9913\_exp4

March 2, 2024

[2]:

**class**

**Student**

:

**def**

\_\_init\_\_

(

self

, name, roll\_no):

self

.

name

=

name

self

.

roll\_no

=

roll\_no

**def**

setAge

(

self

,age):

self

.

age

=

age

**def**

setMarks

(

self

,marks):

self

.

marks

=

marks

**def**

display

(

self

):

print

(

f

"

The name of the student is

**{**

self

.

name

**}**

and roll.no is

**{**

self

.

↪

roll\_no

**}**

"

)

print

(

f

"

The student is

**{**

self

.

age

**}**

years old

"

)

print

(

f

"

Marks:

**{**

self

.

marks

**}**

"

)

student\_1

=

Student(

"

Mark

"

,

9913

)

student\_1

.

setAge(

19

)

student\_1

.

setMarks(

99

)

student\_1

.

display()

The name of the student is Mark and roll.no is 9913

The student is 19 years old

Marks: 99

[3]:

**class**

**Time**

:

**def**

\_\_init\_\_

(

self

, hours, minutes):

self

.

hours

=

hours

self

.

minutes

=

minutes

**def**

addtime

(

self

, time\_2):

self

.

hours

+

=

time\_2

.

hours

self

.

minutes

+

=

time\_2

.

minutes

**if**

self

.

minutes

>

=

60

:

self

.

hours

+

=

self

.

minutes

/

/

60

self

.

minutes

=

self

.

minutes

%

60

**def**

displayTime

(

self

):

print

(

f

"

**{**

self

.

hours

**}**

hrs

**{**

self

.

minutes

**}**

min

"

)

**def**

displayMinute

(

self

):

print

(

f

"

Time in minutes:

**{**

self

.

hours

\*

60

+

self

.

minutes

**}**

"

)

time1

=

Time(

2

,

80

)

time2

=

Time(

3

,

40

)

time1

.

displayTime()

time2

.

displayTime()

time1

.

addtime(time2)

time1

.

displayTime()

time1

.

displayMinute()

2

hrs 80 min

3

hrs 40 min

7 hrs 0 min

Time in minutes: 420

[5]:

**class**

**CartItem**

:

**def**

\_\_init\_\_

(

self

, product\_id, quantity, price):

self

.

product\_id

=

product\_id

self

.

quantity

=

quantity

self

.

price

=

price

**def**

total\_price

(

self

):

**return**

self

.

quantity

\*

self

.

price

**class**

**PromotionalItem**

(

CartItem

):

**def**

\_\_init\_\_

(

self

, product\_id, quantity, price, discount\_percent):

super

()

.

\_\_init\_\_

(

product\_id, quantity, price

)

*#sends the attributes of*

␣

↪

*promotionaitem to cartitem to be initialised*

self

.

discount\_percent

=

discount\_percent

/

100

*#convert discount to*

␣

↪

*decimal(10% = 0.1)*

**def**

total\_price

(

self

):

**return**

super

()

.

total\_price()

\*

(

1

-

self

.

discount\_percent)

*#calculate*

␣

↪

*price with discount by using the totalprice in cartitem(super)*

**class**

**RegularItem**

(

CartItem

):

**pass**

*#it has the same attributes as CartItem hence it is passed*

**class**

**ElectronicItem**

(

PromotionalItem, RegularItem

):

**pass**

*#same as above*

**class**

**ClothingItem**

(

PromotionalItem, RegularItem

):

**pass**

*#same as above*

electronic\_item

=

ElectronicItem(

1

,

10

,

500

,

10

)

clothing\_item

=

ClothingItem(

2

,

5

,

200

,

20

)

cart

=

[]

cart

.

append(electronic\_item)

*#adds electronic\_item to cart*

cart

.

append(clothing\_item)

*#adds clothing\_item to cart*

total\_price

=

0

**for**

item

**in**

cart:

total\_price

+

=

item

.

total\_price()

