**Lecture 1 – Object-Oriented Programming (OOP) Basics**

**Main idea:**  
OOP ka matlab hota hai code ko modular aur reusable banana. Matlab tum apne program ko chhote chhote parts (classes/objects) mein tod do, taki maintain karna easy ho aur duplication na ho.

**Key concepts:**

* **Class** → Blueprint/template hota hai (jaise "Car" ka design).
* **Object** → Us class ka actual instance (jaise tumhari gari, meri gari).
* **Abstraction** → Sirf zaroori cheezein dikhana, details hide karna.
* **Encapsulation** → Data aur methods ko ek bundle mein rakhna aur unnecessary access block karna.
* **Inheritance** → Purani class ka code reuse karke nayi class banana.
* **Polymorphism** → Same function/method different way se kaam kare depending on context.
* **Cohesion** → Ek module ka kaam ek hi clear purpose ho.
* **Coupling** → Modules ke beech dependency kam ho.

**Generative AI ke liye kaam ka:**

* Classes ka use model architectures banane mein hota hai (e.g., NeuralNetwork class).
* Abstraction/Encapsulation kaam aata hai jab tum AI pipeline ko modular banao.
* Inheritance se tum custom layers, datasets ya models bana sakte ho.

**Lecture 2 – Advanced OOP Features & Functions**

**Main idea:**  
Yahan tum function aur classes ke advance tools seekh rahe ho jo AI code ko flexible banate hain.

**Key concepts:**

* **\*args / \*\*kwargs** → Flexible arguments pass karne ka tarika (training functions mein useful).
* **Special methods** (magic methods) → e.g., \_\_init\_\_, \_\_str\_\_, \_\_add\_\_ etc.
* **Decorators (@)** → Functions ko modify karne ka clean way (training logs, preprocessing pipelines).
* **@staticmethod & @classmethod** → Class ke methods jo instance ki zarurat nahi rakhte.
* **Private variables** → Variable ke naam ke aage \_ ya \_\_ lagana to hide from outside.

**Generative AI ke liye kaam ka:**

* Decorators ka use training ke before/after hooks banane mein hota hai.
* \*args/\*\*kwargs AI frameworks (TensorFlow, PyTorch) mein bahut hota hai jab parameters dynamic ho.
* Special methods kaam aate hain jab tum apna AI model class bana rahe ho.

**Lecture 3 – Property, Inheritance, Iterators, Generators**

**Main idea:**  
Ye tumhare OOP ko aur flexible aur efficient banata hai.

**Key concepts:**

* **Getters & Setters / @property** → Variables ko control ke saath set/get karna.
* **Inheritance + super()** → Parent class ka code reuse karna aur override karna.
* **Lambda** → Small one-line functions (fast prototyping).
* **List comprehension** → Short way to make lists.
* **Iterators & Generators** → Memory efficient looping (dataset loading mein useful).

**Generative AI ke liye kaam ka:**

* **Generators** ka direct use data pipelines mein hota hai (lazy loading of training data).
* **List comprehension & lambda** ka use data transformation fast banata hai.
* **@property** ka use model parameters control karne ke liye hota hai (e.g., learning rate).

**Short Summary for Generative AI**

Tumhein in topics se mainly ye master karna hoga:

1. **Classes & Inheritance** – AI models ko modular banane ke liye.
2. **Decorators** – Training/prediction hooks.
3. **Generators** – Large dataset efficiently load karne ke liye.
4. **\*args & \*\*kwargs** – Flexible function design.
5. **Special methods (\_\_init\_\_, \_\_str\_\_, etc.)** – Custom model behavior define karne ke liye.
6. **Encapsulation & Abstraction** – AI code ko clean aur reusable rakhne ke liye

**🔑 Main Topics Jo Tumhe Generative AI ke liye aane chahiye**

**1. Error Handling (try-except-finally)**

* **Kya hai:** Code run karte waqt error aajaye to program crash na ho, balki gracefully handle kare.
* **Use in AI:** Jab tum models train karte ho ya files read/write karte ho, errors aate hain (missing file, GPU error, etc).
* **Example:**

try:

x = 10 / 0

except ZeroDivisionError:

print("Error: divide by zero")

finally:

print("Training continue karega, crash nahi hoga")

**2. Regular Expressions (RegEx)**

* **Kya hai:** String search aur pattern matching ka tool.
* **Use in AI:** Text cleaning, email validation, token filtering. NLP mein bahut use hota hai.
* **Example:**

import re

txt = "The rain in Spain"

x = re.findall("ai", txt) # ['ai', 'ai']

👉 Jab tum raw text data ko preprocess karte ho, regex must hai.

**3. File I/O (Input/Output)**

* **Kya hai:** File read/write karna.
* **Use in AI:** Dataset load karna (CSV, JSON, images), results save karna.
* **Example:**

f = open("data.txt", "w")

f.write("AI data saved")

f.close()

**4. Data Structures (List, Stack, Queue, Linked List, Graphs, Trees)**

* **Kya hai:** Data ko organize karne ka tareeqa.
* **Use in AI:** Efficiently data manage karna (training batches, graphs, backpropagation).
* **Example:**

stack = []

stack.append("image1")

stack.append("image2")

print(stack.pop()) # LIFO: image2

👉 Neural networks ko implement karne ke liye graphs aur linked lists ka concept indirectly use hota hai.

**5. Big O Notation (Time/Space Complexity)**

* **Kya hai:** Code kitna fast hai aur kitna memory khata hai.
* **Use in AI:** Large datasets handle karne ke liye efficient code likhna zaroori hai.
* **Example:**
  + O(1) → Fastest (dictionary lookup).
  + O(n^2) → Bahut slow jab dataset bada ho.

