**Coffee\_maker.py**

**class CoffeeMaker:**

"""Models the machine that makes the coffee"""

def \_\_init\_\_(self):

self.resources = {

"water": 300,

"milk": 200,

"coffee": 100,

}

def report(self):

"""Prints a report of all resources."""

print(f"Water: {self.resources['water']}ml")

print(f"Milk: {self.resources['milk']}ml")

print(f"Coffee: {self.resources['coffee']}g")

def is\_resource\_sufficient(self, drink):

"""Returns True when order can be made, False if ingredients are insufficient."""

can\_make = True

for item in drink.ingredients:

if drink.ingredients[item] > self.resources[item]:

print(f"Sorry there is not enough {item}.")

can\_make = False

return can\_make

def make\_coffee(self, order):

"""Deducts the required ingredients from the resources."""

for item in order.ingredients:

self.resources[item] -= order.ingredients[item]

print(f"Here is your {order.name} ☕️. Enjoy!")

**menu.py**

**class MenuItem:**

"""Models each Menu Item."""

def \_\_init\_\_(self, name, water, milk, coffee, cost):

self.name = name

self.cost = cost

self.ingredients = {

"water": water,

"milk": milk,

"coffee": coffee

}

class Menu:

"""Models the Menu with drinks."""

def \_\_init\_\_(self):

self.menu = [

MenuItem(name="latte", water=200, milk=150, coffee=24, cost=2.5),

MenuItem(name="espresso", water=50, milk=0, coffee=18, cost=1.5),

MenuItem(name="cappuccino", water=250, milk=50, coffee=24, cost=3),

]

def get\_items(self):

"""Returns all the names of the available menu items"""

options = ""

for item in self.menu:

options += f"{item.name}/"

return options

def find\_drink(self, order\_name):

"""Searches the menu for a particular drink by name. Returns that item if it exists, otherwise returns None"""

for item in self.menu:

if item.name == order\_name:

return item

print("Sorry that item is not available.")

**money\_machine.py**

**class MoneyMachine:**

CURRENCY = "$"

COIN\_VALUES = {

"quarters": 0.25,

"dimes": 0.10,

"nickles": 0.05,

"pennies": 0.01

}

def \_\_init\_\_(self):

self.profit = 0

self.money\_received = 0

def report(self):

"""Prints the current profit"""

print(f"Money: {self.CURRENCY}{self.profit}")

def process\_coins(self):

"""Returns the total calculated from coins inserted."""

print("Please insert coins.")

for coin in self.COIN\_VALUES:

self.money\_received += int(input(f"How many {coin}?: ")) \* self.COIN\_VALUES[coin]

return self.money\_received

def make\_payment(self, cost):

"""Returns True when payment is accepted, or False if insufficient."""

self.process\_coins()

if self.money\_received >= cost:

change = round(self.money\_received - cost, 2)

print(f"Here is {self.CURRENCY}{change} in change.")

self.profit += cost

self.money\_received = 0

return True

else:

print("Sorry that's not enough money. Money refunded.")

self.money\_received = 0

return False

**MAIN.py**

from menu import Menu

from coffee\_maker import CoffeeMaker

from money\_machine import MoneyMachine

money\_machine = MoneyMachine()

coffee\_maker = CoffeeMaker()

menu = Menu()

is\_on = True

while is\_on:

options = menu.get\_items()

choice = input(f"What would you like? ({options}): ")

if choice == "off":

is\_on = False

elif choice == "report":

coffee\_maker.report()

money\_machine.report()

else:

drink = menu.find\_drink(choice)

if coffee\_maker.is\_resource\_sufficient(drink) and money\_machine.make\_payment(drink.cost):

coffee\_maker.make\_coffee(drink)