Q21. X-O Game using OOP Python

https://youtu.be/GCYYkOSKj80?si=xf11YzpsG7eM1Jx-

==========

Q2 . Summarize DataClass in Python

https://youtu.be/HJkY_Bbiqcc?si=HdWG3CHaUANtw2NU

==========

Q3. Summarize Multi Inheritance in Python

Method Resolution Order (MRO)

==========

Q4. Dictionary Comprehension Example

==========

Q5. Composition

Create a class called Engine with attributes:

fuel_type (e.g., "petrol", "diesel", "electric")

horsepower

Modify the Car class to include an Engine object as an attribute. Update the Car class's __init__ method to accept an Engine object.

Create a Car instance with an Engine and test its functionality.

==========

Q6: Implement a Vector Class with Dunder Methods

Create a Vector class that represents a 2D vector with x and y components. Implement the following dunder methods to support common operations:

__init__: Initialize the vector with x and y components.

__repr__: Return a string representation of the vector in the format "Vector(x, y)".

__add__: Add two vectors component-wise and return a new Vector instance.

__sub__: Subtract one vector from another component-wise and return a new Vector instance.

__mul__: Multiply the vector by a scalar (number) and return a new Vector instance.

__eq__: Compare two vectors for equality (return True if both components are equal).

__len__: Return the magnitude (length) of the vector as an integer (rounded using round()).

__getitem__: Allow accessing components using indexing (e.g., v[0] for x and v[1] for y).

Example :

```
v1 = Vector(2, 4)
v2 = Vector(3, 1)

print(v1)           # Output: Vector(2, 4)
print(v1 + v2)      # Output: Vector(5, 5)
print(v1 - v2)      # Output: Vector(-1, 3)
print(v1 * 3)       # Output: Vector(6, 12)
print(v1 == Vector(2, 4)) # Output: True
print(len(v1))      # Output: 4 (magnitude of Vector(2, 4) is ~4.47, rounded to 4)
print(v1[0])        # Output: 2 (x component)
print(v1[1])        # Output: 4 (y component)
```