Write a python program to assist a hotel manager in offering rooms to customers based on their preferences. The system will allow customers to specify their preferences (e.g., room type, budget, amenities) and then offer them a list of available rooms that meet those preferences.

**Hint: Consider the following to develop the python code:**

1. **Room**: This class will represent individual rooms with their properties (e.g., room type, price, amenities, etc.).
2. **Hotel**: This class will represent the hotel, which contains a list of available rooms.
3. **Customer**: This class will represent a customer and contain their preferences for room selection.

**Python Code for the Room Offering System:**

class Room:

def \_\_init\_\_(self, room\_number, room\_type, price, amenities):

self.room\_number = room\_number

self.room\_type = room\_type

self.price = price

self.amenities = amenities # List of amenities (e.g., Wi-Fi, air conditioning)

def \_\_repr\_\_(self):

return f"Room {self.room\_number}: {self.room\_type}, Price: ${self.price}, Amenities: {', '.join(self.amenities)}"

class Hotel:

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

self.name = name

self.rooms = [] # List to store available rooms

def add\_room(self, room):

self.rooms.append(room)

def get\_available\_rooms(self):

return self.rooms

class Customer:

def \_\_init\_\_(self, budget, preferred\_room\_type, required\_amenities):

self.budget = budget

self.preferred\_room\_type = preferred\_room\_type

self.required\_amenities = required\_amenities

def filter\_rooms(self, rooms):

matching\_rooms = [

room for room in rooms

if room.price <= self.budget

and (self.preferred\_room\_type.lower() in room.room\_type.lower() or self.preferred\_room\_type == "")

and all(amenity in room.amenities for amenity in self.required\_amenities)

]

return matching\_rooms

def offer\_room(self, hotel):

available\_rooms = hotel.get\_available\_rooms()

filtered\_rooms = self.filter\_rooms(available\_rooms)

if not filtered\_rooms:

print("No rooms match your criteria. Please adjust your preferences.")

else:

print(f"\nHere are the rooms that match your preferences (Price <= ${self.budget}, Preferred Room Type: {self.preferred\_room\_type}, Amenities: {', '.join(self.required\_amenities)}):")

for room in filtered\_rooms:

print(room)

# Example usage:

# Create a hotel

hotel = Hotel("Ocean View Hotel")

# Add rooms to the hotel

hotel.add\_room(Room(101, "Single", 100, ["Wi-Fi", "Air Conditioning", "TV"]))

hotel.add\_room(Room(102, "Double", 150, ["Wi-Fi", "Air Conditioning", "TV", "Minibar"]))

hotel.add\_room(Room(103, "Suite", 250, ["Wi-Fi", "Air Conditioning", "TV", "Minibar", "Ocean View"]))

hotel.add\_room(Room(104, "Single", 90, ["Wi-Fi", "TV"]))

hotel.add\_room(Room(105, "Double", 200, ["Wi-Fi", "Air Conditioning", "TV", "Minibar", "Balcony"]))

# Customer inputs preferences

print("Welcome to the hotel! Please provide your preferences:\n")

try:

budget = float(input("Enter your budget (in USD): $"))

preferred\_room\_type = input("Enter your preferred room type (Single, Double, Suite, or leave blank for any): ")

required\_amenities = input("Enter the amenities you require (comma separated, e.g., Wi-Fi, TV): ").split(",")

required\_amenities = [amenity.strip() for amenity in required\_amenities]

# Create a customer object

customer = Customer(budget, preferred\_room\_type, required\_amenities)

# Offer rooms based on customer's preferences

customer.offer\_room(hotel)

except ValueError:

print("Invalid input! Please enter valid numbers for budget.")