Python OOP Concepts

Anoop S Babu
Faculty Associate
Dept. of Computer Science & Engineering
bsanoop@am.amrita.edu

What is OPP ?

- Object oriented programming
- Programming paradigm based on objects
- Model real world entities to an object
  - Which has some data and perform certain functions
- For example a car.
- A car has color, model, price etc.
- It performs functions – start, stop, break, shift gears, accelerate and more.

Classes in Python

- Class serves as a blue print for the entity.
- An entity has two characteristics:
  - Attribute
  - Behavior
- Class also defined with these two characteristics

Defining Class in Python

- Syntax

```
class ClassName:
    """Documentation string"""
    <statement-1>
    .
    .
    .
    <statement-N>
```

Defining Class in Python - Example

- Consider the class for the entity student.

```
class Student:
    """Base class to all Students"""
    def __init__(self,regno,name,program):
        self.regno = regno
        self.name = name
        self.program = program

    def dispStudentInfo(self):
        print("Register Number: ",self.regno)
        print("Name: ",self.name)
        print("Program: ",self.program)
```

Python Objects

- Instances of class
- Implements the class
- Required to use the data and methods defined in a class
- A class can have multiple objects.
- All the objects have their own copy of attributes

Creating Objects in Python

- Syntax

```
ObjectName = ClassName(attribute values)
```

- Example

```
S1 = Student(101,"Ananya","MCA")
S2 = Student(102,"Poorna","MCA")
```

Accessing methods using Objects

- Using the dot operator along with the object.
- Example

```
S1.dispStudentInfo()
S2.dispStudentInfo()
```

- Output

```
Register Number:  101
Name:  Ananya
Program:  MCA
Register Number:  102
Name:  Poorna
Program:  MCA
```

Attributes and Methods

Class Attributes

- Attribute common to all objects

Example

```
class Student:
    """Base class to all Students"""
    university = "Amrita Vishwa Vidyapeetham"
    def __init__(self,regno,name,program):
        self.regno = regno
        self.name = name
        self.program = program

    def dispStudentInfo(self):
        print("Register Number: ",self.regno)
        print("Name: ",self.name)
        print("Program: ",self.program)
        print("University:",self.university)

S1 = Student(101,"Ananya","MCA")
S2 = Student(102,"Poorna","MCA")

S1.dispStudentInfo()
S2.dispStudentInfo()
```

Output

```
Register Number:  101
Name:  Ananya
Program:  MCA
University: Amrita Vishwa Vidyapeetham
Register Number:  102
Name:  Poorna
Program:  MCA
University: Amrita Vishwa Vidyapeetham
```

Object Attributes

- Attribute specific to an object

Example

```
class Student:
    """Base class to all Students"""
    university = "Amrita Vishwa Vidyapeetham"
    def __init__(self,regno,name,program):
        self.regno = regno
        self.name = name
        self.program = program

    def dispStudentInfo(self):
        print("Register Number: ",self.regno)
        print("Name: ",self.name)
        print("Program: ",self.program)
        print("University:",self.university)

S1 = Student(101,"Ananya","MCA")
S2 = Student(102,"Poorna","MCA")

S1.dispStudentInfo()
S2.dispStudentInfo()
```

Output

```
Register Number:  101
Name:  Ananya
Program:  MCA
University: Amrita Vishwa Vidyapeetham
Register Number:  102
Name:  Poorna
Program:  MCA
University: Amrita Vishwa Vidyapeetham
```

Built-in Class Attributes

- __doc__ - contains the documentation string
- __name__ - contains the class name
- __module__ - contains the module name in which the class defined
- __dict__ - contains the dictionary containing class's namespace
- __bases__ - contains the tuple of base classes.

Built-in Class Attributes – Example

Program

```
class Student:
    """Base class to all Students"""
    university = "Amrita Vishwa Vidyapeetham"
    def __init__(self,regno,name,program):
        self.regno = regno
        self.name = name
        self.program = program

    def dispStudentInfo(self):
        print("Register Number: ",self.regno)
        print("Name: ",self.name)
        print("Program: ",self.program)
        print("University:",self.university)

print("Doc string: ",Student.__doc__)
print("Name: ",Student.__name__)
print("Module: ",Student.__module__)
print("Namespace: ",Student.__dict__)
print("Base classes: ", Student.__bases__)
```

Built-in Class Attributes – Example

Output

```
Doc string:  Base class to all Students
Name:  Student
Module:  __main__
Namespace:  {'__module__': '__main__', '__doc__': 'Base class to all Students', 'univ
ersity': 'Amrita Vishwa Vidyapeetham', '__init__': <function Student.__init__ at 0x00
0001AFC06A2700>, 'dispStudentInfo': <function Student.dispStudentInfo at 0x000001AFC0
6A2790>, '__dict__': <attribute '__dict__' of 'Student' objects>, '__weakref__': <att
ribute '__weakref__' of 'Student' objects>}
Base classes:  (<class 'object'>,)
```

Built-in Attribute methods

- getattr(obj,name,[default]) – returns the value of the attribute
- hasattr(obj,name) – returns True if attribute is present, else False
- setattr(obj,name,value) – create a new attribute, returns nothing
- delattr(obj,name) – delete an attribute, returns nothing

Built-in Attribute methods – Example

Program

```
class Student:
    """Base class to all Students"""
    def __init__(self,regno,name):
        self.regno = regno
        self.name = name

S1 = Student(101,"Ananya")
print("getattr(S1,'name'): ",getattr(S1,"name"))
print("hasattr(S1,'regno'): ",hasattr(S1,"regno"))
setattr(S1, "program","MCA")
print("setattr(S1, 'program','MCA'): ",S1.program)
delattr(S1,"regno")
print("hasattr(S1,'regno'): ",hasattr(S1,"regno"))
```

Built-in Attribute methods – Example

Output

```
getattr(S1,'name'):  Ananya
hasattr(S1,'regno'):  True
setattr(S1, 'program','MCA'):  MCA
hasattr(S1,'regno'):  False
```