Python

Class Smartphone:

Def \_init\_(self, brand, model, storage, battery\_life):

Self.brand = brand

Self.model = model

Self.storage = storage # in GB

Self.battery\_life = battery\_life # in hours

Def make\_call(self, number):

Print(f”Calling {number} from {self.model}...”)

Def get\_specs(self):

Return f”{self.brand} {self.model} – {self.storage}GB, {self.battery\_life}h battery”

Example usage

Phone = Smartphone(“Samsung”, “Galaxy S21”, 128, 24)

Print(phone.get\_specs())

Phone.make\_call(“+2348012345678”)

P ython

Class Smartphone:

Def \_init\_(self, brand, model, storage, battery\_life, os, camera\_mp):

Self.brand = brand

Self.model = model

Self.storage = storage # in GB

Self.available\_storage = storage

Self.battery\_life = battery\_life # in hours

Self.os = os

Self.camera\_mp = camera\_mp

Self.battery\_level = 100 # percent

Self.installed\_apps = []

Def install\_app(self, app\_name, size):

If size > self.available\_storage:

Print(f”Not enough storage to install {app\_name}.”)

Else:

Self.installed\_apps.append(app\_name)

Self.available\_storage -= size

Print(f”{app\_name} installed successfully.”)

Def charge(self):

Self.battery\_level = 100

Print(“Battery fully charged.”)

Def use(self, hours):

Battery\_usage = (hours / self.battery\_life) \* 100

If battery\_usage > self.battery\_level:

Print(“Battery too low. Please charge.”)

Else:

Self.battery\_level -= battery\_usage

Print(f”Used for {hours} hours. Battery now at {self.battery\_level:.1f}%.”)

Def take\_photo(self):

Print(f”Photo taken with {self.camera\_mp}MP camera.”)

Def get\_status(self):

Return (

F”{self.brand} {self.model} | OS: {self.os}\n”

F”Storage: {self.storage}GB (Available: {self.available\_storage}GB)\n”

F”Battery: {self.battery\_level:.1f}% | Camera: {self.camera\_mp}MP\n”

F”Apps: {‘, ‘.join(self.installed\_apps) if self.installed\_apps else ‘None’}”

Python

Class Smartphone:

Def \_init\_(self, brand, model, storage, battery\_life, os, camera\_mp):

Self.brand = brand

Self.model = model

Self.storage = storage

Self.available\_storage = storage

Self.battery\_life = battery\_life

Self.os = os

Self.camera\_mp = camera\_mp

Self.battery\_level = 100

Self.installed\_apps = []

Def get\_info(self):

Return (

F”{self.brand} {self.model} | OS: {self.os} | Storage: {self.storage}GB | “

F”Camera: {self.camera\_mp}MP | Battery: {self.battery\_life}hrs”

)

Creating multiple smartphones with unique values

Phone1 = Smartphone(“Samsung”, “Galaxy S22”, 128, 24, “Android”, 50)

Phone2 = Smartphone(“Apple”, “iPhone 13”, 256, 20, “iOS”, 12)

Phone3 = Smartphone(“Google”, “Pixel 6”, 128, 25, “Android”, 50)

Display info for each

Print(phone1.get\_info())

Print(phone2.get\_info())

Print(phone3.get\_info())

Python

Class Smartphone:

Def \_init\_(self, brand, model, storage, battery\_life, os, camera\_mp):

Self.brand = brand

Self.model = model

Self.storage = storage

Self.\_battery\_level = 100 # Encapsulated attribute (protected)

Self.battery\_life = battery\_life

Self.os = os

Self.camera\_mp = camera\_mp

Self.installed\_apps = []

Def get\_info(self):

Return (

F”{self.brand} {self.model} | OS: {self.os} | Storage: {self.storage}GB | “

F”Camera: {self.camera\_mp}MP | Battery Life: {self.battery\_life}hrs”

)

Def use\_battery(self, hours):

“””Decrease battery level but not below 0”””

Self.\_battery\_level = max(self.\_battery\_level – hours \* 10, 0)

Def battery\_status(self):

Return f”Battery level: {self.\_battery\_level}%”

Subclasses with polymorphism on get\_info()

Class Samsung(Smartphone):

Def get\_info(self):

Return f”Samsung Device – {super().get\_info()}”

Class Apple(Smartphone):

Def get\_info(self):

Return f”Apple Device – {super().get\_info()}”

Def get\_info(self):

Return f”Google Device – {super().get\_info()}”

Usage

Phone1 = Samsung(“Samsung”, “Galaxy S22”, 128, 24, “Android”, 50)

Phone2 = Apple(“Apple”, “iPhone 13”, 256, 20, “iOS”, 12)

Phone3 = Google(“Google”, “Pixel 6”, 128, 25, “Android”, 50)

Phones = [phone1, phone2, phone3]

For phone in phones:

Print(phone.get\_info())

Phone.use\_battery(3) # use 3 hours battery

Print(phone.battery\_status())

Print(“---“)

Python

Class Animal:

Def move(self):

Print(“The animal moves in its own way.”)

Class Dog(Animal):

Def move(self):

Print(“Dog runs 🐕”)

Class Bird(Animal):

Def move(self):

Print(“Bird flies 🦅”)

Class Vehicle:

Def move(self):

Print(“The vehicle moves in its own way.”)

Class Car(Vehicle):

Def move(self):

Print(“Car is driving 🚗”)

Class Plane(Vehicle):

Def move(self):

Print(“Plane is flying ✈️”)

List of different objects

Objects = [Dog(), Bird(), Car(), Plane()]

For obj in objects:

Obj.move()