Welcome to this CoGrammar Lecture: Classes

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.

CoGrammar

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** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- 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

Skills Bootcamp
8-Week Progression Overview

Fulfil 4 Criteria to Graduation

✅ Criterion 1: Initial Requirements

Timeframe: First 2 Weeks
Guided Learning Hours (GLH): Minimum of 15 hours
Task Completion: First four tasks

Due Date: 24 March 2024

✅ Criterion 2: Mid-Course Progress

60 Guided Learning Hours

Data Science - **13 tasks**
Software Engineering - **13 tasks**
Web Development - **13 tasks**

Due Date: 28 April 2024

CoGrammar

Skills Bootcamp Progression Overview

✅ Criterion 3: Course Progress

Completion: All mandatory tasks, including Build Your Brand and resubmissions by study period end
Interview Invitation: Within 4 weeks post-course
Guided Learning Hours: Minimum of 112 hours by support end date (10.5 hours average, each week)

✅ Criterion 4: Demonstrating Employability

Final Job or Apprenticeship Outcome: Document within 12 weeks post-graduation
Relevance: Progression to employment or related opportunity

CoGrammar

SKILLS FOR LIFE

SKILLS BOOTCAMPS

Department for Education

CoGrammar Classes

April 2024

Learning objectives

- ❖ Functions Recap
- ❖ OOP
  - ➢ What is OOP ?
  - ➢ Classes in Python
  - ➢ Objects in Python
- ❖ Four Pillars of OOP

Functions Recap

CoGrammar

Functions Recap

Functions

- We can use python built-in functions or we can define our own functions with their own behaviours.

Parameters Variables

- We use parameter variables to receive input to use within the function.

Function Scope

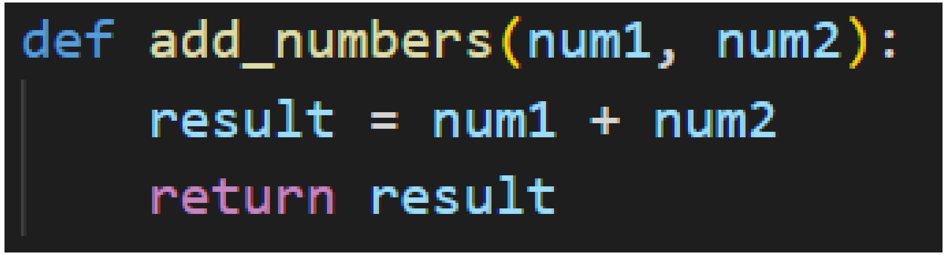
- Functions can use global variables but the main program can’t access variables within the function.

Return

- We can return data from a function using the ‘return’ keyword.

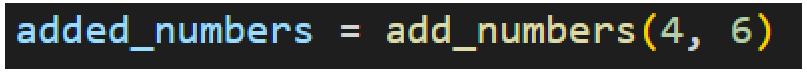
Function Recap cont.

Defining a Function



```python
def add_numbers(num1, num2):
    result = num1 + num2
    return result
```

Calling a Function



```python
added_numbers = add_numbers(4, 6)
```

CoGrammar

OOP

Object Oriented Programming

CoGrammar

What is Object Oriented Programming ?

OOP is a way of organizing code around objects, which are self-contained modules that contain both data and instructions that operate on that data.

Why use OOP ?

- OOP promotes encapsulation by bundling data and behaviour together within objects.
- OOP promotes abstraction by focusing on essential characteristics and behaviours of objects, hiding the underlying implementation details.
- OOP promotes code organisation into independent modules called classes. This separation of concerns allows developers to focus on specific tasks without worrying about the intricacies of other parts of the program.
- OOP reduces code duplication and simplifies development effort.