*#calculate every items price and it*

␣

↪

*simultaneously*

print

(

f

"

Total price:

**{**

total\_price

**}**

"

)

Total price: 5300.0

[6]:

**class**

**Vehicle**

:

**def**

\_\_init\_\_

(

self

, make, model, year):

self

.

make

=

make

self

.

model

=

model

self

.

year

=

year

**def**

display\_info

(

self

):

print

(

f

"

make:

**{**

self

.

make

**}**

, model:

**{**

self

.

model

**}**

, year:

**{**

self

.

year

**}**

"

)

**class**

**Car**

(

Vehicle

):

**pass**

**class**

**Truck**

(

Vehicle

):

**pass**

**class**

**ElectricVehicle**

(

Vehicle

):

**def**

\_\_init\_\_

(

self

, make, model, year, km\_travelled\_by\_battery):

super

()

.

\_\_init\_\_

(

make, model, year

)

*# Added parentheses*

self

.

km\_travelled\_by\_battery

=

km\_travelled\_by\_battery

**def**

charge\_battery

(

self

):

print

(

f

"

**{**

self

.

make

**}**

**{**

self

.

model

**}**

is charged to full capacity

"

)

**class**

**ElectricCar**

(

ElectricVehicle, Car

):

**def**

\_\_init\_\_

(

self

, make, model, year, km\_travelled\_by\_battery):

super

()

.

\_\_init\_\_

(

make, model, year, km\_travelled\_by\_battery

)

**class**

**ElectricTruck**

(

ElectricVehicle, Truck

):

**def**

\_\_init\_\_

(

self

, make, model, year, km\_travelled\_by\_battery):

super

()

.

\_\_init\_\_

(

make, model, year, km\_travelled\_by\_battery

)

tesla

=

ElectricCar(

"

Tesla

"

,

"

Model e99

"

,

2100

,

200

)

toyota

=

Car(

"

Toyota

"

,

"

Model 1

"

,

2010

)

electric\_truck

=

ElectricTruck(

"

eLorry

"

,

"

Model e1

"

,

2300

,

300

)

truck

=

Truck(

"

Lorry

"

,

"

Model 1

"

,

2000

)

tesla

.

display\_info()

toyota

.

display\_info()

electric\_truck

.

display\_info()

truck

.

display\_info()

print

(

"

**\n**

"

)

tesla

.

charge\_battery()

electric\_truck

.

charge\_battery()

make: Tesla, model: Model e99, year: 2100

make: Toyota, model: Model 1, year: 2010

make: eLorry, model: Model e1, year: 2300 make: Lorry, model: Model 1, year: 2000

Tesla Model e99 is charged to full capacity eLorry Model e1 is charged to full capacity

Postlab:-

Q1.

The code will print 5:30 because the print\_time method uses self.time to print the time attribute of the Clock instance, which is set to '5:30' during the object initialization. While the time variable is set to 6:30 which is not printed.

Q2.

1. The code will print '10:30' because the print\_time method uses the local variable time as its parameter, which is set to the value '10:30' when the method is called. The instance variable self.time is not used in the print\_time method.
2. Top of Form
3. This example illustrates that using the same name for method parameters as object attributes (instance variables) can lead to shadowing. In the print\_time method, the local variable time shadows the instance variable self.time. This practice can potentially cause confusion and unexpected behavior. It emphasizes the importance of choosing distinct names for method parameters to avoid conflicts with object attributes.

Q3

1. The code will print '10:30' because paris\_clock is assigned the reference to the same object as boston\_clock, and when the time attribute is updated through paris\_clock, it directly affects the underlying object, which is then printed using boston\_clock.print\_time().

b) This happens because boston\_clock and paris\_clock are not different objects; they both refer to the same instance of the Clock class. When you assign one variable to another in Python (paris\_clock = boston\_clock), both variables point to the same object in memory. As a result, modifying the object through one variable reflects the changes when accessed through the other variable.