CSE 2040 Programming IV Lecture #34

What will we learn today

- Abstract Method
- Abstract Class
- Interface

Abstract Method

- Abstract method is a method whose action is redefined in the sub classes as per the requirement of the objects.
- Generally abstract methods are written without body since their body will be defined in the sub classes anyhow.
  - But it is possible to write an abstract method with body also.
  - To mark a method as abstract, we should use the decorator @abstractmethod.
- On the other hand, a concrete method is a method with body.

Cont’d

- The way to create an abstract class is to derive it from a meta class ABC that belong to abc(abstract base class) module.

```
class Abstractclass(ABC):
```

- A meta class is a class that defines the behavior of other classes. The meta class ABC defines that the class which is derived from it becomes an abstract class.

```
from abc import ABC, abstractmethod
```

Cont’d

```
from abc import ABC, abstractmethod
class Myclass(ABC):
        @abstractmethod
        def calculate(self, x):
                pass
```

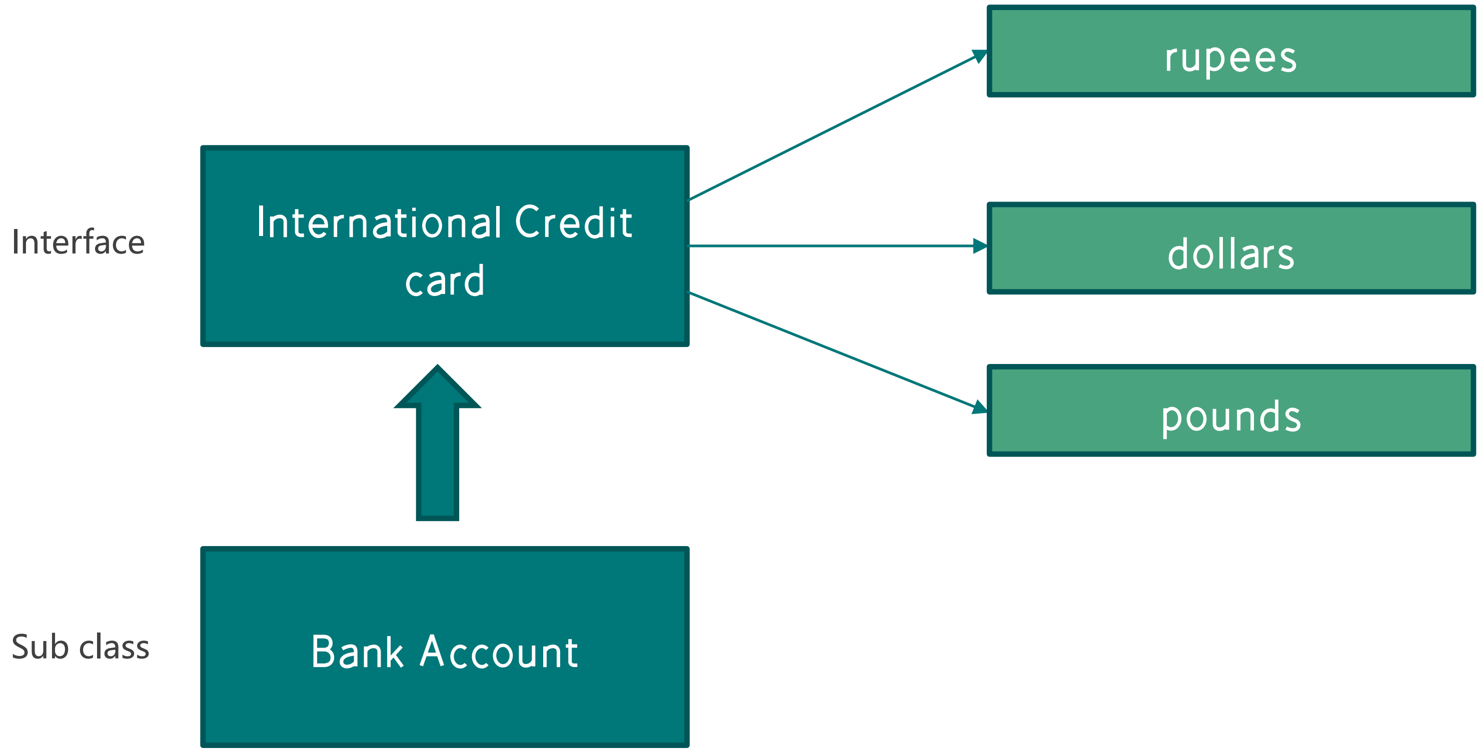
Cont'd

```
#to create abstract class and sub classes which implement the abstract method of the abstract class
from abc import ABC, abstractmethod

class Myclass(ABC):
    @abstractmethod
    def calculate(self,x):
        pass

class Sub1(Myclass):
    def calculate(self,x):
        print('square value: ',x*x)

import math
class Sub2(Myclass):
    def calculate(self,x):
        print('square root: ',math.sqrt(x))

class Sub3(Myclass):
    def calculate(self,x):
        print('cube value: ',x**3)

obj1=Sub1()
obj1.calculate(3)
obj2=Sub2()
obj2.calculate(3)
obj3=Sub3()
obj3.calculate(3)
```

Interface

- In the languages like java, an interface is created using the key word ‘interface’ but in python an interface is created as an abstract class only.
- The interface concept is not explicitly available in python.
- An interface contains methods without body, it is not possible to create objects to an interface.

Interface and sub class



Example program

- Reference to : interfaceexample1.py, interfaceexample2.py

Point to remember

- Python does not provide interface concept explicitly. It provides abstract classes which can be used as either abstract classes or interfaces.
- An abstract class is a class that contains some abstract methods. An abstract class can also contain concrete methods. It is not possible to create an object to an abstract class.
- An abstract class is written when there are some common features shared by all the objects.
- An interface is written when all the features are implemented differently for different objects.
- Both abstract classes and interfaces are example for polymorphism.

Let’s consider problem statement

- Retailer1, a class which represents a retail shop. Retailer1 wants text books of X class and some pens. Similarly, Retailer2 also wants text books of X class and some papers.
- In this case, we can understand that the text_books() is the common feature shared by both the retailers. But the stationary asked by the retailers is different. This means, the stationary has different implementations for different retailers but there is a common features, i.e., the text books.
- In this case, the programmer designs the WholeSaler class as an abstract class. Retailer1 and Retailer2 are sub classes.

References

- Dr. R. Nageswara Rao, Core Python Programming, Second Edition, 2018