Different Types of Methodologies

- ❖ **Procedural Programming:**
  The focus is on writing procedures or routines that perform operations on the data.
- ❖ **Functional Programming (FP):**
  Treats computation as the evaluation of mathematical functions and avoids changing-state and mutable data.
- ❖ **Event-Driven Programming:**
  Focuses on the flow of the program being determined by events such as user actions, sensor outputs, or message passing from other programs.
- ❖ **Logic Programming:**
  Based on formal logic, a program is a set of sentences in logical form, expressing facts and rules about some problem domain.
- ❖ **Modular Programming:**
  Emphasizes separating the functionality of a program into independent, interchangeable modules

CoGrammar

Classes & Objects

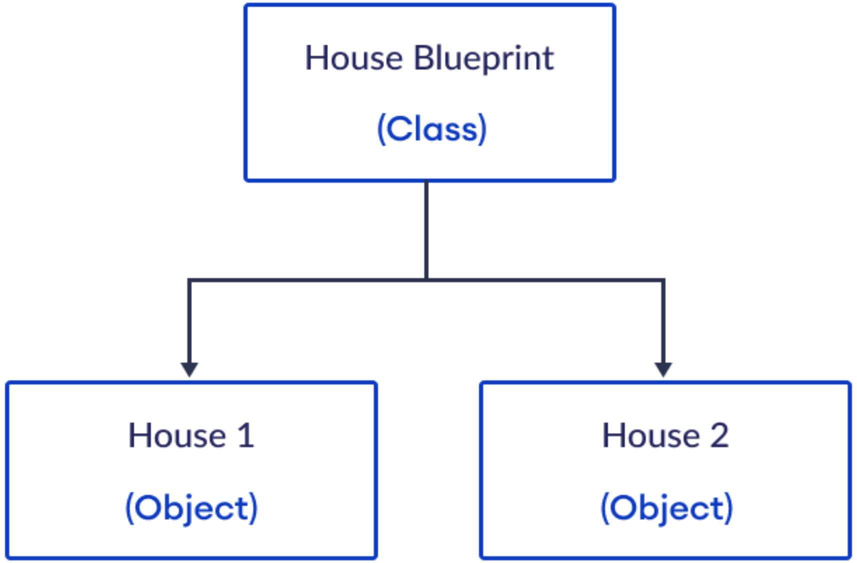
CoGrammar

What are classes ?

- ❖ A class in Python is like a blueprint for creating objects.
- ❖ It defines a set of attributes and methods that the created objects of the class can use.
- ❖ Attributes are the characteristics of an object, while methods are the operations that an object can perform.

CoGrammar

Example



CoGrammar

Class Properties

- Each class can have two main things: **attributes** and **methods**.

- **Attributes** are variables that belong to a class. They represent the properties or characteristics of the class that objects can have.

- **Methods** are functions that belong to a class. They define the behaviors or actions that an object of the class can perform.

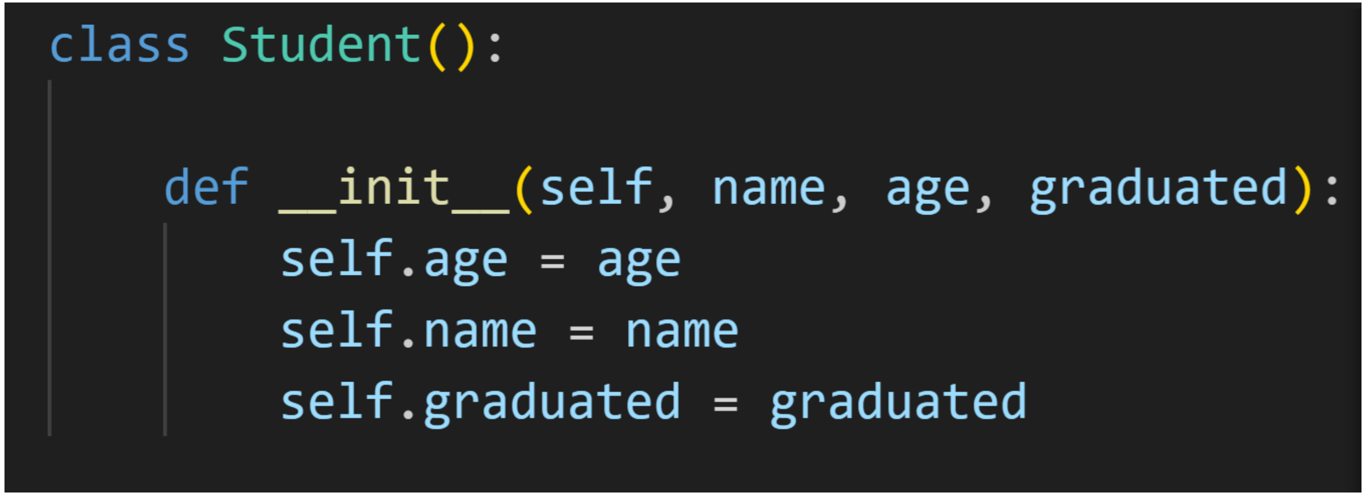
Attributes

- ❖ Attributes are values that define the characteristics associated with an object.
- ❖ They define the state of an object and provide information about its current condition.
- ❖ For a class named ‘House’, some relevant attributes could be:
  - ➢ number_of_bedrooms
  - ➢ year_built

