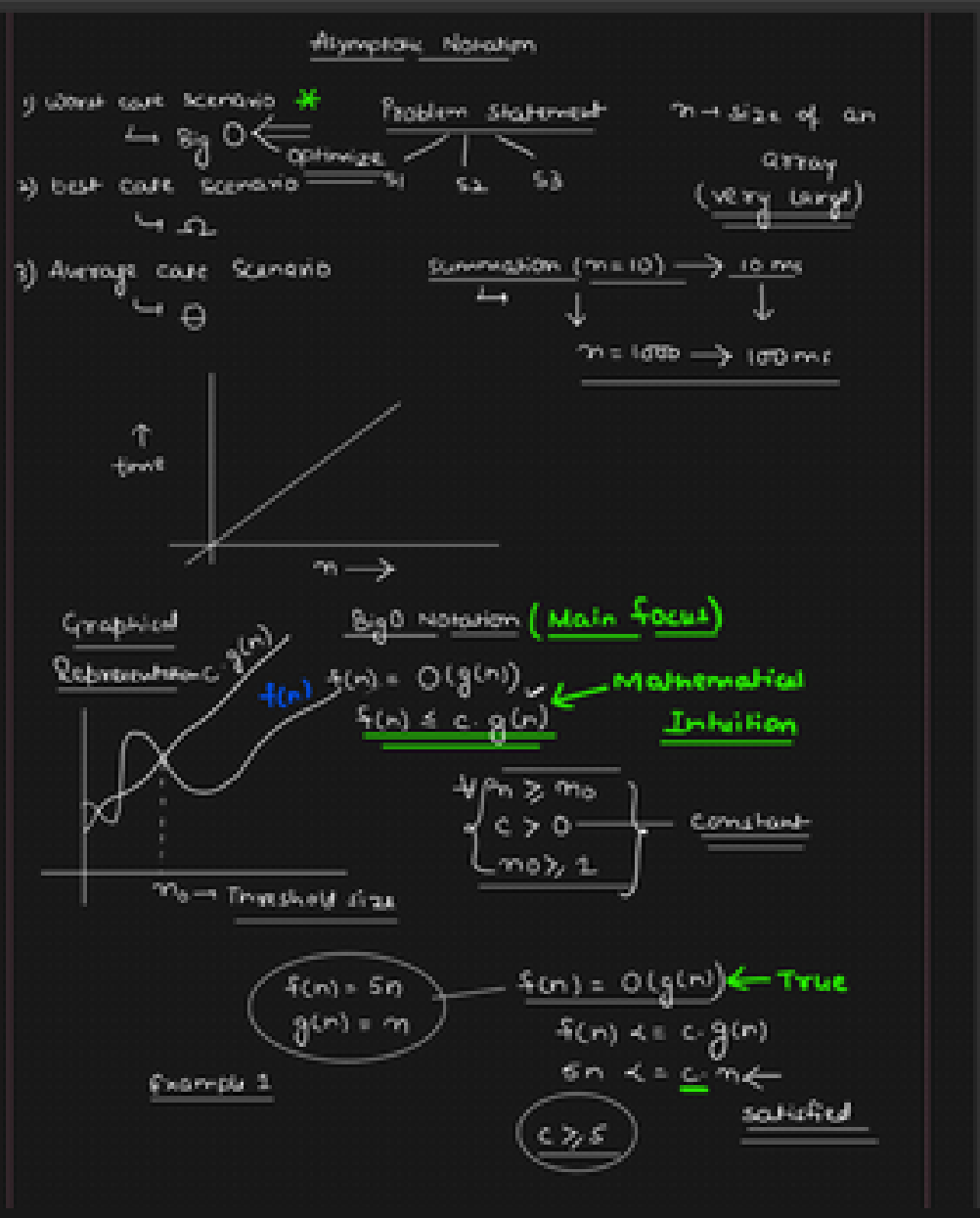
Types of analysis
- Apostiary Analysis
- Apriori *

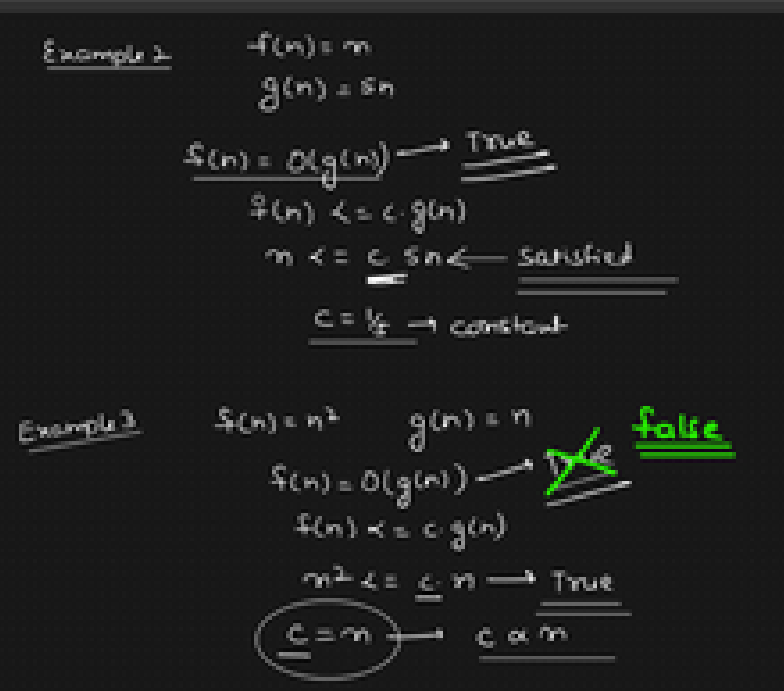
Apostiary → dependent on type of hardware & Language of compiler → Analysis

- → Exact answers
- → Different answers

Apriori → independent (compiler & hardware)
- → Approximate answers
- → Similar answers

ASYMPTOTIC NOTATION





FOLLOW FOR MORE

FOLLOW ME