**6. Pointers & Mutability**

* **Kya hai:** Variable ya object ka memory address aur uska behavior.
* **Use in AI:** Jab tum data ko copy ya share karte ho between functions, samajhna zaroori hai ki data copy hua hai ya reference gaya hai.

**7. Linked List Implementation**

* **Kya hai:** Sequence of nodes (each node points to next).
* **Use in AI:** Memory efficient structures, dynamic data handling, computational graph (PyTorch ke andar).
* **Example (simplified):**

class Node:

def \_\_init\_\_(self, value):

self.value = value

self.next = None

n1 = Node(5)

n2 = Node(10)

n1.next = n2

print(n1.next.value) # 10

**8. Algorithms with Data Structures**

* Insert, delete, search, update → Ye sab linked list, stack, queue, graph mein seekhna important hai.
* **AI mein use:** Training data batches ko efficiently load karna, model graphs banane.

**🚀 Agar Generative AI karna hai to:**

1. **Error handling aur File I/O** → Dataset smoothly handle karne ke liye.
2. **Regex** → Text preprocessing (NLP ke liye).
3. **Data Structures (Lists, Stacks, Queues)** → Training pipeline design karne ke liye.
4. **Big O Notation** → Efficiency maintain karne ke liye.
5. **Linked Lists aur Pointers** → Deep understanding of frameworks like PyTorch / TensorFlow.

**Lecture 7 (Singly & Doubly Linked Lists)**

Ye lecture **Linked List** aur **Doubly Linked List** ke baare mein hai. Is ko simple words mein samjho:

* **Linked List:** Imagine karo ke aap ke paas kuch dabba hain (nodes), aur har dabbe ke andar agle dabbe ka address likha hua hai. Pehla dabba apnay next dabbe ko point karta hai aur woh agle ko, isi tarah ye chain chalti rehti hai. Is ko

**Singly Linked List** kehte hain. Is mein aap sirf aage hi ja sakte hain. Ye data store karne ka ek tareeqa hai, jo tab kaam aata hai jab aap ko bar bar data add ya delete karna ho.

* + **Remove:** Agar koi dabba remove karna ho to aap ko us se pehle walay dabbe ko aakhri dabbe se link karna hoga aur remove hone walay dabbe ko wahan se hata dena hoga.
  + **Reverse:** Linked list ko ulta (reverse) karne ka tareeqa bhi bataya gaya hai.
* **Doubly Linked List:** Ye bhi waise hi dabbo ki chain hai, lekin is mein thoda extra faida hai. Har dabbe mein agle dabbe ka address to hai hi, saath mein pichle dabbe ka address bhi hai. Is se aap aage aur peeche dono directions mein jaa sakte hain. Is ka faida ye hai ke nodes ko delete karna aur list ko reverse karna asaan ho jata hai. Is ko web browsers mein "back" aur "forward" buttons ke liye bhi use kiya jata hai.

**Lecture 8 (Stack, Queue & Trees)**

Is lecture mein teen aur data structures hain: **Stack, Queue, aur Trees**.

* **Stack:** Is ko ek tennis balls ke container ki tarah samjho. Jo ball aakhri mein dali jaayegi, woh sab se pehle nikaali jaayegi. Is ko

**Last-In/First-Out (LIFO)** kehte hain. Misal ke taur par, aap jab browser mein websites visit karte hain, to pichli sites ek stack mein save hoti hain. Jab aap "back" button dabate hain, to sab se aakhri visited website pehle aati hai.

* **Queue:** Ye ek line ki tarah hai, jaise cinema ki ticket line. Jo pehle aayega, woh pehle ticket lega. Is ko

**First-In/First-Out (FIFO)** kehte hain.

**Enqueue** ka matlab line mein aana aur **Dequeue** ka matlab line se bahar nikalna hai.

* **Trees:** Ye data ko ek hierarchy (darjat) mein rakhta hai, bilkul ek family tree ki tarah. Sab se upar

**root node** hota hai. Phir us ke

**children nodes** hote hain aur aage ye silsila chalta rehta hai.

* + **Binary Search Tree (BST):** Ye aik special qism ka tree hai. Is mein har node ke 2 se zyada children nahi hote. Is ka faida ye hai ke aap is mein values ko asani se dhundh sakte hain, kyun ke choti values left side par aur bari values right side par hoti hain.

**Generative AI Ke Liye Python Ki Zaroori Cheezein**

Generative AI ke liye aap ko in cheezon par focus karna chahiye:

1. **NumPy:** Ye basic math calculations ke liye use hota hai. AI models mein arrays aur matrices par calculations karni hoti hain, aur NumPy is kaam ke liye sab se best hai.
2. **Pandas:** Agar aap ko data ko samajhna aur saaf karna hai, to is ki zaroorat padegi. Is se aap spreadsheets ki tarah data ko organize aur analyze kar sakte hain.
3. **Matplotlib & Seaborn:** AI mein data visualization bohat important hai. Aap ko dekhna hota hai ke data kaisa hai. Ye dono libraries data ko graphs aur charts ki shakal mein dikhane ke kaam aati hain.
4. **Scikit-learn:** Ye machine learning ke basic algorithms ke liye hai. Is se aap AI ki aam cheezein, jese data ko alag alag karna ya predict karna seekh sakte hain.
5. **PyTorch ya TensorFlow:** Ye Generative AI ka core hai. Bade bade models, jese ke **Large Language Models (LLMs)**, in platforms par banaye jaate hain. In mein deep learning ke liye tools hote hain, jo Generative AI ke liye sab se zaroori hain.
6. **Hugging Face Transformers:** Generative AI mein jo models pre-trained hote hain, un ko use karne ke liye Hugging Face sab se best platform hai. Is se aap asani se models download kar ke un se text generate kar sakte hain ya images bana sakte hain.