CoGrammar

Class Attributes

❖ __init__ function is called when class is instantiated.



```python
class Student():

    def __init__(self, name, age, graduated):
        self.age = age
        self.name = name
        self.graduated = graduated
```

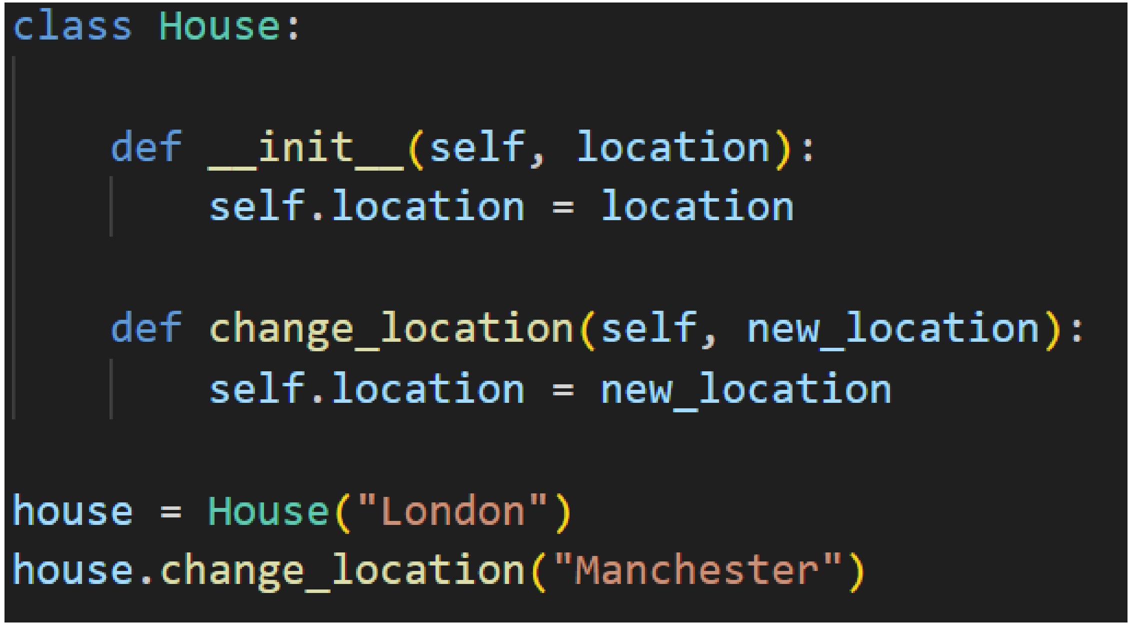
CoGrammar

Methods (Behaviours)

- ❖ Methods, define the actions or behaviors that objects can perform
- ❖ They encapsulate the functionality of objects and allow them to interact with each other and the outside world.
- ❖ For a class named ‘House’, some relevant method could be:
  - ➢ set_location(): Allows updating the location of the house

CoGrammar

Methods (Behaviours)



```python
class House:


    def __init__(self, location):
        self.location = location

    def change_location(self, new_location):
        self.location = new_location

house = House("London")
house.change_location("Manchester")
```

CoGrammar

What are Objects ?

- An object is a fundamental building block that represents a real-world entity or concept. It encapsulates both data and behaviour.

- Objects represent key characteristics or attributes of real world entities.

- Objects also encapsulate the actions or behaviours associated with real-world entities.

