Software Engineering Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. (Fundamental British Values: Mutual Respect and Tolerance)

- No question is daft or silly - **ask them!**

- There are **Q&A sessions** throughout this session, should you wish to ask any follow-up questions.

- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: [Questions](#)

Software Engineering Session Housekeeping cont.

- For all **non-academic questions**, please submit a query:
  www.hyperiondev.com/support

- Report a **safeguarding** incident:
  www.hyperiondev.com/safeguardreporting

- We would love your **feedback** on lectures: Feedback on Lectures

CoGrammar

Enhancing Accessibility: Activate Browser Captions

<u>Why Enable Browser Captions</u>?

- Captions provide **real-time text for spoken content**, ensuring inclusivity.

- Ideal for individuals in noisy or quiet environments or for those with **hearing impairments**.

<u>How to Activate Captions</u>:

1. YouTube or Video Players:

   - Look for the CC (Closed Captions) icon and click to enable.

2. Browser Settings:

   - Google Chrome: Go to *Settings > Accessibility > Live Captions* and toggle ON.

   - Edge: Enable captions in *Settings > Accessibility*.

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar   HyperionDev

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar

Classes and Methods

Learning Outcomes

- Define a class and create instances (objects) of that class.
- Define and use attributes and methods in a class.
- Write constructors to set initial values for object attributes.
- Implement encapsulation by using private attributes and providing public methods to access and modify them.

CoGrammar

Learning Outcomes

- Define and access class attributes.
- Define and access instance attributes.
- Define and call static methods using the @staticmethod decorator .
- Define and call class methods using the @classmethod decorator.

CoGrammar

Classes

CoGrammar

Classes

- Classes are blueprints for creating objects. They define the properties and behaviours that objects of the class will have.

- Classes encapsulate data (attributes) and functionality (methods) into a single unit, facilitating code organisation and reuse.

CoGrammar

Classes...

```python
# Define the Car class
class Car:
    def __init__(self, brand, color):
        self.brand = brand
        self.color = color

    def drive(self):
        return f"The {self.color} {self.brand} is driving."
```

CoGrammar

Attributes

- Attributes represent the state or characteristics of objects. They are the data associated with instances of the class and define what an object of that class looks like.

- Attributes can be variables that store data (instance variables) or methods (instance methods) that define behaviours.

```python
# Define the Car class
class Car:
    def __init__(self, brand, color):
        self.brand = brand
        self.color = color
```

Methods

- Methods are functions defined within a class and they define the behaviours or actions that objects or instances of the class can perform.

- Methods operate on the data (attributes) associated with the class and provide the functionality to manipulate that data.

- Methods can be instance methods, static methods, or class methods.

```python
class Car:
    def drive(self):
        print("The car is driving.")
```

CoGrammar

Objects

- An object is an instance of a class. It is a concrete realisation of the class blueprint, possessing its own unique set of attributes and methods.

- When you create an object, you are essentially creating a specific instance of that class with its own data and behaviour.

```python
# Create an object (instance) of the Car class
my_car = Car("Toyota", "red")
```

CoGrammar

Objects...
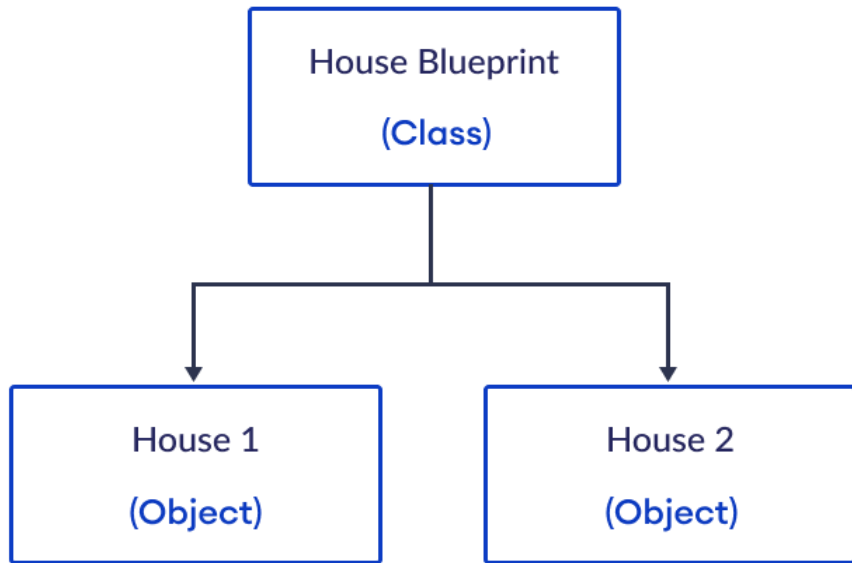
Access Control

Encapsulation in OOP

- Encapsulation is the concept of bundling the data (attributes) and behaviour (methods) that operate on that data into a single unit, or class.

