আসুন প্রোগ্রামিং এর কিছু বেসিক কনসেপ্ট গুলোকে সহজ এবং মজাদার উদাহরণের মাধ্যমে বুঝতে চেষ্টা করি। এখানে অবজেক্ট, ক্লাস, এনক্যাপসুলেশন, ইনহেরিট্যান্স, পলিমরফিজম, অ্যাবস্ট্রাকশন, অ্যাসোসিয়েশন, অ্যাগ্রিগেশন এবং কম্পোজিশন এর উদাহরণ দেওয়া হলো:

**1. অবজেক্ট (Object)**

**উদাহরণ:**  
একটি বিড়াল একটি অবজেক্ট। এর কিছু প্রপার্টি আছে যেমন: রং, ওজন, বয়স। এবং এর কিছু মেথড আছে যেমন: মিউ মিউ করা, ঘুমানো, খেলা।

python

Copy

class Cat:

def \_\_init\_\_(self, color, weight, age):

self.color = color

self.weight = weight

self.age = age

def meow(self):

return "Meow!"

def sleep(self):

return "Zzzz..."

def play(self):

return "Playing with a ball of yarn!"

# অবজেক্ট তৈরি

my\_cat = Cat("Black", 4.5, 2)

print(my\_cat.meow()) # আউটপুট: Meow!

**2. ক্লাস (Class)**

**উদাহরণ:**  
একটি ক্লাস হলো একটি ব্লুপ্রিন্ট বা টেমপ্লেট। যেমন, একটি "বিড়াল" ক্লাস তৈরি করা হলো যার মাধ্যমে আমরা অনেক বিড়াল অবজেক্ট তৈরি করতে পারি।

python

Copy

class Cat:

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

self.name = name

def speak(self):

return f"{self.name} says Meow!"

# ক্লাস থেকে অবজেক্ট তৈরি

cat1 = Cat("Whiskers")

cat2 = Cat("Mittens")

print(cat1.speak()) # আউটপুট: Whiskers says Meow!

print(cat2.speak()) # আউটপুট: Mittens says Meow!

**3. এনক্যাপসুলেশন (Encapsulation)**

**উদাহরণ:**  
এনক্যাপসুলেশন মানে ডাটা এবং মেথড গুলোকে একসাথে রাখা এবং ডাটা গুলোকে প্রাইভেট রাখা। যেমন, একটি ব্যাংক অ্যাকাউন্ট ক্লাসে ব্যালেন্স প্রাইভেট রাখা।

python

Copy

class BankAccount:

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

self.\_\_balance = balance # প্রাইভেট ভেরিয়েবল

def deposit(self, amount):

self.\_\_balance += amount

def withdraw(self, amount):

if amount <= self.\_\_balance:

self.\_\_balance -= amount

else:

return "Insufficient funds!"

def get\_balance(self):

return self.\_\_balance

# অবজেক্ট তৈরি

account = BankAccount(1000)

account.deposit(500)

print(account.get\_balance()) # আউটপুট: 1500

account.withdraw(200)

print(account.get\_balance()) # আউটপুট: 1300

**4. ইনহেরিট্যান্স (Inheritance)**

**উদাহরণ:**  
ইনহেরিট্যান্স মানে একটি ক্লাস অন্য ক্লাসের বৈশিষ্ট্য উত্তরাধিকার সূত্রে পায়। যেমন, একটি "ডগ" ক্লাস "অ্যানিমাল" ক্লাস থেকে ইনহেরিট করে।

python

Copy

class Animal:

def speak(self):

return "Some generic sound"

class Dog(Animal):

def speak(self):

return "Woof!"

# অবজেক্ট তৈরি

dog = Dog()

print(dog.speak()) # আউটপুট: Woof!

**5. পলিমরফিজম (Polymorphism)**

**উদাহরণ:**  
পলিমরফিজম মানে একই মেথডের বিভিন্ন রূপ। যেমন, "অ্যানিমাল" ক্লাসের "speak" মেথড বিভিন্ন প্রাণীর জন্য বিভিন্ন আউটপুট দেয়।

python

Copy

class Cat:

def speak(self):

return "Meow!"

class Dog:

def speak(self):

return "Woof!"

# পলিমরফিজম

animals = [Cat(), Dog()]

for animal in animals:

print(animal.speak())

# আউটপুট:

# Meow!

# Woof!

**6. অ্যাবস্ট্রাকশন (Abstraction)**

**উদাহরণ:**  
অ্যাবস্ট্রাকশন মানে জটিল ডিটেইলস গুলোকে লুকিয়ে রাখা এবং শুধুমাত্র প্রয়োজনীয় ফিচার গুলো দেখানো। যেমন, একটি "কার" ক্লাসে ড্রাইভ মেথড আছে কিন্তু কিভাবে ইঞ্জিন কাজ করে তা লুকানো।

python

Copy

from abc import ABC, abstractmethod

class Vehicle(ABC):

@abstractmethod

def drive(self):

pass

class Car(Vehicle):

def drive(self):

return "Driving a car!"

# অবজেক্ট তৈরি

car = Car()

print(car.drive()) # আউটপুট: Driving a car!

**7. অ্যাসোসিয়েশন (Association)**

**উদাহরণ:**  
অ্যাসোসিয়েশন মানে দুটি ক্লাসের মধ্যে সম্পর্ক। যেমন, একটি "স্টুডেন্ট" ক্লাস এবং একটি "কোর্স" ক্লাসের মধ্যে সম্পর্ক।

python

Copy

class Student:

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

self.name = name

class Course:

def \_\_init\_\_(self, course\_name):

self.course\_name = course\_name

# অ্যাসোসিয়েশন

student = Student("Alice")

course = Course("Math")

print(f"{student.name} is enrolled in {course.course\_name}") # আউটপুট: Alice is enrolled in Math

**8. অ্যাগ্রিগেশন এবং কম্পোজিশন (Aggregation and Composition)**

**উদাহরণ:**

* **অ্যাগ্রিগেশন:** একটি "লাইব্রেরি" ক্লাসে অনেক "বই" অবজেক্ট থাকতে পারে, কিন্তু বই গুলো লাইব্রেরির বাইরেও থাকতে পারে।
* **কম্পোজিশন:** একটি "হাউস" ক্লাসে অনেক "রুম" অবজেক্ট থাকতে পারে, কিন্তু রুম গুলো হাউসের বাইরে থাকতে পারে না।

python

Copy

# অ্যাগ্রিগেশন

class Book:

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

self.title = title

class Library:

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

self.books = []

def add\_book(self, book):

self.books.append(book)

# অবজেক্ট তৈরি

book1 = Book("Python 101")

book2 = Book("Advanced Python")

library = Library()

library.add\_book(book1)

library.add\_book(book2)

for book in library.books:

print(book.title)

# আউটপুট:

# Python 101

# Advanced Python

# কম্পোজিশন

class Room:

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

self.name = name

class House:

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

self.rooms = [Room("Living Room"), Room("Bedroom")]

# অবজেক্ট তৈরি

house = House()

for room in house.rooms:

print(room.name)

# আউটপুট:

# Living Room

# Bedroom

এই উদাহরণ গুলো প্রোগ্রামিং এর বেসিক কনসেপ্ট গুলোকে সহজ এবং মজাদার ভাবে বুঝতে সাহায্য করবে।