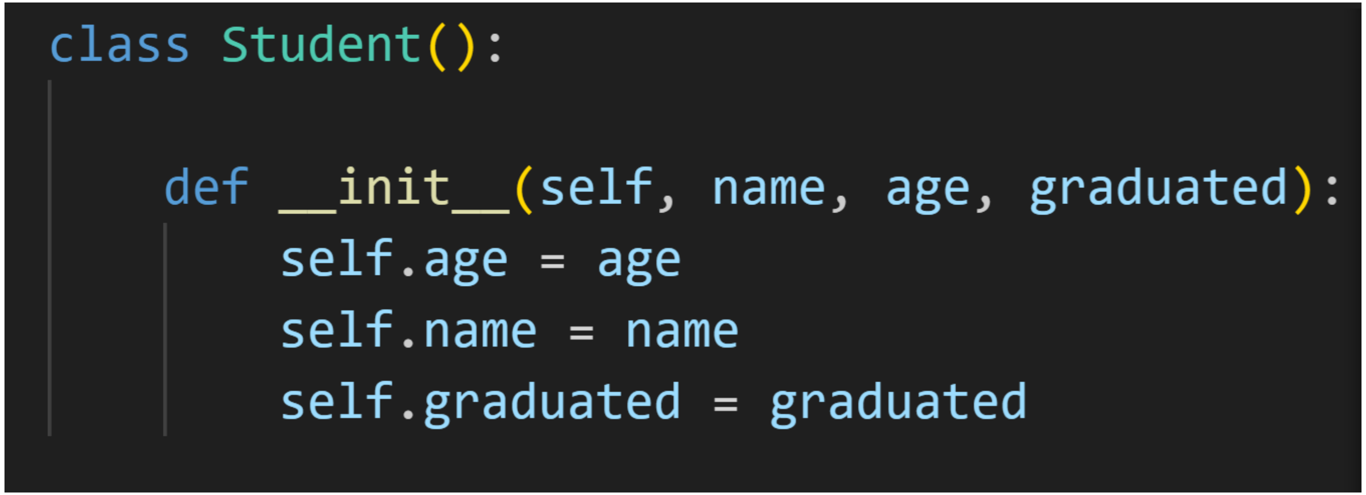
CoGrammar

Objects In Python

- In Python, everything is an object. Every entity, including data values and functions, are considered objects.
- They allow you to hide the internal implementation details of data and only expose methods for interacting with data.
- Without knowing it, you have actually been using objects in Python.
- For example: string.split() - this uses the split() method present in the string object.
- Imagine needing to call split(string, delimiter) - not as powerful of a notation!

CoGrammar

Attributes



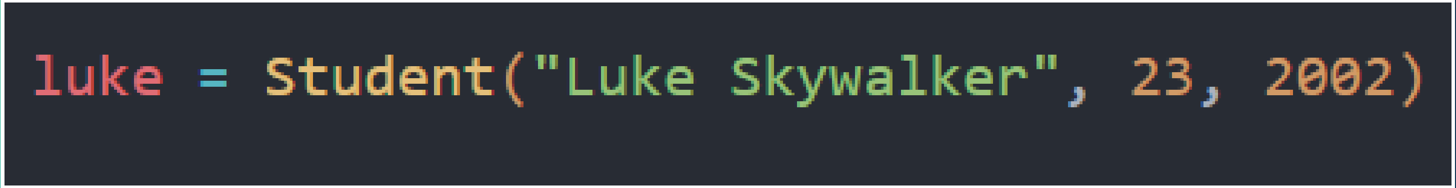
```python
class Student():

    def __init__(self, name, age, graduated):
        self.age = age
        self.name = name
        self.graduated = graduated
```

CoGrammar

Class Instantiation

Class takes in three values: a name, age and graduation year.



```
luke = Student("Luke Skywalker", 23, 2002)
```

CoGrammar

Let's take a break

CoGrammar

Pillars of OOP

Inheritance

- Inheritance allows us to define a class that inherits all the methods and properties from another class.

- Parent class is the class being inherited from, also called base class. Child class is the class that inherits from another class, also called derived class.

Polymorphism

- Polymorphism allows us to access these overridden methods and attributes that share the same name as the base class.

CoGrammar

Pillars of OOP

Encapsulation

- Encapsulation in Python is the concept of wrapping data (variables) and methods (functions) into a single unit (class) and restricting access to the direct modification of an object's attributes.

Abstraction

- Abstraction is the concept of hiding the complex implementation details and showing only the essential features of the object. In a way, it is an extension of encapsulation, focusing on hiding the complexity and only exposing a high-level interface.

Wrapping Up

Summary

Object Orientation in Programming

A way of organizing code around objects, which are self-contained modules that contain both data and instructions that operate on that data.

Classes/Objects in Python

We can create classes then create instance objects of the class that contain the same attributes as the classes we have created.

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE

SKILLS BOOTCAMPS

Department for Education

CoGrammar