- By implementing encapsulation, we can restrict direct access to some of an object's components, promoting controlled access and data protection.

CoGrammar

Encapsulation in OOP

- The core principle here is that we want to hide the internal state of an object and only allow modification through well-defined methods (getters and setters).

- Getters refer to methods that read data and setters refer to methods that update data.

Access Control - Attributes

- Access control mechanisms (public, protected, private) restrict or allow the access of certain attributes within a class.

```python
class MyClass:
    def __init__(self):
        # Public attribute
        self.public_attribute = "I am public"

        # Protected attribute (by convention)
        self._protected_attribute = "I am protected"

        # Private attribute
        self.__private_attribute = "I am private"
```

Access Control - Methods

- Access control mechanisms (public, protected, private) can also restrict or allow the access of certain methods within a class.

```python
def public_method(self):
    return "This is a public method"


def _protected_method(self):
    return "This is a protected method"


def __private_method(self):
    return "This is a private method"
```

CoGrammar

Applying the Access Control

```python
# Create an instance of MyClass
obj = MyClass()

# Accessing public attributes and methods
print(obj.public_attribute)      # Output: I am public
print(obj.public_method())       # Output: This is a public method

# Accessing protected attributes and methods (not enforced, just a convention)
print(obj._protected_attribute)  # Output: I am protected
print(obj._protected_method())   # Output: This is a protected method

# Accessing private attributes and methods (name mangling applied)
# Note: It's still possible to access, but it's discouraged
print(obj._MyClass__private_attribute)  # Output: I am private
print(obj._MyClass__private_method())   # Output: This is a private method
```

CoGrammar

Instance Methods

CoGrammar

Instance Methods

- Instance methods are like actions or behaviours that specific objects can perform.

- These methods have access to the object's data and are defined within the class.

- By using instance methods, we can model how objects interact and behave in our programs, making object-oriented programming a powerful way to structure our code.

CoGrammar

Instance Methods - Example

```python
class Student:
    def __init__(self, name):
        self.name = name

    def study(self):
        print(f"{self.name} is studying hard!")

# Creating a student object
student1 = Student("Alice")

# Calling the instance method
student1.study()
```

CoGrammar

Static Methods

Static Methods

- Static methods are like standalone functions that live within a class.

- They're handy for grouping together related functionality without needing to access specific instance or class data.

- You mark them with the '@staticmethod' decorator to let Python know they're special.

Static Methods - Example

```python
class Car:
    @staticmethod
    def honk():
        return "Beep beep!"

# Calling the static method
print(Car.honk())  # Output: Beep beep!
```

CoGrammar

Class Methods

Class Methods

- Class methods are like special functions that belong to the class itself.

- They're not tied to any particular instance but can do cool stuff with the class as a whole.

- You mark them with the '@classmethod' decorator and they get this fancy 'cls' parameter which stands for the class itself.

CoGrammar

Class Methods - Example

```python
class Car:
    num_cars_sold = 0  # Class variable to keep track of the number of cars sold

    def __init__(self, brand):
        self.brand = brand
        Car.num_cars_sold += 1  # Increment the number of cars sold when a new car is created

    @classmethod
    def get_num_cars_sold(cls):
        return cls.num_cars_sold

# Creating instances of the Car class
car1 = Car("Toyota")
car2 = Car("Honda")

# Accessing the class method to get the number of cars sold
print(Car.get_num_cars_sold())  # Output: 2
```

CoGrammar

Let's take a short break

CoGrammar

Demo Time!

CoGrammar

Conclusion and Recap

Conclusion and Recap

- **Classes**: Blueprints for creating objects that encapsulate both data (attributes) and functionality (methods).

- **Encapsulation**: Classes group related attributes and methods, promoting code organisation and reusability.

- **Instance Methods**: Operate on individual objects and can access and modify instance attributes.

- **Class Methods**: Operate on the class itself and can modify class-level data. Defined using the @classmethod decorator.

- **Static Methods**: General utility methods that don't depend on class or instance data. Defined using the @staticmethod decorator.

CoGrammar

Conclusion and Recap

Where You'll Use Classes and Methods:

- Web Development

- Data Analysis:

- Game Development:

CoGrammar

Classes and Methods Resources

- Official Python Documentation:
  - https://docs.python.org/3/tutorial/classes.html
  - https://docs.python.org/3/tutorial/classes.html#class-and-instance-variables
- Online Tutorials:
  - https://realpython.com/python3-object-oriented-programming/
  - https://realpython.com/instance-class-and-static-methods-demystified/
- Indently YouTube Channel
  - https://www.youtube.com/watch?v=PIKiHq1O9HQ

CoGrammar

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE — *SKILLS BOOTCAMPS*

Department for Education

CoGrammar